Pablo Moscato · Natalie Jane de Vries
Editors

Business and Consumer Analytics: New Ideas

Springer

Business and Consumer Analytics: New Ideas

Pablo Moscato • Natalie Jane de Vries
Editors

Business and Consumer Analytics: New Ideas

Springer

Editors
Pablo Moscato
School of Electrical Engineering
and Computing
The University of Newcastle
Callaghan, NSW, Australia

Natalie Jane de Vries
School of Electrical Engineering
and Computing
The University of Newcastle
Callaghan, NSW, Australia

ISBN 978-3-030-06221-7 ISBN 978-3-030-06222-4 (eBook)
https://doi.org/10.1007/978-3-030-06222-4

Library of Congress Control Number: 2019934369

Preface

It is hard to find something else to say about data science today that has not already been said. Perhaps what has not yet been discussed is that the progress in this area is profoundly challenging the way concepts have to be presented to newcomers. Data science brings together concepts from disciplines like computer science, statistics and applied mathematics, and the applications reach all possible aspects of life and economics. Consequently, universities around the world are having difficulties in addressing the need for a reformulation of their courses. There is a clear need for new ways to present the material, with an emphasis in understanding the key concepts, the novel applications and the impact of the techniques. This is, undoubtedly, a huge endeavour as there is no universally established curriculum for data science. In fact, we feel there is a need for students and practitioners who have been trained in one particular field to find a "shortcut" that would allow them to understand other areas.

This book can be seen as a first step in that direction. We aim at "bridging the gap" between some core new ideas in data science with the application in business and consumer analytics.

There are several reasons to choose this intersection as our first target. Advances in data science, data analytics and data mining methodologies are bringing many novel contributions to business and marketing applications. On the other side, the scale of e-commerce activities and the possibility of reaching a novel understanding of consumer behaviour are a driving force that pushes and challenges the field of data science. It is clear that the trend is here to stay. Conducting business and speaking out to consumers will be forever paired with data analytics. We have gained an incredible capability for collecting large amounts of widely varied data, and providing business insights from these data sources has become an important and continuous task of many researchers and business professionals.

At the time of writing this work, all the techniques included are considered novel in the area of business and consumer analytics. We are sure that more work needs to be done for many of them to reach the scalability necessary to deal with datasets of millions of consumers and products. That is a necessity of daily operations of many companies; we feel that this need for scalability will be met by the natural algorithmic progress provided by computer science. This said, we are interested in the new ideas currently at the crossroads between developments in data science, optimization, network analytics, computational complexity, artificial intelligence and machine learning, evolutionary algorithms and their application to business scenarios. We are confident that many of these techniques will soon flourish and become more widely adopted by businesses.

There are several lessons that are normally learnt when you finish a book. For the two editors, some of them were early ones at the beginning of the work, and they, in turn, helped to mould the introductory section to address them. For instance, the overzealous preoccupation of computer scientists with the provision of highly predictive analytic learning systems often clashes with the interest of business professionals. The latter tend to prefer models with perhaps less variables, at a cost of having a reduced predictive capability, that nevertheless contain the necessary levers that can lead to improved decision-making, reducing cost and maximizing profit. For both sides of the intersection we are looking at (i.e., data scientists and business professionals), even the word "problem" conveys a different meaning, and the quest of "efficiency" in computer science may be misinterpreted by a business marketer. We also aimed at presenting some fundamentals on marketing and consumer behaviour to enlighten the "hardcore computer scientists" about some of the needs and wants of marketing and business professionals. We quote: *"Data has been king for well over a decade by now, but the way we use it is undergoing some serious change. Gone are the days of awe at pretty charts and heat maps. Gone, too, is any patience for analytics unaligned to action"*.[1] These differences in the use of language and purpose are discussed in a tutorial way to help engage both communities towards a common goal. Once again, although we are certain that more needs to be done, our intention was clear; we tried to fill this gap.

Apart from the two introductory chapters, the rest of the book is organized as follows. Each section is centred in one specific area of fast development, which is either methodological or application based. The clustering and pattern mining section contains recent developments that are important for customer segmentation and targeting. They go from an introductory tone (Chaps. 3 and 4 are of that type), while the other two chapters relate to more advanced methods currently under development.

The network section contains a review chapter, which again is of an introductory nature, followed by an introduction to the area of centrality analysis (which is a main topic for areas of computational social science and product analyses). The remaining four chapters provide a variety of methods and applications related to network analytics including novel applications in survey analyses and co-purchasing network analytics.

In the 1980s, a new word was coined by Fred Glover called "metaheuristics", which are generic techniques aimed at guiding heuristics for a problem at hand

[1]https://www.information-management.com/opinion/data-science-underlies-everything-the-enterprise-now-does.

which can be posed as an optimization task. Analogously, he is proposing the use of the word "meta-analytics" for identifying high-level techniques aiming at guiding a multitude of methods working together to address a data analytic problem. These ideas are introduced, and then several chapters are dedicated to specific implementations of these methodologies: two with a focus in ensemble learning and one in a classification problem.

Memetic algorithms are a paradigm that was championed by one of the editors, and during the last years, it has become a highly active field of research following the publication of the *Handbook of Memetic Algorithms* by Springer in 2011. While applications in many fields of science and technology exist, there is an enormous potential for the area of business and consumer analytics. The chapters included in this work are just the tip of the iceberg of the activity that currently exists. It includes applications in product and customer visualization, personalized recommendation systems, facility location and vehicle routing, and orienting problems. We also note that one of the contributions in the network section includes a memetic algorithm application addressing overlapping community identification in co-purchasing graphs from the Amazon group.

The final section includes a more application-oriented major theme around travel and fashion analytics with three contributions in tourism and one in recommendation systems for a fashion e-commerce service. While the major theme is on personalization of services, the chapters are self-contained and methodologically address the questions using different techniques (fuzzy clustering, mathematical programming and ranking-based techniques).

We have tried, whenever possible, to use the same dataset in some of the different chapters. This allows readers to understand the possibilities that the different techniques gave individually and to concentrate the descriptions of all the datasets in a single Appendix chapter at the end of this book.

Callaghan, NSW, Australia

Pablo Moscato
Natalie Jane de Vries

Acknowledgements

The editors would like to thank a multitude of people who helped create this interdisciplinary and collaborative piece of work. Firstly, our thanks go to Luke Mathieson, Sergio Gomez, Carlos Cotta and Fred Glover who helped us to put together the various sections in this book by inviting fellow collaborators and sharing co-edition. We would like to thank all of the reviewers and proofreaders of the many contributions in this volume, those who did so anonymously and those who did so for fellow colleagues. Further, we would like to thank Ademir Cristiano Gabardo and Mohammad Nazmul Haque for their help in a range of things from formatting to technical issues and editing figures.

Finally, the editors would like to thank all of the *data parasites*[2,3] around the world who, like us, strive for a transparent and open research environment where collaboration, not competition, is the main motivation for researchers sharing data and where we strive towards the common goal of the advancement of science and human knowledge. We humbly dedicate our "Open Access Data" Appendix chapter to you!

Natalie de Vries would like to thank her family, friends and partner Simon Faint for their personal support of all her professional and academic endeavours. She would also like to thank fellow editor Pablo Moscato for the great experience and memories gained from creating this book together.

Pablo Moscato thanks Regina and Anna Clara for their patience and the support through the many stages of this project. He also thanks his fellow co-editor,

[2]*Data parasites* is a term that came to be popular after a rather controversial editorial published in *The New England Journal of Medicine* (NEJM) in January 2016 (http://www.nejm.org/doi/full/10.1056/NEJMe1516564) criticized the sharing of data. This editorial received a huge backlash from the research community who are in support of open access trends and data sharing activities. The satirical "Data Parasite Award" has since also been established handing out two awards: "the first recognizes an outstanding contribution from a junior parasite (postdoctoral, graduate, or undergraduate trainee), and the second recognizes an individual for a sustained period of exemplary research parasitism".

[3]http://researchparasite.com/.

Ms. de Vries, for being essential in their “largest catch” together. He also considers that, while a few cats, but no sharks, slowed the team to get the fish, not even “the great DiMaggio who does all things perfectly” could have been a better partner in this boat.

Looking back, and measuring all the obstacles to their projects created by some people over the last years, the editors say: “*‘Eat that galanos. And make a dream you’ve killed us’*. This book is also dedicated to you.”

Contents

Contributors

James Abello DIMACS and Computer Science Department, Rutgers University, New Brunswick, NJ, USA

Thomas Almenningen Department of Computer and Information Science, Norwegian University of Science and Technology, Trondheim, Norway

Joseph Andria Dipartimento di Scienze Economiche, Aziendali e Statistiche, University of Palermo, Palermo, Italy

Regina Berretta School of Electrical Engineering and Computing, The University of Newcastle, Callaghan, NSW, Australia

Benjamin Biesinger Institute of Computer Graphics and Algorithms, TU Wien, Vienna, Austria

Massimo Cafaro University of Salento, Lecce, Italy

Buyang Cao China Intelligent Urbanization Co-Creation Center for High Density Region, School of Software Engineering, Tongji University, Shanghai, China

Jamie Carlson Newcastle University Business School, The University of Newcastle, Callaghan, NSW, Australia

Carlos Cotta ETSI Informática, Universidad De Màlaga, Malaga, Spain

Natalie Jane de Vries School of Electrical Engineering and Computing, The University of Newcastle, Callaghan, NSW, Australia

Giacomo di Tollo Dipartimento di Economia, Universitá Ca' Foscari, Venezia, Italy

Marta Disegna Accounting, Finance & Economics Department, Faculty of Management, Bournemouth University, Bournemouth, UK

Pierpaolo D'Urso Department of Social Sciences and Economics, Sapienza University of Roma, Roma, Italy

Ademir Gabardo School of Electrical Engineering and Computing, The University of Newcastle, Callaghan, NSW, Australia

Fred Glover Leeds School of Business and the College of Engineering & Applied Science, University of Colorado, Boulder, CO, USA

Antonia Godoy-Lorite Department of Mathematics, Imperial College London, London, UK

Sergio Gómez Universitat Rovira i Virgili, Tarragona, Spain

Roger Guimerà Institució Catalana de Recerca i Estudis Avançats (ICREA), Barcelona, Catalonia, Spain

Departament d'Enginyeria, Química, Universitat Rovira i Virgili, Tarragona, Catalonia, Spain

Mohammad Nazmul Haque School of Electrical Engineering and Computing, The University of Newcastle, Callaghan, NSW, Australia

Martin Havig Department of Computer and Information Science, Norwegian University of Science and Technology, Trondheim, Norway

Bin Hu AIT Austrian Institute of Technology, Mobility Department - Dynamic Transportation Systems, Vienna, Austria

Mario Inostroza-Ponta Universidad de Santiago de Chile, Santiago, Chile

Helge Langseth Department of Computer and Information Science, Norwegian University of Science and Technology, Trondheim, Norway

Hoong Chuin Lau Singapore Management University, School of Information Systems, Singapore, Singapore

Stefan Lessmann Humboldt-Universität zu Berlin, School of Business and Economics, Berlin, Germany

Magdalene Marinaki Technical University of Crete, School of Production Engineering and Management, Chania, Greece

Yannis Marinakis Technical University of Crete, School of Production Engineering and Management, Chania, Greece

Riccardo Massari Department of Social Sciences and Economics, Sapienza University of Roma, Roma, Italy

Luke Mathieson School of Electrical Engineering and Computing, The University of Newcastle, Callaghan, NSW, Australia

School of Software, University of Technology Sydney, Ultimo, NSW, Australia

Nikolaos Matsatsinis Technical University of Crete, School of Production Engineering and Management, Chania, Greece

Daniel Mawhirter Colorado School of Mines, Golden, CO, USA

Mustafa Mısır Nanjing University of Aeronautics and Astronautics, College of Computer Science and Technology, Nanjing, Jiangsu, China

Pablo Moscato School of Electrical Engineering and Computing, The University of Newcastle, Callaghan, NSW, Australia

Hai Thanh Nguyen Department of Computer and Information Science, Norwegian University of Science and Technology, Trondheim, Norway

Telenor Research, Trondheim, Norway

Łukasz P. Olech Department of Computational Intelligence, Faculty of Computer Science and Management, Wrocław University of Science and Technology, Wrocław, Poland

Raffaele Pesenti Dipartimento di Management, Universitá Ca' Foscari, Venezia, Italy

Marco Pulimeno University of Salento, Lecce, Italy

Günther R. Raidl Institute of Computer Graphics and Algorithms, TU Wien, Vienna, Austria

Heri Ramampiaro Department of Computer and Information Science, Norwegian University of Science and Technology, Trondheim, Norway

Cesar Rego School of Business Administration, University of Mississippi, Oxford, MS, USA

Marta Sales-Pardo Departament d'Enginyeria Química, Universitat Rovira i Virgili, Tarragona, Spain

Claudio Sanhueza Lobos School of Electrical Engineering and Computing, The University of Newcastle, Callaghan, NSW, Australia

Herman Schistad Department of Computer and Information Science, Norwegian University of Science and Technology, Trondheim, Norway

Kevin Sun Rutgers University, New Brunswick, NJ, USA

Ringolf Thomschke Humboldt-Universität zu Berlin, School of Business and Economics, Berlin, Germany

Dimitra Trachanatzi Technical University of Crete, School of Production Engineering and Management, Chania, Greece

Eleftherios Tsakirakis Technical University of Crete, School of Production Engineering and Management, Chania, Greece

Stefan Voß University of Hamburg, Institute of Information Systems, Hamburg, Germany

Part I
Introduction

Chapter 1
Marketing Meets Data Science: Bridging the Gap

Pablo Moscato and Natalie Jane de Vries

Abstract It is certain that computer science is completely reformulating the way that business is being conducted around the world. We are witnessing the increasing availability of large volumes of data together with the advances in artificial intelligence, machine learning and optimization techniques. Breakthroughs in statistics, discrete applied mathematics and new algorithms are leading to the development of a new interdisciplinary field: data science. The purpose of this chapter is to provide a bridge, a short-cut to understand some of the questions that computer science deals with in a context of developing new techniques to get knowledge from data.

Keywords Analytics · Marketing and customer behaviour analytics · Data mining · Marketing

1.1 Data Science for Marketing

Once upon a time…things were a bit simpler. Indeed. Several decades ago, when computers started to be used for marketing and business intelligence, they were mainly used to analyse surveys, evaluate simple statistics or to produce interactive displays to compute the results of some mathematical models that humans would create [169]. That was a perfect marriage. The different disciplines seemed to complement each other and would not need anything more, they satisfied the mutual requirements very well and a natural symbiosis occurred. Marketers and business analysts would first identify a few things to build a case for a quantitative study and would then ask statisticians' help. Typically it would be a triad consisting of: (a) an assumedly important problem would be identified, (b) a solution that can be implemented would need to be evaluated if introduced, (c) an expected outcome

P. Moscato (✉) · N. J. de Vries
School of Electrical Engineering and Computing, The University of Newcastle, Callaghan, NSW, Australia
e-mail: Pablo.Moscato@newcastle.edu.au; natalie.devries@newcastle.edu.au

P. Moscato, N. J. de Vries (eds.), *Business and Consumer Analytics: New Ideas*,
https://doi.org/10.1007/978-3-030-06222-4_1

would benefit the business. If this triad can lead to a testable experiment, statistics would help to analyse the data, to draw some conclusions and the whole process would provide some business insight.

In fact, nobody can argue against the use of statistics, they bring a necessary quantitative dimension which is indispensable for competitiveness. However, there are some limitations. For decision-making some people cite a few common problems with such ad hoc experimentation: small sample sizes, the use of frequencies instead of tolerances, probabilities instead of possibilities and presence of outcome bias. It is also true that experimentation clearly helps to get insights, identify new problems and, when carefully planned, helps to understand the core difficulties of the company.

1.1.1 Data Is Almost Everything Now, Enabling Change

There was something missing in that old symbiosis; discussing such problems is not really the purpose of this chapter. Instead, here we choose to make the reader appreciate another wave of change that of introducing computer science techniques in marketing, business and customer analytics (without leaving statistical methods behind). We are sure that traditional hypothesis-driven research in business and marketing will continue, stronger than ever. Here we will concentrate on the new changes coming from a *data-driven* revolution. It is fuelled by the increased availability of data gathered, and stored, by new technologies. Today, we are moving into the era of *Data Science*, and again, it all started not with products and services but with the humble consumers and by putting them in the centre of the scene.

It is always interesting to look at the past, the signs of change are already there. We quote:

> The view that an industry is a customer-satisfying process, not a goods-producing process, is vital for all business people to understand. An industry begins with the customer and his or her needs, not with a patent, a raw material, or a selling skill.

We are sure that the reader will think about companies like Amazon, Google, Apple, or Virgin almost immediately. However, we are neither quoting Bezos, Page, Brin, Jobs, Zuckerberg, nor Branson. We are citing here an academic scholar, Theodore Levitt, from an article titled *"Marketing Myopia"* written almost 60 years ago [137]. That article had a clear message, it is a manifesto for change. In it Levitt also said:

> ...the entire corporation must be viewed as a customer-creating and customer-satisfying organism. Management must think of itself not as producing products but as providing customer-creating value satisfactions. It must push this idea (and everything it means and requires) into every nook and cranny of the organization. It has to do it this continuously and with the kind of flair that excited and stimulates the people in it.

Users and customers, like decades before, are continuously generating huge amounts of data. What is now really unprecedented that we have the capacity to gather this data and transform it into knowledge. Even just considering the statistical-only approach, the time cycles between hypotheses formation and their testing and validation have been dramatically shortened. Ron Kohavi, at the Knowledge Discovery from Databases 2015 conference in Sydney, explained how hundreds of online controlled statistical experiments are conducted on a daily basis to understand user behaviour. He explained how *"data trumps intuition"* (see a video from a similar talk he gave that is available in YouTube[1]). After all, another early quote, this one attributed to W. Edwards Deming, also warned us:

In God we trust, the rest bring data.

It is then easy to recognize that the ethos of some of the companies cited above embody a new thinking pioneered by Levitt. From solving "tiny problems", some companies scaled-up their skills and managed to disrupt existing business models occurring at a global scale (and they have created new business niches in the process). They quickly moved from "tiny problems" to "big data", and then to world scale, but actually they are profoundly based in a customer-centric and adaptive new view of the role of the company business intelligence.

1.1.2 While Data Is Everything, You Have Nothing Without Understanding It

The large volumes of data collected by companies around the world are increasingly being exploited with methods that employ computer science techniques. However, this is not a new field; "Why is computer science so useful *now*?" Well, there are the obvious aspects of dealing with computers that have changed. We are increasingly having them allied to the necessary skills to conduct large-scale online statistical testing (as the ones Kohavi discusses). But computer science has established itself as a new discipline for the past 70 years and, at its core, it has a very clear manifesto that of being the quintessential approach to identify, characterize and solve problems involving the processing of information. It creates knowledge from large databases containing information.

This emphasis in problem identification and problem-solving of computer science is interesting for data-driven companies. They directly benefit on the behaviour of people trained in the discipline. We can quote Levitt's article [137] (the article is so rich that it seems an inevitable temptation to continue quoting it);

If thinking is an intellectual response to a problem, then the absence of a problem leads to the absence of thinking.

This said, what computer science brings to companies, business and consumer analytics is a new view, a novel perspective. Computer scientists have a proactive role in an organization, shared by the roots of the discipline in Applied Mathematics.

[1]https://www.youtube.com/watch?v=qtboCGd_hTA.

Questions like: *"Can this be done at this price?" "Can I serve all my customers if each of their requirements need to be met?" "What is the optimal schedule of tasks?"* and many others indicate the role of computer scientists at enabling optimal decisions and how they can base their actions on the result of algorithms operating on the data available.

From a pure psychological perspective, computer scientists bring "new blood" to the mix of skills and human capital of the company. They tend to be constantly on the hunt for "the next problem", the "next variation" that can make a problem seemingly intractable, tractable. Computer scientists will thrive in finding those new problems that would create the "need" for new thinking. They will help to "bridge the gap", if properly motivated, to create the intellectual response required and to open new business niches for companies. We discuss the computer science's "world view" and some of its primary goals later in Sect. 1.3.

In addition to their different complementary perspective, computer scientists also constantly look for efficiencies, reductions of times and costs, increments in prediction. More recently, adopting a customer-centric view, they were central in the development of the wave of *"personalization"*. All these areas, also linked to discrete applied mathematics, operations research and management science, are becoming hugely useful for companies that thrive on the benefits of a data-driven agenda to drive marketing and selling (note, en passant, that as clarified by Levitt [137], these are two different things). He said:

> The difference between marketing and selling is more than semantic. Selling focuses on the needs of the seller, marketing on the needs of the buyer. Selling is preoccupied with the seller's need to convert his product into cash; marketing with the idea of satisfying the needs of the customer by means of the product and the whole cluster of things associated with creating, delivering, and finally consuming it.

The convergence of new methods coming from computer science and statistics is delivering incredible breakthroughs in the areas of automated learning by machines and computers. The field of Artificial Intelligence will dramatically change business and marketing is no exception. However, just "predicting" events, even at the level of predicting what an individual consumer will do/like, is not good enough without an "understanding" of the causes of that choice. Products and services can then be the consequence of an informed marketing process, making it deliver in its true role [137]. This said, the increased availability of data from consumers can be translated, via computer analysis, into both understandable and actionable insights on their needs.

1.1.3 Data Science: Do We Need Another Name in "Buzzland"?

An article in the Harvard Business Review popularized the idea that Data Science is the *"Sexiest Job of the 21st Century"*. But looking smart, Mr. Bond, would not be enough. . . you will need to study hard if you want to become a true data scientist and get one of these “sexy jobs”. For the old players in this game it is not news. But what really is this “new field”?

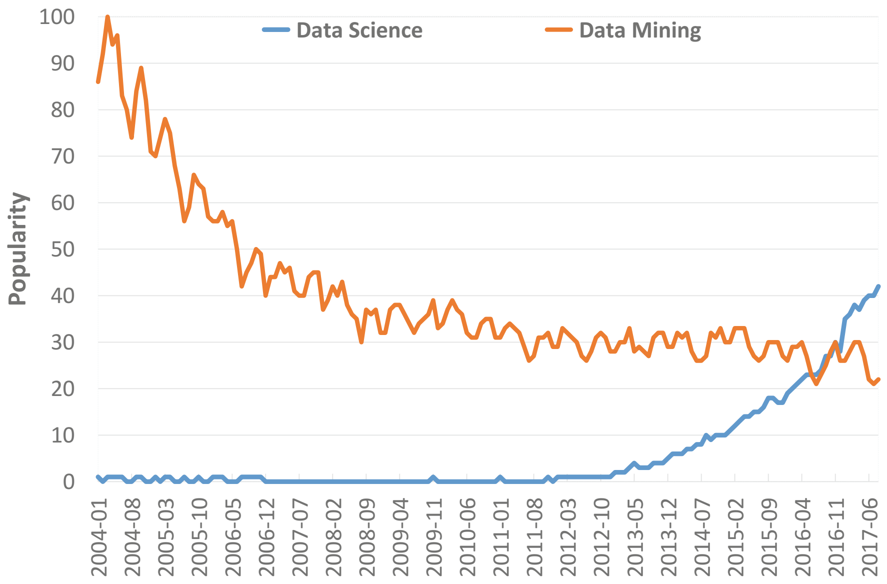


Fig. 1.1 Interest of *“Data Mining”* over *“Data Science”* from the year 2004 till 2017 (measured by Google Trends). The chart indicates that Data Science is starting to surpass in popularity Data Mining from the beginning of 2016

“Data Science” is increasingly being recognized as a hub, an interdisciplinary field dedicated to transforming data into useful knowledge. To be trained in the area you will need to master fundamental knowledge in algorithms for data mining, predictive analytics, Machine Learning and Computational Intelligence. Aside of this, Data Science blends techniques from Computer Science, Statistics, Applied Mathematics, Operations Research, Management Science, Artificial Intelligence, together with Psychology and Economics. The interdisciplinary nature of Data Science is challenging academic institutions around the world. Countries like the UK have opted to create a network of affiliated institutions under the umbrella of the Alan Turing Institute,[2] as an integrated national response to the need of research and training.

The numbers in the *y*-axis of Fig. 1.1 represent the search interest of the term relatively measured against the highest point for the mentioned period of time. A value of 100 indicates the peak popularity for the term. A value of 50 means that the term is half as popular than the most popular term in the period.

[2]https://www.turing.ac.uk/.

The question is not if we need a new name, it is already here and it is based on a very coherent message. There is a quest for a data-driven approach for changing science as well as understanding consumers; thus, we expect that its relevance will continue to grow in the next decades as "algorithmic" and "computational thinking" extend their influence (Fig. 1.2).

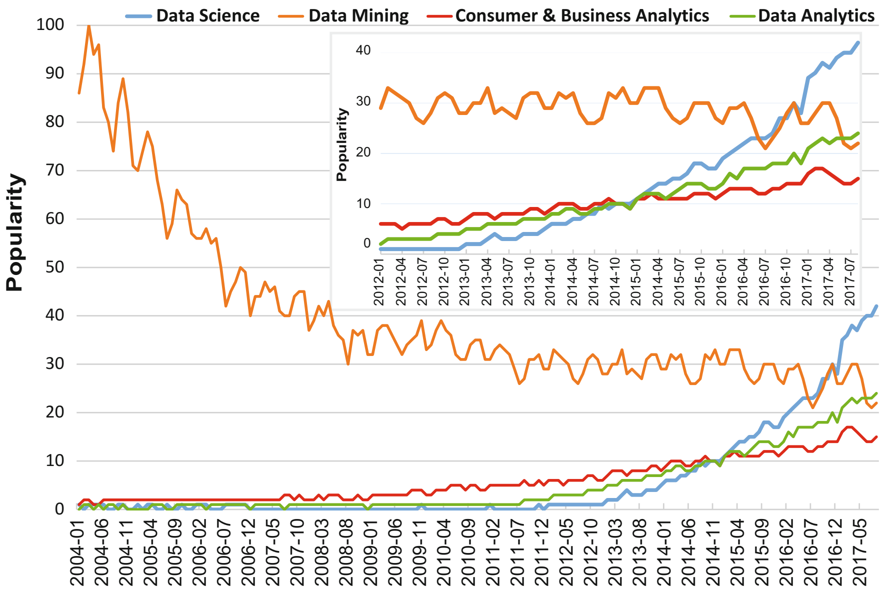


Fig. 1.2 The popularity of the search terms from the year 2004 till July 2017 (again, measured via Google Trends). For clarity the inset shows the popularity of those terms during the period of 2012–2017

1.2 The Algorithmic Revolution

A central theme of data science is the identification of algorithms for the solution of problems that can be solved with computers, one of the core objectives of computer science. We can ask: *"Why are they so central?"*

1.2.1 If Algorithms Already Rule the World: Why Another "Revolution"?

It is true, algorithms already rule the world; everything we do with computers is based on them. What are they? One "computational" or "mechanical" definition is that they are *step-by-step detailed instructions for a machine to execute*. A more philosophical view, quoting Steven Skiena, is that algorithms are *"the ideas behind computer programs"*. More than ever before, algorithms are at the core of all our activities. Some of the world's most successful companies of the digital and networked global economy are now entirely based on them.

Why is another "revolution" coming? There is a clear path by which algorithms will be redefining the Knowledge Economy and will be creating a new one. This is certainly not an understatement. For instance, the progress observed in some areas of machine learning and Artificial Intelligence (AI) has been phenomenal since we first discussed the possibility of editing this book in 2012. Computer-based algorithms are now beating our world's masters in both Chess and Go, by "training themselves" to do so.

The revolution is coming because we are living in a period of accelerated change. Current tablets and smartphones have performances that are comparable to those of the world's best supercomputers three decades ago. This means that we have now what used to be "supercomputer power" at our fingertips, in our pockets or even on your wrist while you rest on a sofa or go for a run through the park. A number of companies are turning things around by thinking of clever strategies to give users more power, by personalizing the customer experience in ways not previously thought possible.

In terms of AI advances, the recent result product the *DeepMind* company is fascinating. This company is responsible of *AlphaGo* the first computer program that managed to defeat a human world champion in the game of Go. The approach required human supervision, so the algorithm would adapt its decision based on human expert moves. But when *AlphaGo* became the world master in the game, what is next? The company put then *AlphaGo* in the "driver's seat", or, in a twist, perhaps a better wording would be: *AlphaGo* became its own teacher. The new algorithm, *AlphaGo Zero*, starts *tabula rasa* and then learns from, and develops the capacity to predict the original *AlphaGo* algorithms decisions, resulting in a new algorithm with superhuman performance that defeated the original *AlphaGo* 100 times without losing a single match [207].

> Humankind has accumulated Go knowledge from millions of games played over thousands of years, collectively distilled into patterns, proverbs and books. In a space of a few days, starting *tabula rasa* (i.e., from scratch), AlphaGo Zero was able to rediscover much of this Go knowledge, as well as novel strategies that provide insights into the oldest of games.

In an interesting twist, AlphaZero was "repurposed" from Go to Chess. *In only four hours* the system came from just the basic knowledge of the rules of Chess to be at a level in which it has beaten *Stockfish 8* the current world champion chess program in a 100-games match up [205]. It also learned Shogi (Japanese Chess) in just 2 h and defeated another world-class program called *Elmo* [205].

This is an extraordinary acceleration, provided by the closed loop of algorithms teaching algorithms, reaching superhuman abilities in a matter of hours. This feat has no previous comparison in the history of our world and it is really revolutionary.

Algorithms now decide what to show to us; which books we may like, what to watch, where to stay, how to travel, where to study, what to buy and in which companies to invest. Although hype also exists, even in the medical area, algorithms are now allowing us help to diagnose and even predict the occurrence of certain diseases. Algorithms are creating new knowledge by teaching themselves from the data they generate.

1.2.2 Bad Algorithms Might Go, But Good Ideas Remain

The progress with *AlphaGo* and *AlphaGo Zero* is in part due to new existing hardware (with some versions of *AlphaGo* running over 176 GPUs, while other versions and *AlphaGo Zero* run on 4 TPUs, the Tensor Processing Units created by Google specially for machine learning, which are now being reported to run at 45 Teraflops, that is, they are capable of performing 45 trillion floating-point operations per second).

It is also just to say that some algorithmic framework ideas are, in some sense, more transcendental than hardware and the latest technological achievements. Good algorithms, and good algorithmic ideas, remain for a long time. At the core of *AlphaGo* and *AlphaGo Zero* there are tree search techniques [206, 207] that have been proposed many decades ago. In some cases an algorithm may "remain forever" once you have mathematically proved that there is no better algorithm for a particular problem. A formal mathematical proof could bring one question to a close: "how complex" is the computational problem? That is why the field of computer science always strives to find the best possible algorithm for a given problem. In some cases researchers can find them and a problem is, in some sense to be defined later, a "case closed".

New algorithms are also inspired by new hardware, and the core ideas will remain, evolving with the new generations of hardware. For instance, the technical presentations at the Parallel Computing and Transputer Applications conference in Barcelona (PACTA '92, 20–24 September 1992) had a general focus on the "Teraflop Grand Challenge". The general conclusion arrived at that conference was that: *"a Teraflop computer would cost hundreds of millions of pounds; need to be housed in a small warehouse; and require an extensive cooling system. To operate the machine would require upwards of three Megawatts, requiring its own electricity sub-station to down-load off the national grid. It was noted that switching such a machine on or off would require the approval of the local Electricity authority on each occasion"*.[3] Twenty years later NVIDIA unveiled a 1.3 Teraflop GPU for Supercomputing based on the Kepler architecture. We have now this dreamed, once utopical supercomputing power, and it is a reality. A system built on many of these "building blocks" achieved 27,000 Teraflop peak performance (Titan at Oak Ridge National Laboratories).[4] The new building blocks are proving even faster, NVIDIA's

[3]http://www.chilton-computing.org.uk/inf/transputers/p011.htm.

[4]http://energy.gov/articles/new-titan-supercomputer-named-fastest-world.

Volta GPU now has a peak performance at around 7 Teraflops. The game console Xbox One X delivers 6 Teraflops. In terms of supercomputing, Sunway TaihuLight, in Wuxi, China, maintains its leadership with a mark of 93.01 Petaflops (data from November 2017); thus, it tops the list of the 500 fastest supercomputers in the world.

Supercomputing brings an impressive new world to business analytics which is yet to be fully explored. Together with new algorithms it will give large corporations (as well as data-driven and computationally wise start-ups) an unprecedented capacity to analyse large datasets. Supercomputing will bring a transformational capacity to small and medium enterprises which is still waiting to be properly used. When allied to Big Data this will shake marketing [37, 64, 211].

While hardware progress and prowess is impressive, however, it is "the ideas behind computer programs" that live on. For instance, at the same supercomputing conference we just mentioned (PACTA '92), a relatively new methodology named *"Memetic algorithms"* was presented for the first time in the European community [162]. It was one of the 183 papers accepted; now it is not only the most cited paper of that conference, it has become a driving idea that has become stronger than ever before and we dedicate an entire section to it and the current applications of memetic computing in business analytics and data science. Originally, they found an initial motivation in the use of these large scale computing systems based on parallel architectures (see [160] for a historical account of the development of this field until 2012). With the advent of "supercomputing at your fingertips", smart phones and other systems allow users to tailor solutions to their own needs. There is a pending revolution, that of personalized systems, that would benefit from the user experience and that wishes to bring the value of the wealth of data available online with a unique perspective.

What is important about the story of memetic algorithms and their development is that it is a clear example which shows that some methods remain in practice many years later. While the "algorithmic revolution" is here now, its origins go back three decades or more, when computing became personal. It may be the case, then, that we should say that now we are living the times that what it was considered "supercomputing" has become personal and is in our pockets.

1.2.3 Is the Disruption Real?

In any revolution there are new opportunities, challenges, as well as new problems ahead. Nothing comes without a cost, especially in revolutionary times. The global changes of our intellectual and economic endeavours are imminent. The clear trouble of manufacturing industries, for many countries, may have been an early warning of more disruption to come. A perfect storm is brewing, powered by accelerated progress in Mathematics and Computing; its key force is the global deployment of Artificial Intelligence (AI) techniques. In the mid-1960s, one of the founders of Artificial Intelligence, the Nobel Laureate in Economics H.A. Simon, predicted that "machines will be capable, within twenty years, of doing any work a man can do". Our current machines are not able to do "any work a man can do";

instead, they are much better than the best humans in specific and certain tasks, e.g., in games like Go as discussed in the prior section, a trend that started with Checkers (Chinook vs. M. Tinsley, 1994) or Chess (Deep Blue vs. G. Kasparov, 1997), as well as Jeopardy (Watson vs. B. Rutter and K. Jennings, 2011).

The "digital revolution" should now be renamed "the algorithmic revolution". In reality, everything that we do with computers corresponds to a set of step-by-step detailed instructions designed to accomplish some final outcome in a machine. Our anthropocentric perspective "Everything we hope to do with computers requires the design and implementation of algorithms" may still be valid, but the design of algorithms is no longer the privilege of our race, machines now can also design algorithms. And algorithms have become a powerful new force in the Knowledge Economy. An algorithm presented by Brin and Page in a conference in Brisbane, Australia, back in 1998, named PageRank [30], is credited to be the foundation for Google. A similar algorithm was used in the 1996 search engine RankDex designed by the CEO of Chinese search giant Baidu. Algorithms also give us great opportunities. Algorithms are the true engine of the new Networked Knowledge Economy.

The Digital Economy is now considered a key driver of growth for many developed countries, it will also shape the future of some economies that will use the news technologies in creative ways. Europe has already recognized this fact and is working to get all possible global advantages of it. Before Brexit, Europe was planning to merge 28 national markets to a single digital one. Europeans estimate a global contribution to their continent of 415 billion Euros per year and the creation of 3.8 million jobs.[5] This new Digital Economy will create unprecedented opportunities for the delivery of goods and services, boost existing skills, allow life-long learning and facilitate investment for the creation of ICT start-ups and new companies. The sheer size of this new affluent cohort of 500 million people can only be matched by the growth of the Asia-Pacific economies and brings opportunities and challenges.[6]

1.2.4 If "Data is Dumb"... Is "Big Data" Dumber?

"Algorithms are where the real value lies. Data is inherently dumb. Algorithms define the way the world works". These were the words of Peter Sondergaard, Senior Vice President, Gartner Research, in his opening talk at the Symposium ITxpo (Oct. 5, 2015). This is no overstatement. No actionable insights can be found by looking at data without any methodological tools such as algorithms. Industry is clear about the benefits of an algorithmic-based approach, supported by the best Computer Science practices, to develop the economy.

[5]https://ec.europa.eu/commission/priorities/digital-single-market_en.

[6]https://www.gov.uk/government/publications/the-age-of-algorithms.

There is no doubt that with the increasing availability of large datasets, there is a huge potential delivered by scientific activities to transform the generated data into knowledge. Peter Sondergaard estimates that an emergent "Algorithmic Economy" will develop from the Internet of Things (IoT) [208] . He predicts that the impacts will be ubiquitous and massive;

> Products will be defined by the sophistication of their algorithms. Organizations will be valued based not just on their big data, but the algorithms that turn that data into actions and ultimately customer impact.

The prediction is that *"by 2020, 30 billion mobile phones, tablets, computers, wearable technology devices and other types of connected devices will be in use"* and that *"the incremental revenue generated by the IoT suppliers is estimated to reach 309 billion per year by 2020"*.[7]

It is interesting to point out that the part of the prediction that says that organizations "will be valued based not just on their big data, but the algorithms that turn that data into actions and ultimately customer impact" may have already been verified. A relatively new company called Jet.com (established in Jan. 2014) was valued and purchased at USD 3.3 billion in 2016. Part of its success is based on an algorithm that allows consumers to find the best deal based on the actual contents of their digital shopping carts. With an estimated return-on-investment of approximately 15 times for an e-Commerce company and the news that an entire shopping mall previously valued at USD 200 million was purchased in Jan. 2017 for just USD 100 [18], it is clear that Internet retailing based on algorithmic solutions is having a clear impact.

1.2.4.1 Big Data vs. Large Datasets

Many people and many media reporters, perhaps for lack of understanding, neglect the importance that algorithms have. They also tend to confuse "Big Data" with "larger datasets", so it would be relevant to give some clear definitions. The current Wikipedia entry for "Big Data" is rather unsatisfactory: *"'Big data' is a broad term for data sets so large or complex that traditional data processing applications are inadequate"*.[8] Defining something by our current "inadequacy" to do something with it is not a good start. However, it does point at the need to have powerful algorithms that can extract meaning from it. Any relevant online blogpost or academic referred paper on "Big Data" will explain some common definition that includes the "four V's of Big Data"; *Volume, Variety, Velocity* and *Veracity*. This is where the difference lies between simply a "larger dataset" and true "Big Data" [39].

[7] http://www.businessnewsdaily.com/5450-internet-of-things-business-opportunities.html.

[8] https://en.wikipedia.org/wiki/Big_data.

The Volume of data of course does refer to sheer size but it is also the number of datasets that are increasingly collected, stored and available to business leaders and researchers which is challenging us. It is predicted that by 2020, *40 zettabytes* (43 trillion gigabytes) of data will be created.[9] This is a 300% increase since 2005. Although greater amounts of data make it harder to analyse all the information available, it is the other three V's that truly make Big Data so hard to deal with.

Today's data landscape has an extremely high variety and velocity. This means that there is not just one type of big dataset that makes up the whole equation to understanding a problem, trend or topic. Businesses need to derive data from many different sources which come in many different formats, combine all this data, analyse it and generate useful insights. For instance, data coming from social media sites in the form of text or numeric values such as "likes" needs to be combined with financial data trends, sales figures and possibly historical datasets for a manager to make a completely informed decision about their next strategic move. Furthermore, high Velocity means that all these high volumes and varieties of data are increasing at an ever accelerating pace. You may have heard that nowadays, one flight of a Boeing 787 creates half a Terabyte of data. Similarly, during one trading session of the New York Stock Exchange, around 1 TB of data may be generated. In a lot of cases, decisions need to be made almost instantaneously, sometimes not even by a person, but by algorithms and machines. This means that a high velocity of data-streams needs to be accounted for in the computing power and the algorithms scalability aspects. Finally, the last "V" stands for "Veracity". Basically, it refers to the uncertainty of data. This topic has received a lot of attention in recent years as businesses and consumers have become more aware and concerned about integrity, privacy and data security in our digitally interconnected world. Not only does poor data quality or poor data standards cost economies and organizations a lot of money, it also risks the integrity of findings coming from these data sources leading to potentially catastrophic business decisions.

This is why we need to tread especially carefully when dealing with Big Data as we do not want to have "bigger" and "dumber" data without any extra understanding or advancement in knowledge. What we want is to make ever more informed decisions, use the information available to us in ever-increasingly efficient ways and enhance the life of consumers, organizations, businesses and economies through the use of data science techniques.

1.3 Computer Science: An Unusual but Rightful Introduction

Firstly, we provide a brief background to the computer science discipline in general, what it encompasses and some of the reasons for the unavoidable omnipresence of data analytics in any transaction, purchase, or planning today. A business or

[9]http://www.ibmbigdatahub.com/infographic/four-vs-big-data.

marketing person will listen to the words *algorithms, heuristics, optimization, machine learning, machine teaching, grammars, graphs, hypergraphs, etc.*, and soon become overwhelmed about the whole different vocabulary. We will try to give a shortcut to help them be introduced into this new language. Before we start explaining some of the fundamentals, there is one "step zero" which is to discuss what "a problem" means. This also gives us the opportunity to set up the first comment between the Business and Marketing and the Computer Science communities, as they perceive this word in a dramatically different way.

Understanding the Meaning of the Word "Problem"
Perhaps the first advice to bridge the gap between the language of a computer scientist (let's call her Anna) and that of a business or marketing person (let's call him Ben) is regarding the use of a word: "problem". The computer scientist would be very happy and glad to hear: "we have a new problem for you?" while the exact opposite may be the case for the marketing or business professional. There is a clear reason for that. For the computer scientist, the world "problem" brings excitement. People trained in the development of algorithms understand "problem" as a mathematically well-defined challenge. Usually, there is a set of "problem instances" that define the possible inputs of the problem and there is a "task" to be addressed. Depending on the problem, the number of possible inputs can be finite or infinite. Generally the objective is to find an algorithm that provides the right "output" for each possible instance, thus "solving" the particular task (if the algorithm provides the right output for all possible inputs).

In this sense, a "problem" becomes a kind of a "contract" (i.e., there is a well-specified type of *inputs* and there are well identifiable deliverables, the *outputs* or *solutions*. This is part of the reason why a computer scientist likes the word "problem". It is expected that they can easily understand what constitutes the set of given inputs, which is the type of outputs that might be expected as solutions, and, consequently, they can design an algorithm that connects both ends.

Instead, for a person with a business or marketing background, as perhaps for the rest of the population as well, "a problem" is often associated with an undesired difficulty in accomplishing something. Thus it is often the case that, when somebody from a large company identifies some "problem to be addressed" in some aspect of their operations, for instance, in logistics, it is generally not a clear-cut precise definition of a problem as in computer science. Some computer scientists may say that it is an "ill-defined" problem, and not "a real problem". Most of the engagement between the client and the provider of computational services revolves in understanding the true nature of these "ill-posed questions" and transforming them into a set of "well-posed" and "clear-cut" objective computational problems that can be

(continued)

addressed by algorithmic means, something that today means that solutions can be found via algorithms and these, in turn, can be implemented in software.

The advice should also include something more: be patient and enjoy the interaction. Nothing better than a good culture clash to make some magic happen and bridge the gap.

1.3.1 The Complexity of Problems

Our computer scientist, Anna, would certainly agree that we are following an unorthodox path to explain algorithms and what they can do, as well as *what they cannot do*. For Ben, and perhaps for the rest of the population, it is difficult to understand *"Why can't we just find a simple algorithmic solution that works for all problems?"* Well, to put it in simple terms, we currently *do not* have algorithmic solutions for all possible problems (and we do not expect that computers can solve all problems, see the educational video about this point[10]).

It is true, however, that for many problems we can have, in principle, step-by-step unambiguous procedures that can be coded and may give relatively good solutions and also be very fast, which is always a plus. However, they may lack performance guarantees (i.e., they may give you a feasible solution, although it may not necessarily be an optimal one).

Why do we have this global quest of matching problems with the best algorithm we can design for them? Note that the major goal of computer science is to classify problems. We would be interested in knowing, for instance, "the best instance" of a problem that Ben brings to the discussion table. Anna could even work out which that case would be. Perhaps the solutions she can provide, in the marketing context of Ben's business, can bring actionable insights. If for Ben, identifying the general problem under study (which may be too computationally hard to address) has "a special case problem" (which can be solved quickly), may lead to an ultra fast algorithm. Ben could then adapt his business strategies consequently so that he can exploit this fact for profit. However, it is generally the case that computer scientists are interested in finding the *worst-case scenario*. Having this knowledge has clear benefits, for instance, allowing to have preventative controls avoiding these situations, but it is also important to help classify problems.

1.3.1.1 Why Worst-Case Analysis Does Matter?

For a given algorithm, there might be an instance (or a class of instances, potentially an infinite number) for which there is an algorithmic solution and for which the

[10]Proof That Computers Can't Do Everything, (The Halting Problem) available in YouTube at: https://www.youtube.com/watch?v=92WHN-pAFCs.

algorithm has its worst performance. For instance, imagine that an airline company has a very well organized/optimized schedule of n flights. If, for instance, a couple of the planes had some technical issues that oblige them to a delay for repairs, it may need to face several decisions. The situation is clearly something that could eventually happen (albeit with a small probability, having $k \ll n$ planes with technical difficulties is certainly not unexpected), so algorithms need to be designed for the problem (i.e., support decisions of rescheduling flights, minimizing delays, reducing cost of operations, etc.) with the final overarching objective of returning to the cyclic scheduling of operations in the minimum number of days possible as well. These algorithms should not be designed "for some of the most common possible failures"; they have to address the problem regardless which is the failure. For instance, if an algorithm is designed for the problem of facing that "at least k planes needed to be repaired and cannot flight", then the algorithm may have a worst-case scenario that depends on the selection of a number $k' \leq k$ planes.

Note that for another algorithm the worst-case scenario may occur for a different selection of these k' planes. This creates an interesting dynamics between a computer scientist (generally involved in algorithm design) and the decision makers, as there is also a strong connection between the optimal cyclic scheduling that the company decides to normally operate with. Some cyclic scheduling/timetabling may cost to the company slightly more but be more robust to some types of failures (which could have catastrophic consequences in terms of brand reputation).

The performance of algorithms is usually measured by the number of computational steps required to give the final output/solution. And then, there is a natural question for Anna...if Ben brings a new problem, one that nobody has ever seen before, can she obtain a mathematical proof that guarantees that for the given problem, and in the worst case, there is an algorithm *that always terminates* and *retrieves the optimal solution* while it also satisfies certain efficiency requirements? That might explain why Anna and other computer scientists are so motivated to hunt for new problems, they bring new life to their professions. They aim to identify the "computational complexity" of the new problem, so this brings us a rule of engagement for Ben and Anna. The creation of a new customer service system may lead to the requirement of solving certain computational problems. Ben expects that Anna will find the worst-case scenario, a possible query that may make the service useless. If the worst-case can be solved in "reasonable time", then the system may be built; otherwise, a complete redesign of the service is probably needed.

Don't Use the Word "Complexity" in Vain

Another word that clearly may have different meanings to people coming from disparate disciplinary fields is one we have just used: *"complexity"*. In Marketing and Economics, people have been using this word for almost everything [147], most of the time as a proxy or a synonymous for "difficult", or "hard to understand". A unique definition of "complexity" would then be

(continued)

entirely futile and even individual ones which may be particular to one aspect of a system/situation are hard to come up with [200]. Cosma Shalizi's online notebook on the different measures of complexity is an excellent reference for those who want to mathematically delve into the many uses of this word in a wide range of fields.[11]

At first, the reader may think that we refer to this since there is an intrinsic subjectivity at the core; what may be "complex" for somebody might not be for some other person. Arthur C. Clarke once said: *"Any sufficiently advanced technology is indistinguishable from magic"*. Analogously, any given new problem always looks "complex" at the beginning when you are coming from a different field or when you face it for the first time. In fact, the lack of an objective and universally accepted definition for "complexity" does not mean that computer scientists use this word loosely. A central aspect of the Theory of Computation is to study the *computational complexity* of a problem. For further reading, after finishing this chapter, we also recommend an excellent tutorial by Craig Tovey [217].

1.3.2 Selecting Small Feature Sets: A Central Problem in Data Analytics

We have said that computer scientists love problems and we are not an exception here. Some are closer to our heart than others. The problem we are going to introduce is "a little gem". It will help us to present several concepts. We will discuss it within a scenario that can arise in business analytics practice.

Let us suppose that we have the following dataset (Table 1.1). Several products have been purchased by the same buyer(s) and they have different characteristics. We will call each of these characteristics *"attributes"* or *"features"*. For the sake of simplification, we will assume that these features are Boolean (i.e., either the object has each the attribute or not). We will denote the feature state as either being "1" or "0" (i.e., either we can observe/measure that characteristic in an object (1) or not (0)). One such a feature could be (in the case of clothing, for instance) is a product for women (value=1) or not (i.e., it is for men and value = 0). This indicates a single piece of information (a feature) that tells us something about the product; it allows to distinguish between products a pair of products if the values are different. The products are divided in two groups; the curious thing is that one of them is rejected by the consumer(s), while the others are not (all the others are purchased). What is wrong with that product? Could a subset of the set of features give us the necessary clues to understand customer behaviour?

[11] http://bactra.org/notebooks/complexity-measures.html.

In Table 1.1 we present data for seven products, each one has five characteristics (i.e., their features, one column per feature), and the only one that was not purchased is represented by the state of the features of the last row.

Table 1.1 An instance of the ONE-OUT k-FEATURE SET

0	1	1	1	0	Sold
1	1	0	0	0	Sold
0	0	0	0	0	Sold
1	0	0	1	1	Sold
0	0	1	1	1	Sold
0	0	1	0	1	Sold
0	1	0	1	1	Unsold

In this case we are modelling a real-world problem in which seven similar products (corresponding to rows of the matrix) have been purchased by a user but one has been rejected. Five different attributes/features (columns) can be observed in each of the objects, and they are present (coded as "1" if the object "is positive" for a question that test for the attribute and it is "0" if it is "negative" for this attributed). One pair of features that distinguishes the unsold object is the only one that has the joint pattern of features (1, 1) which are positive for features 2 and 5

It may be possible that no single feature could individually "explain" why a product was rejected. If that would have been the case, one feature would have been present in a particular state in the non-purchased object and and that state, for that particular feature, would have not been observed in the six others (the reader can check that there is no such a feature in this example). However, if we look at the observed features in the second and fourth columns, both are "1", meaning that the non-purchased product has these two characteristics. That is excellent in general, but unfortunately there is one exception. While almost all the sold objects do not have this combination (both being "1"), the first row indicates that there is one purchased object that has the pattern. Formally speaking, we were wishing that the set $\{2, 4\}$ was a *2-feature set*, a subset of the features that can help us to "explain" why one of the objects is rejected, while the others do not. But we have found an exception, a natural question arises…does there exist a 2-feature set for this database? As a matter of fact, the answer is "Yes" for this decision problem. There is one, the reader can also check that $\{2, 5\}$ is indeed a 2-feature set.

If we are going to inform marketing, or the product design department about which combinations of attributes may lead to a product being rejected, several other natural questions arise: (a) does there exist *another* 2-feature set apart from $\{2, 5\}$? (b) we know that $\{2, 5\}$ is optimal in the sense that there is no other feature set having smaller cardinality, are there other 3-feature sets that do not include the set $\{2, 5\}$ as a subset? We can answer this second question first, the answer is "Yes" again; the set $\{1, 3, 5\}$ is indeed a 3-feature set (the non-purchased object has the pattern (0, 0, 1), respectively, on these three features). This pattern is not present in any other purchased object. The product design department may be interested in this observation as they may have an alternative explanation that would help them to understand why this product was rejected.

The reader would note that enumerating a *collection* of feature sets may lead to important insights for the marketing department. They may give alternative views on the rejection of that product. This approach is indeed very different from what they can have with the use of statistics alone. We are searching for all possible sets of features that satisfy a certain requirement.

We now go back to the first question. The issue can be iteratively solved in two steps. First, we remove the second column of our database and we ask, again, if there is a 2-feature set. If the answer is "No", then we incorporate that second column again and we remove the fifth column and again we ask if there is a 2-feature set. If the answer is again "No", we conclude that the {2, 5} is the unique optimal feature set for this database.

We can see that this approach to knowledge extraction from databases is *combinatorial* in nature and that statistics is not the branch of mathematics that deals with these problems. The reader may notice that if, realistically we have information about a set of 1000 products with 80 features each, we have a relatively large number of possible combinations to check. Combinatorial *optimization* methods come to the rescue [70]; they are then used to find smaller feature sets. For instance, an exact method would give you the one of smallest cardinality, and that could potentially be very useful.

1.3.3 The Computational Complexity of Problems and the Efficiency of Algorithms

Problems like the one we have just introduced immediately bring into consideration one of the central missions of computer science: to characterize the computational complexity of *all* known well-defined problems. For the particular case of having to identify the minimal feature set, in the case of 1000 products with 80 features each, a "brute force" approach would need to enumerate all possible solutions. This is indeed a staggering number (2^{80}) and it is a few orders of magnitude different than the conjectured number of stars in the observable universe. A "brute force" enumeration is clearly out of the question. Obviously, we need to find a better way. As we said before, one area of discrete applied mathematics (and computer science) called "combinatorial optimization" deals with these problems. Again, problems in this domain could have radically different computational complexities.

1.3.3.1 Size Does Not Matter

Let's compare with another combinatorial problem most people are familiar with called *"Sorting"*. Assume you are given 80 different objects and that somebody has given them some merit value. Just to make things "harder", let's even imagine that all of them have a different merit value, an integer from that ranges from 1 to 80; you are required to put them in order. Obviously, there is only one way of ordering 80 different numbers from the largest to the smallest value. This means that, if you are given that task, you will need to give "the right order" among the 80! (i.e., eighty factorial) possible orderings that 80 numbers can have. This is the number of feasible solutions of the problem for such an instance and its approximately 7.157×10^{118}, which is massively larger than the number of *elementary particles* in the visible universe.

We note that the mere size of the space in which we have to search in order to find feasible solutions is not very relevant. Actually, computer science considers *"Sorting"* an *easy* problem because they know that an *efficient* algorithm exists for it. Sheer size of the search space, alone, is not enough to characterize the complexity of problems. Something else is needed.

1.3.3.2 What Is an Efficient Algorithm?

We now explain the concept of efficiency in algorithms. We have said before that an algorithm is a "step-by-step" procedure for solving a problem. For the problem called "Sorting", there are several algorithms that guarantee that they will bring you the optimally sorted sequence. For one particular input sequence of n objects, the number of steps required by an algorithm can be bounded in the worst case. For instance, for sorting, there is an algorithm called *"Merge Sort"* invented by John von Neumann in 1945. The number of steps required by this algorithm is bounded by a polynomial function (p) in the number of objects (n) (i.e., in this case it has been proved that $p(n) = k * n\, log(n)$, with k being a constant which is independent of n). This type of algorithms are said to be *efficient* and the sorting problem is said that can be solved *"in polynomial time"* (because there is at least one known algorithm that has a time performance which in the worst-case scenario is bounded by a polynomial function in the size of the input).

The situation contrasts with other problems. For some of them, we already know that there is no polynomial-time algorithm (because a researcher has mathematically proven that). For instance, in the scientific article *"The Odds of Staying on Budget"* [84], the authors refer to a very intuitive problem coming from a tourism area; they consider this basic question:

Is it possible to travel from Copenhagen to Kyoto in less than 15 hours?

which can be answered either "Yes" or "No" in polynomial-time (since this question matches the problem "find a shortest path in a graph" for which we know there is a polynomial-time algorithm proposed by Dijkstra[12]). However, in a real-life scenario, uncertainties in the flight path and airport conditions may introduce unexpected delays. Then problems of the type:

> Given a budget constraint $b > 0$ and a probability threshold τ what is the complexity of determining whether the probability of paths reaching a designated target state with cost consistent with b is at least τ?

give rise to several types of problems, some of them which are in a class called EXP-Complete. This class contains as its elements the problems that can be solved in a number of steps bounded by a function which is $f(n) = k\ 2^{p(n)}$, where $p(n)$ is a polynomial function of n (and n represents the size of your input and k is a constant that is independent of the size of the input). This means that, *in general*, a query of that type, for some types of possible inputs, would require an algorithm that would need a significant extra amount computing time. This problem of answering those queries shares membership in this class with other EXP-Complete problems like computing a perfect strategy in generalized forms of games like Chess [71], Go (with Japanese Ko rules) [185] or even Checkers [186]. Even if a human travel agent has oracle-like powers, it would be difficult to know if what the agent is saying is true or not by computing means. It is then unlikely that we will see some online flight ticket-selling software be empowered by algorithms that perfectly answer queries like that; most customers would not accept such long delays for receiving answers.

Note also that *"in general"* does not mean *"on average"*. In some cases, the query may be resolved very quickly, due to the nature of the input, but an algorithm may take exponential time for another query, leading to a highly heterogeneous customer experience and bad reviews of the system. Computer science and computational complexity provide the right framework for these problems to be properly discussed and their running time behaviour properly estimated.

For other known computational problems, we have to raise our shoulders and say… *"We simply do not know"*, their complexity status is still open. For the feature set problem we have presented before, it actually belongs to an important class of problems for which *we just do not know*. The same happens for many other problems in computer science, if somebody asks Humanity *if there is an efficient algorithm for it*, we should all raise our shoulders and say in unison "We do not know!". However, for the feature set problem we know something more than we discuss next.

1.3.3.3 NP-Completeness

What are those problems for which we really do not know if there exists an efficient algorithm that can be applied in any case and that it is also guaranteed that will bring the optimal answer? Interestingly, many important questions in Customer and

[12]Problems for which the answer can only be either "Yes" or "No" are called *decision* problems.

Business analytics can be modelled as problems such as those two above. It is thus utterly important to know how we can address this issue.

There are problems that are classified as belonging to a computational complexity class called *“NP-complete”*. Generally speaking, these problems are easy to formulate, and in some instances deceptively simple to understand, yet are very hard to solve (sometimes in both exact and approximate ways). In many scenarios in data analytics, they are the right mathematical model for understanding a core question of interest; they are crucial for extracting knowledge from data. They also have a peculiarity, if somebody claims to have found (or even “guessed” or “given” by a mythical oracle) one feasible solution for the problem we can use an efficient algorithm to validate (or disprove) the claim. Thousands of NP-Complete problems have already been identified and they are omnipresent in our life, our economic activities and in every aspect of science. NP-Complete problems were quickly identified in the Business (e.g., [222]) and data analytics domains (e.g., [95]). Nearly 100 papers were published in the first decade since the introduction of the class (1971–1981), having “NP-Complete” as part of the title. In July 2017, a simple Google Scholar search query retrieves approximately 239,000 hits when using the term “NP-Complete”. Consequently, the theory of NP-Completeness is considered to be “the cultural ambassador of Computer Science” since it has influenced almost every aspect of science and technology by identifying computational problems that need to be addressed by sophisticated computational means.

Rule of Engagement: Understand That for Computer Scientists, Optimization “Reduces” to Solve Decision Problems

It is perhaps a good moment to explain the relationship between *optimization* problems and *decision* problems. The latter is one in which a particular problem can be posed as a question about the input values and the answer can only take two possible outputs, i.e., it is either “Yes” or “No”. In an optimization problem the task is to find the *best* solution from a larger set of *feasible* solutions.

From a computer science theoretical perspective, they are actually tightly linked! Suppose that Anna brings some sort of “magical algorithm” $\mathcal{A}$ that solves the decision version of the k-feature set problem (k-FS, for short), Ben can then solve the optimization version (which we will call opt-FS) using the following algorithm. The strategy has two major steps. He will first use $\mathcal{A}$ to determine the minimum k for which there is a k-feature set (let’s call this value k_{opt}). He now knows the optimal value, now he has to find one. The second step is also iterative. He will then repeatedly pick an arbitrary feature, and he will remove it from the instance (now our instance has $n - 1$ features and still has the same number of samples m). Again, he uses $\mathcal{A}$ to determine if the new instance has a k_{opt}-feature set, if it does (i.e., the answer of $\mathcal{A}$ is “Yes”), the removed feature was not essential (so we pick another arbitrarily chosen

(continued)

feature to delete and iterate on this procedure); otherwise, this feature helps to discriminate at least a pair of samples, so we remove all pairs of samples from different classes that this feature discriminates (in our previous example, products that were "sold" and "unsold"). We can safely do this as this feature "explains" this dichotomy. Ben will add the feature to the list of features in the feature set we are seeking and iterate, this time asking if the remaining input instance has a $(k_{opt} - 1)$-feature set. Ben's algorithm iteratively reduces the instance and will stop when we have identified k_{opt} essential features and we have run out of samples.

We leave to the reader, and Ben, the task of analysing how many times $\mathcal{A}$ will need to be used (in the worst-possible scenario for a k-feature set problem involving n features and m samples that belong to two different classes). However, we hope the reader has already got the feeling that Ben's algorithm is relatively simple and would work provided that $\mathcal{A}$ is also very efficient. Unfortunately, no such algorithm $\mathcal{A}$ is known for k-Feature set that is also mathematically guaranteed to run in polynomial-time. If were are lucky, and eventually somebody finds one, then Ben's algorithm will also run in polynomial-time. Some day, some day, it may be found. Some people are still hopeful, are you?[13]

1.3.4 Networks and Graphs

A huge amount of data analysis methods require the use of mathematical entities known as *graphs* and *networks*; many decision problems based on them were soon found to be NP-complete after the first NP-complete problem was identified. Readers familiar with graph and network mathematical notation can skip the grey box below in which we give some basic definitions. This book also dedicates a section with several problems and new algorithmic ideas as well as an introductory chapter that naturally connects with this one. A few definitions are still needed here for completeness of the following discussions.

In discrete mathematics and computer science, a *network* is distinct from a *graph* and they have their own set of very interesting associated problems [11]. A key difference of networks is that, in addition to weights on the edges, they also have *capacities*. As a consequence, networks are used to model traffic, scheduling problems with circulations on demands, delivery of fluids in pipelines or currents in electrical circuits and so forth. Unfortunately, other disciplines have started to use the word "network" to what has been known for years as "undirected graphs" (either

[13]Millennium Prize: P vs. NP http://theconversation.com/millennium-prize-p-vs-np-4246.

weighted or not), so some people are suggesting to use the term "flow network" or "transportation network" now to characterize these other mathematical entities.

Algorithms for solving optimization problems in graphs are one of the cornerstones of computer science undergraduate education. They are used to model many types of relationships occurring in business and data analysis problems. Even classical problems, like the *graph colouring problem*, find applications in areas like microeconomics [167] or timetabling [194].

In graph colouring the task is to find, given an undirected graph $G(V, E)$, an assignment of k different colours to each of the vertices of the graph such that no two pair of vertices connected with an edge have the same colour. It is then obvious that the answer is "Yes" if $k = |V|$ (you just use a different colour for each vertex) and that the answer is "No" if $|E| \geq 1$ and $k = 1$. Then, for any particular graph (having at least one edge) there is a value of k (let's call it k^*) such that the answer is "Yes" for k^* but it is "No" for $k^* - 1$. Obviously, we only need two colours for a tree (or a forest) (graphs that have no cycles). In fact, it is known that we can always answer if a graph can be coloured with two colours in polynomial-time, but for $k = 3$ we need to raise our shoulders and say *"We don't know!"* if such algorithm exists. The problem was proved to be NP-complete in 1972.

A generic trick of computer scientists is then to transform particular quests into well-defined series tasks, and then formulate them as graph optimization problems for which they can use known algorithms. One of the areas they are particularly trained in is in searching for solutions to problems that try to maximize or minimize a particular function of interest. They call *objective functions* these merit functions that rank solutions according to a preference, so it is not surprising that we will follow this approach. We will now show with some examples how some "classical" optimization problems in graphs are at the core of current models of knowledge extraction from large datasets. If you are not accustomed with standard graph notation, the following "Rule of Engagement" can help you bridge the gap.

Rule of Engagement: Learn About Graphs and Networks and Use Standard Mathematical Notation, It Facilitates Dialogue

Firstly, a brief introduction to some basic notions of Graph Theory and its notation is presented here in order to provide context.

A simple undirected graph is denoted as $G(V, E)$ in which V is a non-empty set of *vertices* (sometimes also called *"nodes"*) and E is a set of unordered pairs of distinct elements of V called edges. The *cardinality* of the sets of vertices and edges is denoted as $|V|$ and $|E|$. Standard set notation applies, for instance, $V' \subseteq V$ indicates that V' is a subset of the vertices of V (but it could potentially include them all, i.e., $V' = V$). An *edge weighted graph* is denoted as $G(V, E, W)$ in which V and E are defined as before but each edge now has associated a weight (i.e., W is a set of weights). We refer

(continued)

to the sets E and V as $E(G)$ and $V(G)$, to indicate that they are the set of edges of G and analogously, the set of nodes of G, respectively [82].

A *path in a graph* $G(V, E)$ is a sequence of edges of G that connects a sequence of vertices. In an undirected graph $G(V, E)$, we say that *two nodes a and b are connected* if the set of edges E contains a subset of them that form a path between nodes a and b.

A *graph is connected* if every pair of vertices in the graph is connected. A *connected component of a graph* is a maximal connected subgraph of G; in this case, each node and each edge belong to exactly one connected component.

A simple undirected graph is a *tree* if in it any two vertices are connected by exactly one simple path. A graph is a *forest* if it is a disjoint union of graphs that are all trees, which means that in a forest all the connected components are trees.

Given a connected, simple undirected graph $G(V, E)$, a *spanning tree* T of that graph is a tree such that $E(T) \subseteq E(G)$ and T connects all the vertices of G. Given a weighted graph $G(E, V, W)$ we can enumerate all its spanning trees and order them according to the total sum of weights of all edges of the tree. Accordingly, a tree is a *minimum spanning tree* of a weighted graph (denoted as $MST(G(E, V, W))$), if it is a spanning tree of G with the total sum of weights of its edges (its weight) being less than or equal to the weight of every other spanning tree of $G(E, V, W)$.

A weighted graph $G(V, A, W)$ is called *weighted directed* or a *weighted digraph* if it has *arcs* instead of *edges* connecting nodes. Often used to represent some sort of relationship, the set of arcs is called $A(G)$.

1.3.5 Finding Influential Members of Social Networks: Dominating Sets

In the area of social media and network analytics, the representation of a problem via some sort of “graph equivalent” is pretty obvious for both our marketing colleague, Ben, as well as for Anna, our computer scientist. Thus it is pretty obvious that the problem of selecting the most representative users of a social network may indeed lead to some optimization problem defined on graphs.

Assume, for instance, that you are running a marketing campaign and you know that a friendship relationship exists among a set of individuals. Your budget is limited and you may be pressured by a boss that you can only have enough funds to reach out to 15 individuals. Given a graph that represents these pairs of friendships, is it possible to find 15 nodes (where each node corresponds to a different person) such that for each node that has not received directly a marketing message, at least one of their friends has received it? Obviously, this is another decision problem (it

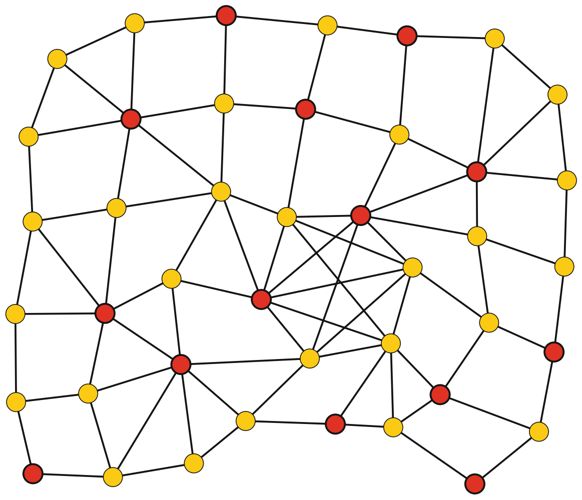


Fig. 1.3 A graph with 42 nodes and a dominating set of vertices of 14 vertices (marked in red). If this is a social network, those 14 vertices can be used to start a "word-of-mouth" campaign by just targeting them first. If we seek to reduce marketing costs, we would then be interested in finding the dominating set with the smallest number of vertices (hoping the message expands through the network)

just requires a *"Yes"* or *"No"* answer). Anna will probably enunciate it as *"Given a graph G*, *does it have a 15-vertices dominating set?*"

Well, the answer will depend on the graph we are receiving as input, of course. Again, Anna will call this graph *"the instance"* of the problem. In Fig. 1.3 we see one such example. It is clear that for this graph the right answer to the question is *"Yes"*. You can actually do even better, in this case we can use 14 vertices and do what has been required. Actually, we have still "one extra to spare", adding any extra node to make it 15 will obviously still be a feasible solution (some nodes may be dominated even more than once).

1.3.5.1 Domination…at a Distance

Ben's original idea and the cooperation with Anna went down the drain. In fact, the direct marketing idea is not sensible for larger networks. Ben knows he might be in trouble with his boss. After a few minutes running a heuristic with his computer he found that he needed to contact approximately 923,567 people with his direct marketing campaign if the goal was that for any pair of people that are friends in the social media at least one is directly contacted. He hoped that the exact algorithm the computer scientist team was running would give him better news. However, after 2 months of computing time, and a significant dollar bill paid to Amazon Cloud Services, the exact algorithm returned an optimal solution of 923,117 (yes, this can happen…).

Trembling before presenting the results, he has another plan. He has heard that old idea of Frigyes Karinthy about the "six degrees of separation", the one that basically states that everyone is six or fewer steps away, by way of introduction, from any other person in the world. Perhaps this is also true in the social network.

What if we if require to identify the minimum number of nodes to be dominated such that if there is a node that is not directly dominated it will not be "at a distance greater than d". When we have $d = 1$ this is the original problem we were discussing before, but searching for optimal solutions for increasing values of d, Ben would eventually find a value of d for which the cardinality of the dominating set "fits the budget" and the direct marketing campaign could be possible. If d is still small, word-of-mouth may help to propagate the message.

This type of problems belong to the area of "distance domination" [87] or the (k, r)-*centre problem* [53]. We have introduced this example to show that a dialogue can exist, based on data, such that we could be using mathematical modelling for a campaign for direct marketing. Exact algorithms can give a clear indication of what is feasible or not, the cost it would require and then, consequently, decision makers could judge other alternatives.

Problems related to domination at a distance may be new in marketing, but not in business analytics (where location problems abound). If we look further into the past, and even the Emperor Constantine the Great had to consider them to protect the Roman Empire [178]. There are several variants of these domination problems being studied. We anticipate that combinatorial problems finding influential nodes in graphs are likely to become a hot topic of research in the near future [20, 80].

1.3.6 Cleaning Data by Removing Its Contradictions: Vertex Covers

Other problems can be transformed into a graph optimization problem. We give here another interesting example. We know that the use of raw data, without proper preprocessing and analysis, may be highly misleading for business intelligence purposes. If a problem is said to be an *offline* problem, you can safely assume that you have all the information that is required for decision-making (i.e., you can design an algorithm that has the complete instance, it has all the input that is required to produce a solution).

In real scenarios, data scientists spend significant amount of time "cleaning" the dataset. Some sort of internal consistency is necessary before using the data. For instance, you may need to first check if the collected evidence has some sort of "global logical contradictions" or if it is "contradiction free". How can graph optimization algorithms help you with that?

With so many forensic-related TV series, perhaps we can introduce the problem using a crime-fighting investigative scenario. Let us suppose you have interrogated a number of people who may have information regarding a certain crime. You want to uncover "the truth" (i.e., what has really happened). However, raw data may hide this information in a "cloud of noise". For instance, some of the contradictions may be due to honest mistakes of the respondents, recollection errors or other types of involuntary actions, while others may be originating by a group of individuals purposely trying to misguide the investigation. The interrogation process only gives a number of statements that have originated from a set of individuals, a dataset to work with, it cannot give you "the truth". You need to transform data into knowledge.

We will say, that typically, we can take all the interviews and completely distil from them a set of statements that can be either "true" or "false", while at the same time "everything else remains the same" (i.e., their true/false status is not influenced by the status of any other particular statement). We can then create *an undirected graph* and define a problem on it. Given then this set of statements the undirected graph has one different vertex for each statement we have (i.e., a one-to-one mapping between statements and vertices). An edge will thus connect any pair of vertices v_i and v_j, respectively, representing statements s_i and s_j, if and only if these two statements are in logical contradiction.

Let us suppose that we have generated such graph and we called it $G(V, E)$ and is the one shown in Fig. 1.4. To simplify the presentation, we have drawn the graph on the plane such that there is a vertex at each point on a two dimensional grid and the lines show the edges between nodes. No edges of G are superposed in this layout, although there exists a single pair of edges that cross each other.[14]

If the responses are coming from people who are lying (or just confused, or a mix of both), their statements may likely have several contradictions with other people' statements (or even with their own if obtained at different occasions). Instead of working with raw data, we may be willing to identify *the minimum cardinality set of statements that, if eliminated, could remove all the contradictions of the entire set of statements in our database*. Let's call that minimum number k. Your quest is then to find this minimum number k; you are seeking a solution for the MIN VERTEX COVER Problem in that graph you have just constructed.

If indeed this is a criminal investigation, and assuming that nobody has been forcing people to give these statements, you would be really interested in finding if a small number of people, p, have been responsible for most of these k conflicting statements, in particular if $p << k$, you would call them back for further interrogation and evidence gathering (are they not saying the truth or just confused?).

In reality, this quest is probably too ambitious. We want to find a group of k statements that, if removed, eliminated *all* the contradictions in the whole set of statements. That is a pretty big ask. The person in charge of the police investigation, perhaps, could counter-propose something else. Why do not check for a group of k' statements (represented by a set of nodes V'), of cardinality k' (i.e., $k' = |V|'$) such that, for all vertices not in V', but in contradiction with some statement, at least one of the contradicting statements should be represented by a vertex in V'.

[14] *En passant*, an area of computer science that is concerned with visualization algorithms, creates aesthetically and perceptually informative layouts of data structures such as graphs. Chapter 16 presents one approach and several references can help the reader to have an introduction to the topic.

Interestingly, this second problem is the MIN DOMINATING SET Problem which we have discussed in Sect. 1.3.5. Some people like the intuition that comes from considering that the graph represents a museum, the nodes represent the intersections of corridors and the edges the corridors. In the MIN DOMINATING SET Problem we are interested in finding the minimum number of guards that could just run one corridor and reach any other intersection of corridors, in MIN VERTEX COVER we are interested in finding the minimum number of guards that can "watch" every single corridor (by staying in the intersections and multitask a bit).

We can then expect that the optimal solution for a graph given by a MIN DOMINATING SET algorithm will give less number of vertices than one of MIN VERTEX COVER, and this may not only bring potentially less people for further interrogation, it may be "easier" to solve. Wrong. Both problems are NP-complete, so there is not such a hope at the moment.

1.3.7 Solving Problems by Reaching a Collective Consensus: The Role of "Recombination" of Solutions

How to deal with these optimization problems in real situations? We now take a bit of a departure from traditional exposition of concepts in computer science. It is also an opportunity to honour some early pioneers who have employed a novel way of thinking and found strategies that may have not even required the use of complex mathematics or computing to solve large scale industrial problems.

Back in 1964, when faced with a facility layout problem, Kase and Nishiyama proposed a method that circumvented expensive computing unavailable at the time for them. They proposed a kind of "game" to solve this problem. A set of "rules of the game" were proposed so that company employees interacted by exchanging ideas with the final objective of *collectively* designing a new layout for a company [110]. A group of "referees" and "players" iterate in the alternative "design" of a layout by working independently and by synchronizing to exchange information about their partial solutions to the problem. Together, the team is addressing the task at hand by "recombining their ideas", thanks to a two-tiered system of players and referees.

The proposal is interesting since it is from a time when computers were perhaps not available for them. The authors proposed that the game is useful to motivate the players (employees) interest in decision-making by consensus. It also helped to create a good feedback flow from the employees helping to identify relevant information and basic data that is needed and which could then be translated in strategies to the referees (who may be assumed to be the next level decision-makers in the company hierarchy).

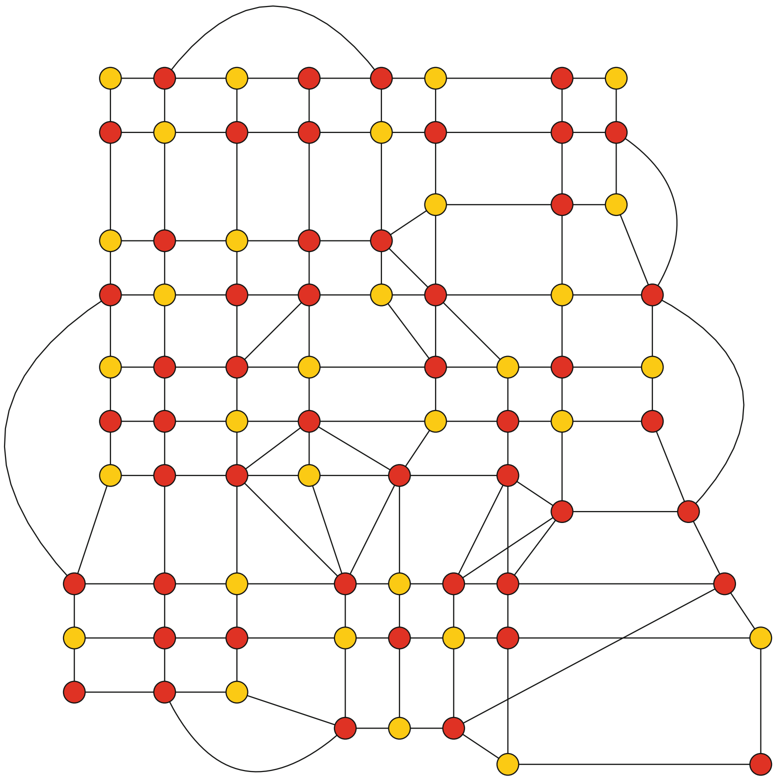


Fig. 1.4 A hypothetical case of an instance of a data cleaning problem modelled as a graph optimization problem. Here we are looking at a graph with 80 vertices. Is it possible to select a subset of the vertices, of a given fixed cardinality k, such that all edges have at least one endpoint that belongs to this subset? A feasible solution of the MIN VERTEX COVER problem on the graph is shown (the vertices are coloured in red)

We highlight the similarities to what we are going to present here. One of the basic procedures that is most commonly used in the field of evolutionary computation (an alternative for the approximate solution of combinatorial optimization problems) is the *recombination* of feasible solutions. For instance, let us suppose that through some sort of constructive randomized algorithm, or thanks to suggestions of the members of your team (such in Kase and Nishiyama's game), two proposed solutions of the MIN VERTEX COVER Problem are available. Let's suppose we are given two solutions like the ones shown in Figs. 1.4 and 1.5. The vertices coloured in red in Figs. 1.4 and 1.5 are two different vertex covers (all edges have at least one red vertex at an endpoint, or have both of them coloured in red). Of course, in both cases there is a bit of "redundancy", we are not illustrating the example with any initial pair of solutions that are optimal in a "local" sense. They may be, of course, readily improved by removing some of the red vertices and still be a vertex cover.

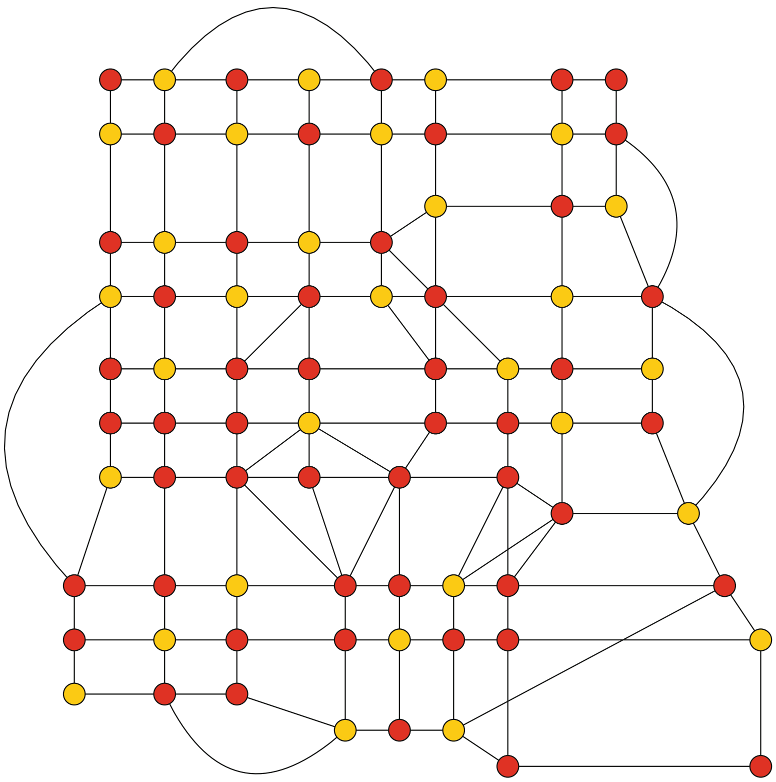


Fig. 1.5 The set of vertices coloured in red constitutes another solution for MIN VERTEX COVER for the same graph of Fig. 1.4

It is thus tempting to think of these two solutions as something that may have been generated by human trial-and-error, and as a consequence, it would have been generated with some "errors", e.g., some unnecessary marked vertices, might have been introduced in the solutions. Imagine that this is a graph of a large museum and the museum's Director aims at finding a placement of guards such that each corridor (an edge) can be guarded by at least one person in an intersection. These two solutions may have been provided by two members of the guard, and probably they are far from optimal. The two solutions have 49 and 50 vertices, respectively. Can we recombine their "ideas"?

The reader can easily check that the marked vertices in red in Figs. 1.4 and 1.5 are indeed vertex covers of the graph (i.e., all edges of the graph have at least one vertex marked at their endpoints). The two solutions are not necessarily *minimal* covers. We can remove the red mark of some of these vertexes (turning it into "unmarked", i.e., in yellow) and the remaining set in red is still a vertex cover of the graph.

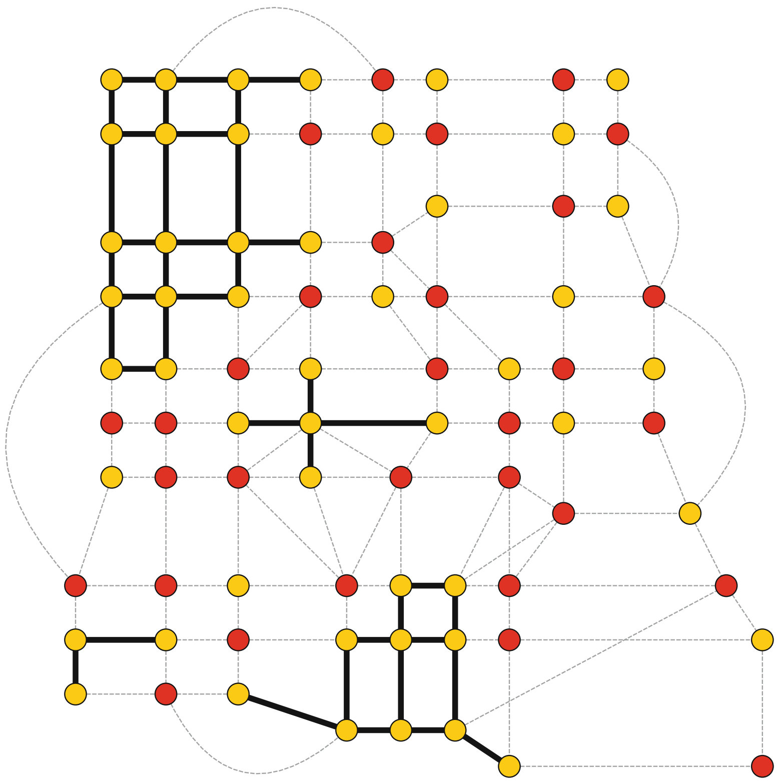


Fig. 1.6 In red, the vertices that are common to the two vertex covers shown in Figs. 1.4 and 1.5. The thickness of the edge indicates that is covered by a red line or not, so the thicker edges indicate those for which there was no consensus between the two solutions about how it should be covered. Thus, four connected components need now to be covered to obtain again a feasible solution of the problem

We will now show how a new solution to the problem can be constructed by *recombining* these two feasible solutions. We will just present one possible algorithmic way of doing this to illustrate the general idea. We start by finding the vertices that are present in both covers.

In Fig. 1.6 we now show in red the vertices that were present in both vertex covers. This set, denoted as $C_1 \cap C_2$ induces a subgraph $G'(V', E') \subseteq G(V, E)$, where $E' \subseteq E$ is the set of edges covered by vertices in $C_1 \cap C_2$ and V' is the union of the set $C_1 \cap C_2$ and the set of vertices that have all their touching edges already covered (i.e., in E'). This subgraph is shown with thin edges and the vertices of $C_1 \cap C_2$ are shown in red in Fig. 1.6. With thick lines we show the edges of G that are not adjacent to vertices in $C_1 \cap C_2$ (both endpoints of these edges are now switched to yellow, as there was no consensus on the two solutions on how to cover them). Note that this graph is not connected and has four connected components.

Interestingly, each one of these four fragments can be coloured by a simple *reduction rule* that can be stated as:

> If the input graph has a vertex u of degree 1, the other endpoint vertex v *must be* in the cover.

The rationale is simple, by adding v to the cover, we guarantee that edge (u, v) is covered; we do not need to include vertex u in the cover and we may have the extra benefits of covering some other edges by choosing v.

1.3.7.1 Safe Data Reduction Rules Can Lead to a Proven Optimal Consensus

We can rewrite this simple rule for the vertex cover problem in the following terms:

> If there is an edge $(u, v) \in E$, where E is the edge set of an undirected graph $G(V, E)$, and v has degree 1, then the graph G has a k-vertex cover if, and only if, the graph G', the graph obtained from G by removing vertices u and v and all their incoming edges, has a $k - 1$ vertex cover.

This reduction rule is then said to be *safe*, and while in this case this fact is pretty intuitive and mathematically proving its safeness is easy; for other rules more elaborate mathematical proofs are needed.

In this case, the repeated application of the reduction rule leads to a complete covering of all the edges of the graph, so for this particular pair of solutions, we have been able to find another solution which is optimal in the sense that, with the constraint of respecting that certain vertices are present in both covers, it has the minimum number of remaining vertices needed to cover the whole graph.

We invite the reader to check that indeed the repeated application of this reduction rule leads to marking another 13 vertices as being in the cover which leads to a total of 44. The step-by-step process is shown in Fig. 1.7. Interestingly in this case the new solution obtained by recombination, shown in Fig. 1.8, is different, and much improved, than both input solutions.

When we can prove that a recombination algorithm is always able to find the optimal solution while respecting a certain property, we say that this is a *dynastically optimal recombination* algorithm [43]. Theorems and theory are called to identify which type of optimal recombination algorithms can be proposed (under certain circumstances); this is a very active field of research for memetic algorithms when hybridizing exact techniques.

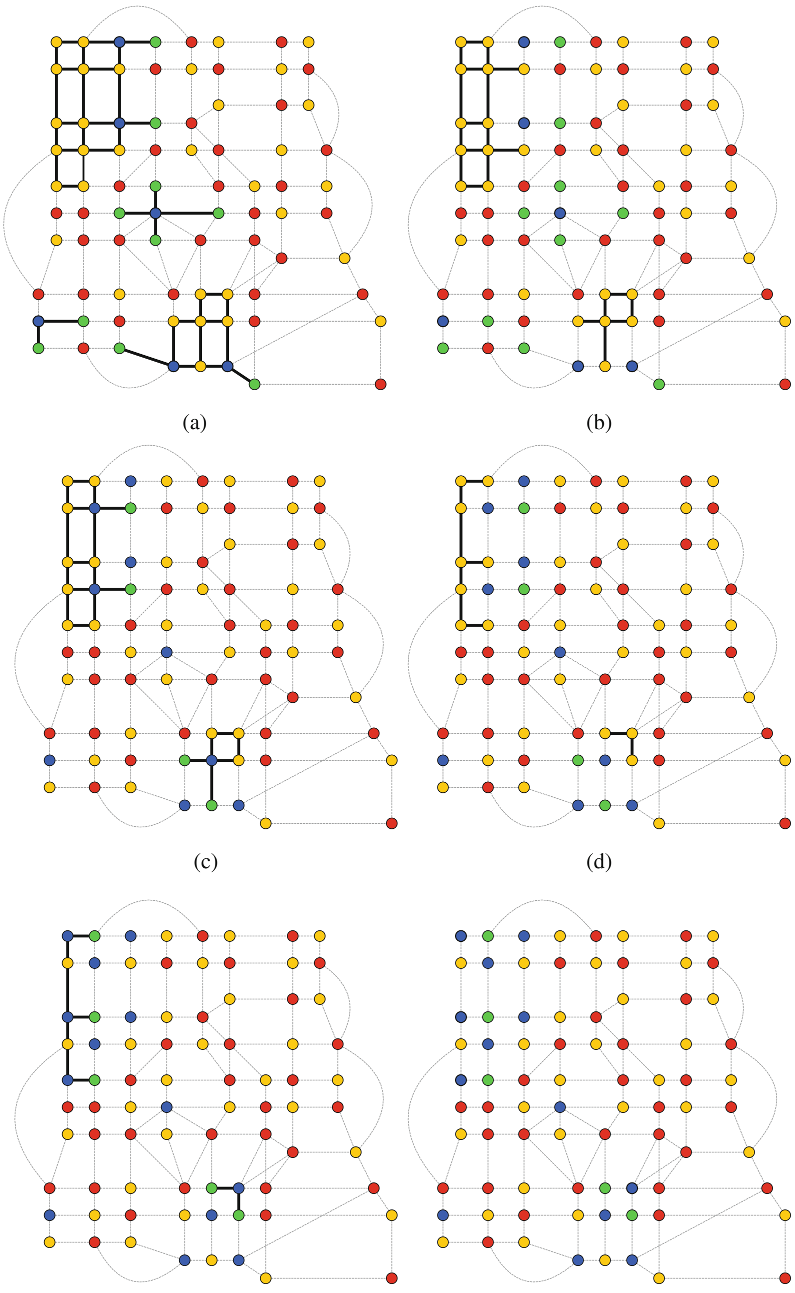


Fig. 1.7 Pictures of reduction rule in action. (**a**) Round 1: selection. (**b**) Round 1: reduction. (**c**) Round 2: selection. (**d**) Round 2: reduction. (**e**) Round 3: selection. (**f**) Round 3: reduction

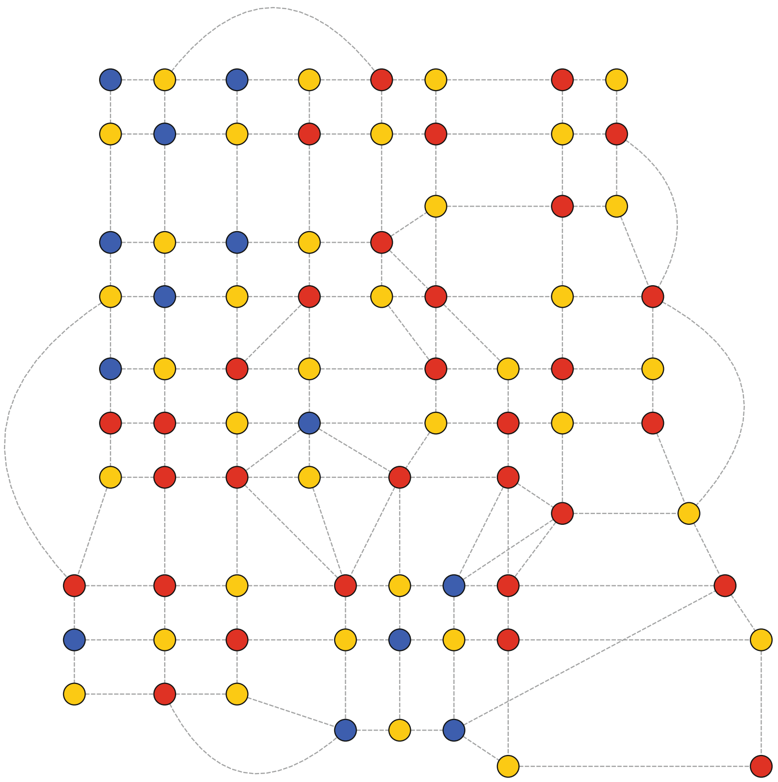


Fig. 1.8 The dynastically optimal vertex cover that can be reached using the two feasible solutions of this problem of Figs. 1.4 and 1.5. Thanks to the use of safe data reduction rules we can guarantee that no better solution exists that satisfies that all common vertices (the consensus, vertices marked in red) are in the new "recombined" 44-vertex cover. The vertices marked to be in the vertex cover by the reduction rule appear in blue. Recombination has then found a solution that it is better than both "parent" solutions of Figs. 1.4 and 1.5 (with 49 and 50 vertices in their covers)

1.4 Algorithms, Approximation Algorithms, Heuristics and Metaheuristics

If an algorithm is a well-defined procedure that will always converge, in a finite number of steps, to the best feasible solution of a given problem, what should we do if we do not have such an algorithm? Sometimes this is not because "we are not smart enough"; actually this situation is not unusual and may reasonably appear early on in the process of dealing with a new problem. Following a discussion from Foulds [70], we can list some of the situations that we may face in practice:

- Both Anna and Ben may come with a great idea for an e-commerce application. However, a nice visualization of the data is required and no algorithm is known for the problem at hand. After a few back-of-the-envelope calculations, Anna convinced herself that there is an algorithm that can do the required job, but it will take a large amount of time, thus prohibiting its use (i.e., she may have proved that the problem is EXP-Complete as discussed in Sect. 1.3.3.2). Depending on the particular input, what can be considered as "acceptable delay" can go from milliseconds to years, it does not really matter, but it just renders the approach "unfeasible in practice".

- The input of the problem already requires to make some sort of approximation. This means that we are confronted with a problem that has a well-defined input, yet the data has to be approximated in some way. This said, "selecting the best" (according to some objective value that ranks feasible solutions in order or preference) may be highly irrelevant.
- The problem has been defined by a number of people, which may have different views on what is good and what is not. For instance, a set of managers may be willing to design a visualization tool for a website that presents several products. However, it is not clear which is a "natural measure" of similarity between the products.
- In the area of personalization of services, part of the input comes from the user. A computer system may need to present alternative solutions (i.e., a website that suggests flights), produce a preliminary ranking based on average patterns of behaviour and then *learn* from a specific user to adapt to the particular needs.

When these situations cannot be resolved, there are several options. We will now address one and leave others for later. We may say that, perhaps, Anna and Ben should avoid the idea of getting optimal solutions in these scenarios and resort to *heuristics*.

1.4.1 Heuristics and Approximation Algorithms

In the fast pacing real world, heuristics are used everywhere. There are many examples, for instance: *"allocate ten percent of the average of the last three years' sales revenues to the marketing department"* (or so they may wish our marketing readers...a hopeful heuristic for them!). These are heuristics in the sense of "rule-of-thumb procedures". However, heuristics for computational problems can be extremely sophisticated and they are an important tool. They can provide managers a quick implemented solution that could make their work competitive, particularly if another company is also employing heuristic approaches. In a commercial battleground, a good heuristic implemented early in a new market niche can make a big difference.

Computational heuristics are methods that are well-defined and, like algorithms, they are step-by-step procedures that, given the input, will *attempt* to give you a feasible solution, and in some cases they also *aim* to give you the optimal one. However, they may provide you with an "escape route" for the four difficulties itemized above. This comes at a price. For heuristics, there is no guarantee that they will give you a feasible solution and, to a lesser degree, an optimal one if there are many feasible solutions and some sort of value which can be used to rank them.

Ben's boss may have been suddenly inspired by this new idea. Optimization can help her reduce the cost of the business. Would it then be possible that, for a problem that we may not have an efficient algorithm that finds the optimal solution, it may still be possible to give "pretty good solutions" with some sort of approximation guarantee? That will make her look good with the company's Board of Directors, she may know how far from optimality her decision was (in the worst case!).

Computer scientists have asked themselves these questions a while back, and in the 1970s the idea started to attract the interest of many. Unlike heuristics, with their recognized problems of lack of guarantee, perhaps we can trade optimality for a different guarantee, one of being "close enough" to the optimal solution up to a constant factor. Let's say, for instance, that there is a guarantee of being at most 10% of the optimal solution for any instance of the problem. That seemed to be an interesting research direction.

From the work by David S. Johnson and other colleagues in the early 1970s, many researchers developed the field of *approximation algorithms*. This vibrant area of research flourished for many years, yet this does not mean that they are suitable for all types of practical applications. However, there might be cases in areas where the computational cost associated with running them may dwarf against the benefits of having some sort of guarantee given on the final result.

Some of the mathematical proofs required to obtain tight mathematical results are very elaborate. The early progress during the early 1970s years lead to promising results on some problems like MAX INDEPENDENT SET, MIN SET COVER and MIN COLOURING. In turn, this catalysed an interest that remains until today. Though these techniques may not be practical in many cases, the field pushes the limits of algorithmic design techniques and researchers create elaborate proofs, algorithms and data structures to obtain the results.

Heuristics and Approximation Algorithms also showed that not only "time" and "space" are important resources in the design of algorithms, also sparingly use of randomness may help as well.

1.4.2 Basic Strategies for Computational Heuristics

For many theorists, in the area of algorithm design, ease of coding is actually of no relevance at all but in the real world it is of great importance. We have a paradoxical situation, mathematical methods that do not have theoretical guarantees of performance are generally neglected regardless of all empirical evidence of the usefulness in practical settings. However, theoreticians are well aware of some of the limitations. Downey, Fellows and Stege, in [58], alert to the challenge to theoretical computer science currently stands when they affirm:

The undeniable successes of sometimes "mindless" and generally unanalysable heuristic algorithms puts computer science theory in an uncomfortable position.

This said, we choose to discuss heuristics before other concepts are presented due to their ubiquitousness in practical computing.

Heuristics are not "mindless", and are based around some main key strategies. Foulds describes four major strategies for the design of heuristics [70].

1. **The Improvement Strategy**—In many cases, a feasible solution is known, or can be quickly computed, for a given problem. If there is an objective function of interest, we may search in the space of possible solutions via an iterative improvement scheme. The input for this type of heuristics is then a feasible solution either suggested or built via another algorithm. A step-by-step procedure is then designed such that the solution is iteratively improved by a series of changes. If these modifications are somehow "relatively small", people refer to this strategy as a *"local search"* [173, 225]. Assume, for instance, that Ben identified the set of vertices in Fig. 1.3 as an influential group in a social network. Anna then could write a program that would iteratively improve on this feasible solution by trying to reduce the cardinality. She may test, for instance, if all possible ways of removing two vertices and including at most one lead to a reduction of either two or one vertices in the dominating set. If no such improvement can be found, you can extend your search trying to remove more vertices [88]. This heuristic strategy then becomes, like others, a really easy thing to implement and, in fact, experienced people on these approaches can quickly produce heuristics that exploit this approach to the full extent of its potential.
2. **The Construction Strategy**—In other marketing and business scenarios, a clean-slate approach is often preferred. A solution can be obtained by adding a small component at a time. For instance, in a delivery problem involving n requests, it may involve starting with a closed partial solution, e.g., a loop that from a depot visits two clients that need to be served and returns to the depot. Initially, these could be those that are further away from each other, so the initial "tour" starts from a depot and visits these two. Then, iteratively, all remaining $n-2$ clients are considered, one at a time, selected with some criteria and added to the route between the pair of existing clients in the tour such that the overall length of the tour is minimized.

   These types of strategies generally employ other mechanisms of "looking ahead", because adding an element to a solution may hinder the possibility of a chain of a potentially more beneficial set of choices later in the construction phase. The final output should be a feasible solution of the problem so, for some problems, it is not unusual to run a construction heuristic followed by an iterative improvement one. The first one guarantees that at least one feasible solution is given, which is already a non-trivial feat for some problems.
3. **The Component Analysis Strategy**—Image databases of consumer interest, for instance, are so large that the only reasonable strategy, for many problem domains, is to partition them into smaller subsets for analysis. Improvement or Construction heuristic strategies may be used to deal with each of the subsets

separately (like, for instance, in problems involving visualizations). There are cases in which the solutions for individual components need to be "aggregated" to build the final solution for the whole database. This may require "resharing" of individual elements to other components leading to a number of subsequent heuristics to mitigate the effects of these new insertions in already established partial solutions. It is one of the many prices to pay when datasets are too large that either time, or memory, would make other more direct approaches not implementable. However, this "divide and conquer" may have its very useful side and, when well employed, could lead to efficient implementations.

4. **The Learning and Adaptive Strategy**—This approach tries to benefit from previously gathered information during the process of searching, constructing or improving solutions. There are a variety of methods to do this. At the end of the spectrum, we have strategies such as the only ones discussed by Foulds, based on the use of a tree data structure. The method can then track previous decisions by coding it with a tree, the sequence of choices can be traced by a path through the tree. It is then the choice of which branch to take governed by a "heuristic function" which in turn can be guided by the outcome of earlier decisions. Some exact methods generate a feasible solution and then continue computing improving on that one. One example of this strategy is *truncated Branch-and-Bound*. However, in the past three decades this strategy has expanded considerably and we now have a number of strategies that also *adapt* through the search process.

1.4.3 Metaheuristics and Variants

In the early 1980s, with the advent of the personal computing revolution, a new type of strategies emerged. A term was later coined for this approach: "metaheuristics". This denomination encompasses a number of high-level and domain-independent strategies that provide a set of guidelines to improve the performance of heuristics.

Some of the very first metaheuristics, for instance, *Simulated Annealing* [116] or *Tabu Search* [77], were so easy to understand and implement that these methodologies quickly started to be applied to many problems. Folklore stories abound high-expenditure in commercial optimization software for solving major logistics problems which were providing feasible solutions but were still rather unsatisfactory from a return-on-investment point of view, but after a few lines of code were included to implement a metaheuristic, the improved systems managed to turn millions of dollars in savings for the associated company.

Today there are a number of different metaheuristics being applied in business and customer analytics apart from the two mentioned before. Other types include genetic algorithms, ant colony optimization, memetic algorithms (part of the field of Evolutionary Computation), variable neighbourhood search, (adaptive) large neighbourhood search, etc. How can metaheuristics empower the different strategies for heuristic design? We can discuss them in the context of Foulds' classification:

- **Metaheuristics and the Improvement Strategy**—To use methods based on iterative improvement we need to find clever ways to "represent" feasible solutions of a problem. There are strong requirements to fill too. For instance, if we have an objective function (and the task is either to find its, possible multiple, global maximum or minimum) at least one of them should be a member of the space of represented solutions (sometimes called the *configuration space*).

  Metaheuristic methods may help a basic Improvement Strategy (like a simple local search technique) by preventing that it may get "stuck" in a bad region of configuration space (according to the objective function), a suboptimal solution, or by bringing the search "closer" to where the global optima may be located in the space. They do that by different techniques, for instance, by allowing a temporary worsening of the solution currently being considered (e.g., both Tabu Search [77] and Simulated Annealing [116] work that way). Alternatively, metaheuristics could be designed to "restore feasibility". For some problems some representational schemes could be very self-evident and improvement techniques be very efficient on them, at the hindrance of creating configurations that do not represent feasible solutions. In that sense, metaheuristics help to restore feasibility.
- **Metaheuristics and the Construction Strategy**—We have mentioned before that a construction-based heuristic can guarantee the creation of a feasible solution, which an improvement strategy can subsequently try to ameliorate. This tandem process can be iterated as well. After the improvement strategy has somehow "stagnated", the process can be iterated by selecting other "seed" parts and running the construction-based heuristic again. Alternatively, if there is a guiding principle for the constructive heuristic, this principle can be slightly changed, leading to some sort of adaptation of the process. We will then discuss this more in the context of the final strategy which involves learning. Another famous technique is called GRASP, from Greedy Adaptive Randomized Search Procedure [65, 66, 179, 183]. A state-of-the-art survey of this methodology can be found in [67]. In some sense GRASP is an example of a metaheuristic that tries to combine the Construction, Improvement, and Learning and Adaptation Strategies in a single package.
- **Metaheuristics and the Component Analysis Strategy**—It is perhaps a good idea to discuss one particular problem from the area of logistics in which this approach could lead to a good combination of the three strategies into a metaheuristic. Suppose you have a single depot and a fleet of vehicles, each having a maximum load capacity, and you quickly notice that you will need to use many vehicles to satisfy the demand. You need to deliver a set of orders, there are no time restrictions when the delivery needs to be done. First, you partition the set of geographical locations into a nearly equal number of orders. Then you can use a constructive heuristic to trace routes, and optimize the total travel time with an iterative strategy. Most likely, for some of the vehicles you have exceeded the maximum capacity, rendering your solution infeasible. Thus you can use an iterative improvement heuristic to "shuffle" customers among the different vehicles so that the maximum load is not exceeded. When for a

vehicle you achieve that, then you try to optimize the route according to another iterative improvement method that reduces the length of the travel. However, all is dependent on your initial component separation, the partition of the set of customer orders. In a metaheuristic, you need to plan again this component analysis. You may need now to change the partition, perhaps in a more drastic way, and reiterate the process. The next strategy gives one possible way to achieve this.

- **Metaheuristics and the Learning and Adaptive Strategy**—There is an existing great potential in mixing the learning and adaptive strategies with the other design strategies. There are a number of metaheuristic methods that combine them. For instance, soon after Tabu Search was proposed, other techniques like Reactive Tabu Search, for instance, were proposed. In addition, a number of other techniques involving techniques of machine learning followed this trend aiming at helping to adapt the search procedures. The previously mentioned GRASP, from Greedy Adaptive Randomized Search Procedure also contains simple learning/adaptive mechanisms. Other metaheuristics that make use of randomization and adaptation use a population of solutions and are related to the area of *Evolutionary Computation*.
- **Hybridize Always: Memetic Algorithms**—Finally, there is one strategy that aims at combining all the elements of the other strategies and also includes algorithmic design elements from other areas. The key idea is that the hybridization of the techniques would probably give benefits when the individual elements are selected such that a "synergy" would occur. The book dedicates several chapters to this approach (i.e., "memetic algorithms") so this topic is covered in at least a couple of sections. We thus refer to Chap. 13 that covers the essentials in some depth, points to the relevant literature and contains a survey of current trends and applications in business analytics and data science.

1.4.4 Introducing Genetic Programming for Classification and Model Building

We have mentioned Memetic Algorithms, and there is another technique from the field of Evolutionary Computation that we will introduce now. We have used it in several book chapters and it is relevant to include early into the discussion as it is potentially very intuitive to understand.

It should now be clear to the reader that finding good algorithmic and/or heuristic methods for a problem requires insight and a fair bit of knowledge of computer science tools and study. The natural question these days, for almost everything, is: "Can *it* be automated?" The "it" refers to many things. With every job in jeopardy, perhaps it is also the case that significant parts of the "invention" of computer science can be automated, at least in part. *Can algorithm and heuristic design also be automated?*

In fact, it is far to be a new question. Computer scientists have been discussing this idea for quite a while and a lot of research exists in this area. The unquestionable imagination of Alan Turing was there first [221], followed by Richard Forsyth's BEAGLE (Biological Evolutionary Algorithm Generating Logical Expressions) [69]. Forsyth, almost 30 years later than Turing, shows that evolutionary systems allowed the generation of rules for classifying heart patients (as deaths or survivors), Olympic finalists (as long-distance runners or sprinters) and also on sports, categorizing games into draws and non-draws. Another early pioneer is Cramer [45].

In 1992, John R. Koza published a book titled *"Genetic Programming: On the programming of computers by means of natural selection"* [122] which was then followed by other major contributions in 1994 and 1999 [123, 125]. The field had a dramatic evolution since 1992. By 2010, Koza had already assembled a list of 77 problem domains in which a solution provided as a result of Genetic Programming, on a major identified challenge, was already "human competitive", meaning that it is competitive with human-produced results [124]. Genetic programming applications continue to highlight the power of the technique and challenge human performance, including the optimization of software and its performance [38, 40, 128–131].

In general, a GP method represents solutions as *tree structures* (we note other representations exist, but we will try to keep this introduction simple by referring only to this case). Following the standard graph notation, there are vertices and edges in these structures. The vertices that have degree one are called the *leaves* of the tree and the other vertices are said to be *internal*. These internal vertices have associated a mathematical operator. These operators depend on the problem domain, they could be mathematical, e.g., addition, represented as "+", subtraction ("−"), multiplication ("*"), division ("/"), etc. In other problem domains it could even be logical operators, or even primitive algorithms, etc. Each of the leaves will have associated one of the variables associated with the study of which we have data available. As said before, other building blocks associated with the internal vertices could be logical operators (e.g., AND, NOT, OR, XOR, etc.), mathematical functions (e.g., trigonometric functions, exponentiation, etc.) and many others.

Given a task of interest, the key idea is to associate these structures to "programs" performing "actions" which are then "evolved" in a computer to complete a certain task or to optimize some function of merit. There is often quite a generic evolutionary algorithm associated with this process, so these tree structures are generated by processes involving "recombinations" of their structures or "mutations" (i.e., changes applied to small parts of these solutions in a random manner). The overall idea is that, thanks to a guiding function (also known as a *fitness* function) we will eventually generate a program structure (a tree) that performs well for the task at hand.

In general, the trees search for an *equation* of the form $y = f(\mathbf{x})$, where $\mathbf{x} = (x_1, x_2, \ldots x_n)$ is an array containing the variables of the given problem and $f(\mathbf{x})$ is a function of the variables only. The left side of the equation, represented by y, may be a numeric *class label* (like in our example of Table 1.1 in which we had two classes of objects), or a real-valued variable (e.g., it could have been the proportion of objects sold of each particular type). For instance, if we are studying the one-out feature set problem of Sect. 1.3.2, and in particular the instance shown in Fig. 1.1, then the class label y (which can take the values *"sold"* or *"unsold"*) would need to be mapped to the values "1" and "0", respectively. Then a GP method can be invoked to solve this problem. If, for instance, only multiplication, addition and subtraction operators are allowed at internal vertices, one possible solution of the problem that might be obtained is $y = f(\mathbf{x}) = 1 - x_2 * x_5$, which takes the value "0" (unsold) if and only if features 2 and 5 (here represented by x_2 and x_5) are both "1" and are valued "1" (sold) for all other cases.

In this example, we are dealing with a *classification* problem, and it could be easily extended to several labels which are then mapped to different numerical values. For the cases in which y is a real-valued, or continuous variable over some interval (for which we know the value of f at only some particular points) we have a *regression* problem. Arguably, our already more mathematically fluent readers will quickly find some analogy with *numerical regression* in which are given a mathematical equation representing a model of interest and the task is to identify a set of parameters for which such a model "adjust" a given training data. In this case we talk about *symbolic regression*, a particular type of regression analysis in which we search a space of mathematical models (characterized by the trees that the GP system can generate) which produce approximations of the "unknown" function $f(\mathbf{x})$ such that, on the set of given pairs of the form $(\mathbf{x}^{(\mathbf{i})}, y_i)$, the function $f(\mathbf{x}^{(\mathbf{i})})$ minimizes some quantifiable divergence from the respective "expected" values y_i. Different alternatives are then consider to find a function $f(\mathbf{x})$ that "does well" for all the pairs, and this is modelled as different types of optimization problems.

In several sections in the book we will see examples of their applications to problems that relate to consumer behaviour for classification and regression.

Current GP methods may allow to use more than one objective. For instance, for a problem that has a binary outcome we would aim to find models that maximize the *accuracy*, the *balanced accuracy*, the *F1-score*, the *Matthews Correlation Coefficient*, *Confusion Entropy* other quality measures of interest for a classifier [106]. We refer to Chap. 20, Sect. 20.3 for the definition and a discussion about these metrics. At the same time, we would like to minimize the number of variables used. The search of new models could be biased or unbiased by the selection of an initial population of candidate solutions; however, "pure" symbolic regression generally avoids having a seed model for initializing the search.

There are many software solutions available for symbolic regression that implement GP and their variations.[15] In this book we have used the academic version of a package called *Eureqa* produced by the company Nutonian[16] (acquired by DataRobot in May of 2017). This package got a lot of attention after the publication of a paper in the journal *Science* in which the authors "reverse engineer" the laws governing the mechanical behaviours of several systems from experimental

[15]http://geneticprogramming.com/software/.

[16]http://www.nutonian.com.

data [199]. Thanks to its free academic licence, it has allowed us to conduct symbolic regression analyses and we present them in several chapters of this book.

In *Eureqa* the user selects an objective function which the GP will aim to optimize. Its output of is a Pareto optimality curve which trades between the model fitting capacity and its "complexity".

As Chap. 5 explains in more detail, *Eureqa* also uses what are called "building blocks" that the user selects based on the type of data they are using. For instance, in the example following below, we used these building blocks: *integer constant*, *introduction of an input variable*, *addition*, *subtraction*, *multiplication* and *division*. As for the objective function, we selected "Absolute Error".

1.4.5 Genetic Programming for Predicting Who "Churns"

In Chap. 20 we present a case study in which the task is to find a way to predict which clients would "churn", i.e., leave a telecom service company, and which of the customers are highly to be "non-churners" (they remain with the company). The new method we use in that chapter is based on a different technique and we also compare with the results provided by the GP approach. In this section, we just want to illustrate on how GP generates models for churning behaviour. We refer to Chap. 26 for details on this dataset.

To give an idea of how GP works, let's see the following "equations" that Eureqa generates. When they evaluate to "$True$" this means that the client is predicted to churn. The first model to consider is the following:

$$\begin{aligned} Churn = ((Int.l.Plan = No) + Intl.Calls - (Int.l.Plan = Yes)) \\ \leq (Int.l.Plan = Yes) \end{aligned} \tag{1.1}$$

Note that there are two possibilities to discuss, either the client has no International Plan or not. We will first consider the latter. If the client has no International Plan, then $(Int.l.Plan = Yes)$ is found to be $False$, returning a 0, and since $Intl.Calls$ is always a positive integer valued variable, then there is no way that this inequality is satisfied (since $(Int.l.Plan = Yes)$ is $True$ returning a value of 1), so the prediction is that the client is a "non-churner". If the client has an International Plan, then $(Int.l.Plan = Yes)$ is found to be $True$, returning a 1, so $(Int.l.Plan = No)$ is found to be $False$, returning a 0. In this case the model predicts that the client would be a "churner" if the inequality is satisfied, meaning that the number of International Calls made in the period in being sampled, is one the three possibilities (either 0, or 1, or 2). For larger values of the number of $Intl.Calls$, since this is always a positive integer valued variable, then there is no way that this inequality is satisfied, so the prediction in this case is that the client is a "non-churner". We show how the GP method represents this model in Fig. 1.1.

As the reader can appreciate, we do not argue for the usefulness of this model, it is one among hundreds of millions of possible models that the GP code searches in a matter of minutes in its quest to find a good predictive one. It is a very simple model and, most likely, perhaps it can only explain a fraction of the "churn" behaviour. This is reasonable to expect as the model seems to be entirely related to two aspects of the user experience and interaction with the service.

Other co-evolving models can be more complex, for instance:

$$\begin{aligned} Churn = & \frac{(VMail.Plan = No * Day.Mins * Day.Charge * Eve.Charge)}{(Day.Mins + Eve.Mins)} \\ & \geq Area.Code \end{aligned} \tag{1.2}$$

When preparing the data, we have deliberately included the $Area.Code$ as a variable. The GP may, or may not, select this characteristic to build a model. The previous model did not use it, while this one has. Some other studies have left it outside of the scope in the generation of models as it was supposedly not contributing to a better understanding of user behaviour. It is known, however, that in the USA some area codes may have a non-random distribution with several highly populated areas of the states of California and New York having high values for the $Area.Code$ value. It may be the case that this model has found some benefit of including some sort of regional information, together with costs and durations of calls, to produce a prediction.

Now, to simplify the notation (as the next models are fitting better but have more variables) we will use the following convention: $Day.Mins$ (DM), $Night.Calls$ (NC), $Night.Charge$ (NChrg), $VMail.Plan$ (VP), $Eve.Charge$ (EV), $Intl.Charge$ (IC), $Area.Code$ (AC), $Int.l.Plan$ (IP), $Intl.Mins$ (IM). We then have two other equations to consider.

$$Churn = (DM + NC + (VP = No) * EC * IC) \geq AC \tag{1.3}$$

which can be seen as the tree structure of Fig. 1.9b and finally, the last equation indicates that $Churn = True$ if, and only if, the value of the variable $Area.Code$ for a given customer is strictly smaller than the value given by

$$\begin{aligned} & DM + EC * IC \\ & \quad + \frac{\left(VP = No * DM * EC * NChrg + IP = Yes * (IM * IC)^3\right)}{AC} \end{aligned} \tag{1.4}$$

We show Eq. (1.4) as represented by its corresponding tree structure with the aid of Fig. 1.9c.

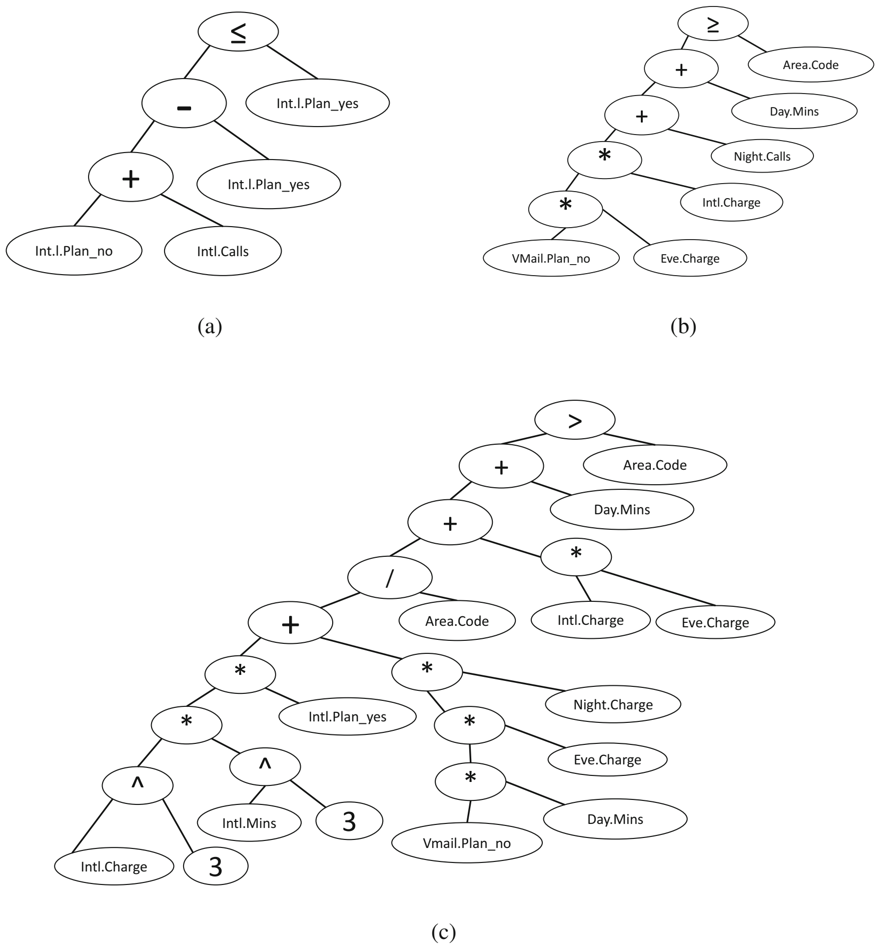


Fig. 1.9 Three different models for predictive churning behaviour in a telecom case study. More complex models have better predictive capabilities but may be more difficult to understand. (**a**) A tree representing Eq. (1.1). It has four leaves and only uses two variables. (**b**) A tree that represents Eq. (1.3). We note that the variable $Area.Code$ is now part of the solution. (**c**) A tree that represents Eq. (1.4). We note that $Intl.Plan$ acts as a sort of "gatekeeper". If this is equal to "Yes", then churn behaviour becomes highly dependent of the value of either $Intl.Mins$ and $Intl.Charge$ to the power of three. This indicates a clear need to address these costs to consumers who have the plan and, probably, also the need of customer-specific plans (as for some the charges are an issue and for others the usage). *Market segmentation* is a technique that is employed to address these issues and it is discussed and applied in several chapters of this book

1.4.6 Mathematical Models and Interpretability

It is hard for somebody trained in the physical or exact sciences to easily accept these equations as a "model" of a phenomena, perhaps due to the use of the word in different fields. Physicists, for instance, quickly search for the “units” in an equation, and these do not seem right in that regard. The first term of Eq. (1.4) is DM which is measured in minutes, and the second term is $EC * IC$ which could be measured in “US Dollars squared”. Clearly, interpreted like this, these individual models will make very little sense. Some may give some insights and many clearly not.

It could be possible, as an alternative, to restrict the domain of models that the GP code is searching to some in which there is some sort of correct “balance” of units of measure. We are not aware that there are many researchers investigating in that direction though, whether restricting the GP to some particular types of model classes, according to some “hints” [3–7], would actually increase its prediction capability in generalization, i.e., given more accurate answers (for classification problems) or more accurate values (for numerical regression problems). Due to previous experiences in learning in neural networks [8, 151, 204] we do expect this also to be the case in “learning using GPs”. Instead, the general practice of a large number of practitioners of machine learning and artificial intelligence methods concentrates in “just finding” a suitable encoding of the data into the input of their adaptive technique (a GP, an artificial neural network, etc.), and then they adapt it (i.e., they produce changes that “train it” using the data). This process generates mathematical expressions that, when evaluated, can help to deliver predictions. These equations are sometimes considered to be *“weak learners”*, rules-of-thumb classifiers that may not necessarily be working very well.

Other more complex machine learning and AI systems can have better predictions at the risk of transforming into *“black boxes”*, hindering interpretability for us, mere human mortals. Even if we can follow the computation of these “internal representations” of the models they are just too intricate to translate it into meaningful natural language. We know about these limitations of machine learning and AI systems for many decades, but now the discussion has become vox populi due to the apparently imminent introduction of self-driving cars in our roads and their legal and ethical implications.[17]

One thing is sure, the study of the automatic generation of mathematical models and interpretability in machine learning is going to be a hot topic of research in future decades [144, 175]. As we will see in another chapter of this book, there is a trade-off between accuracy and interpretability [196].

1.4.7 Recombination and Mutation in Genetic Programming

Since GP is based on the evolutionary computation paradigm of creating, selecting and storing a population of solutions, it is pretty obvious that there should be a mechanism to “recombine” two solutions. In the case we are discussing, such a process produces new models. In Fig. 1.10 we give an example of such a process.

[17] https://www.technologyreview.com/s/604087/the-dark-secret-at-the-heart-of-ai/.

Two solutions are selected (by some algorithmic/heuristic decision process), called *Parent 1* and *Parent 2*, and then two new solutions are created. The selection of internal vertices at random in the two parent trees is followed by the swapping of the corresponding subtrees that start from that vertex. In this case, the operator of that vertex is also swapped.

It is hard to imagine that such a crude procedure for sharing information could indeed lead to the generation of better models. In fact, in most cases, the models generated are indeed far worse than the two parents. After iterating these processes millions of times, the individuals in the population somehow "co-evolve" and it is expected that the individuals' trees share some similarity for the internal vertices and structure near the root and gradually diverge towards the leaves. Recombination may become more useful near the end of the run, when we could see the process helping to "fine tune" a particular model. While we have observed this behaviour in several successful implementations of GPs, it may be the case that a less performing GP lacks this behaviour and consequently a new design of the representation of the solutions is needed.

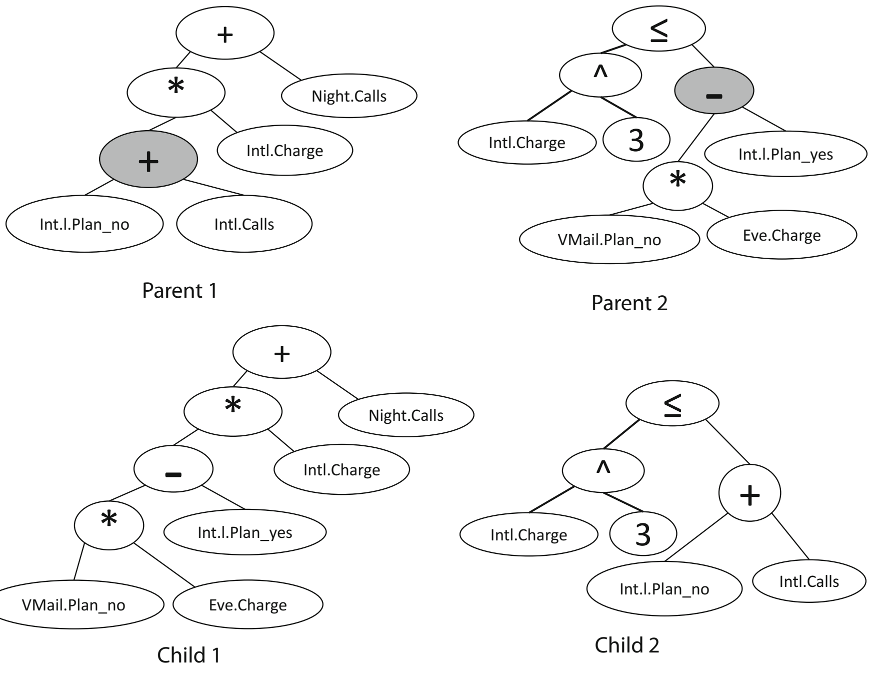


Fig. 1.10 Recombination in Genetic Programming. Two solutions, represented by the two trees labelled *"Parent 1"* and *"Parent 2"* recombine parts of their structures to generate two "children" solutions. This is achieved by selecting at random an internal vertex to both trees and swapping the subtrees, including the selected vertex. This is, in general, a highly disruptive recombination mechanism. Other representations and more sophisticated recombination algorithms also exist in the literature

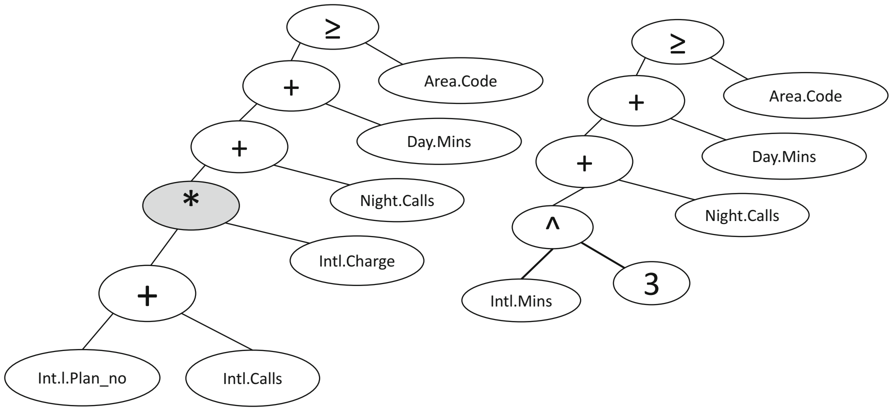


Fig. 1.11 An example of a mutation method in Genetic Programming. The tree to the left of the figure is chosen for mutation, generally according to some prescribed probability. A new solution is created by first identifying a vertex at random (marked in grey here) and then creating a new subtree from that vertex. As in the case of recombination, this procedure can be very disruptive and special problem-domain specific technique may be applied

In addition, GPs have operators that *"mutate"* a given solution. In general, such a word is used to a process that generates a new tree structure from a current one by modifying it without the intervention of another solution of the problem (e.g., without the information provided by another tree structure). In Fig. 1.11 we see an example. An internal vertex has been selected at random (in grey) and a new subtree is created in its place. There are many ways by which this can be done. They go all the way from being a highly randomized procedure to some other schemes that use more problem domain, an approach which is likely to be called "memetic programming" [68, 149, 150] in which individual improvement of these structures is also beneficial.

1.4.8 Genetic Programming for Predicting Wine Quality

There are a number of applications of GP for consumer analysis. In [224], the authors use a GP that uses hedonic (liking) ratings for 40 flavours each composed of the same 7 ingredients at different concentration levels. Surrogate models of human taste are created using these 40 samples, and then, using another optimization technique, new flavours are created that receive high liking scores and consistently liked by a cluster of panellists.

The mechanics of a GP code such as Eureqa allows to be used also for *symbolic regression*, which as we said before, is a type of regression analysis. We now need to approximate a function $f(\mathbf{x})$ such that we are given a set of pairs of the form

$(\mathbf{x}^{(\mathbf{i})}, y_i)$ and the values y_i are numerical and could be integer (e.g., like in a Likert scale as it is frequently used in surveys), rational or even real valued numbers.

As a relatively simple example to introduce GP for consumer analytics and the *Eureqa* software, we illustrate with a publicly available dataset on Wine Quality including information on to hedonic characteristics of Greek wines (i.e., aroma and taste). In [181], Raptis et al. provide data for a *training set* composed of 140 wines for which we have information about three quantitative inputs; the barrel usage (the number of refills of a barrel), the barrel age and the distillate age (the maturation period of each distillate contained in a specific barrel, measured in years). For all of them we also have expert ratings (each on a scale from 0 to 10) of the qualitative evaluation of the taste and the aroma of these wines. They also provide the same information for a *test set* of 20 wines. The objective of Raptis et al. in [181] is to find a predictive model, using the quantitative input variables, to "predict" expert ratings of aroma or taste quality.

In Fig. 1.12, we can see the results of one of the equations produced by the GP system (Eq. (1.5)). The figure shows the "predicted" values from the Aroma in the 20 wines of the test set (red columns), as well as the actual test values (in blue).

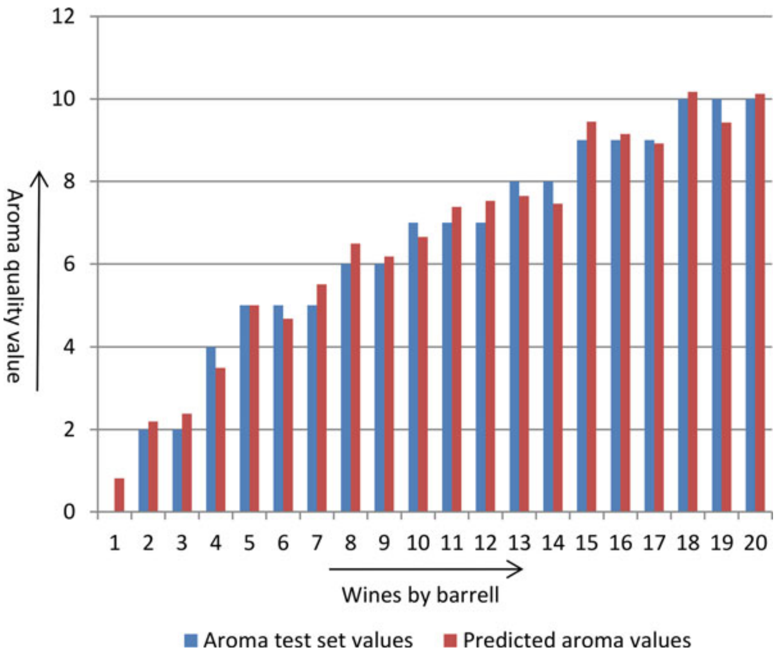


Fig. 1.12 Results of a predictive model that was trained with 140 observations. We show the results on an independent set of 20 samples not used for training. The bar plot shows, for each barrel, the results of the expert panel aggregated opinions on the wine's aroma (in blue, "Aroma test set value", which takes integer values between 0 and 10) and the results of the predictive model found by the Genetic Programming-based system (as obtained with the *Eureqa* software)

$$Aroma\_evaluation = \frac{42 * Distillate\_age}{(10 + Barrel\_usage + 3 * Distillate\_age)} \tag{1.5}$$

The equation was still a fairly “simple” one as its *Eureqa* attributed “size” (an indication of how complex the function is), was 12, as opposed to the best fitting solution of which the “size” was 48. However, what is interesting to note is that for Eq. (1.5), the R^2 Goodness of Fit value is 0.94, and for the best fitting function we found its value turns to be 0.97. It seems that a much simpler model does almost as good of a job at “predicting” the aroma quality attributed by experts as the best fitting (and much more complex) function.

Further, we also used the GP in an attempt to find models to predict wine quality “taste” in the dataset. The “building blocks” and objective function are exactly the same using the same data to predict it. The interesting difference between the experiments for Aroma and Taste is that when predicting Taste, the method performed slightly worse. The R^2 Goodness of Fit value for the best (and most complex) model of predicting Taste is 0.95. However, a model more comparable in terms of “size” and complexity as that in Eq. (1.5) for the taste experiment is shown in Eq. (1.6) and only has a R^2 value of 0.88.

$$Taste\_eval. = (Distillate\_age - 1) + \frac{72}{14 + Barrel\_usage * Barrel\_age} \tag{1.6}$$

The approach used by Raptis et al., who first proposed this problem and contributed the dataset in [181], is based on fuzzy logic and artificial neural networks, consequently less interpretable than these models. Towards the end of the chapter, we will cover some aspects of the latter. Readers may then like to compare the outputs provided by artificial neural systems and those of GP. It is clear that the interpretability of the models of these “weak learners” is a compelling aspect for their utilization. It basically tells us which sets of variables seem to be more important (two perhaps for the aroma and three seem to be necessary to predict taste evaluations). They correlate with domain understanding, i.e., aroma is proportional to maturation and inversely proportional to the times a barrel has been used, while for taste its age is equally relevant.

A GP applied for a classification or a symbolic regression problem is selecting variables (features) that it finds most useful for the task at hand. The selection of which subset is used is based on the performance of the group as a whole. This means that it helps to detect possible synergies between these features. Such an approach for feature selection is called a *“wrapper”*. It has two main problems. When there is not enough samples in your training set, there is a risk to overfit (particularly if you have many features). High-dimensionality is also a problem, if you have too many features the computation of a classifier, for instance, may take too much time. The alternatives are *“filter”* methods. In this alternative approach the selection of a best subset of the features is conducted and completed before we create a classifier or we build a mathematical model based only on those features. In the next section we enter into the discussion of one approach for feature selection.

1.5 Feature Selection and the Marketing of a Presidential Candidate

We will introduce some basic concepts about feature selection in data mining and basic model building with the aid of a very engaging problem domain of timeless appeal.

Nearly 35 years ago, a fortuitous encounter brought together a historian, Allan J. Lichtman from American University of Washington, DC, who had a strong interest in American politics, and a Soviet geophysicist and mathematician who was developing mathematical models and methods for earthquake prediction, Prof. Vladimir Keilis-Borok of the U.S.S.R. Academy of Sciences. At the California Institute of Technology, where both were Fairchild Scholars, they found themselves seated next to each other at a dinner party. They both guessed if the hosts had organized this on purpose as it looked as an odd matching of a pair of dinner guests. But soon they discovered their mutual interest in predicting "big changes". The unlikely research collaboration soon followed, another evidence that a lot can be done at the intersection of sciences with likewise open minded people willing to delve into interdisciplinary research.

Table 1.2 The 12 questions presented by Lichtman and Keilis-Borok

Feature	Question	In 1980
Q1	Has the incumbent party been in office more than a single term?	No
Q2	Did the incumbent party gain more than 50% of the vote cast in the previous election?	No
Q3	Was there major third party activity during the election year?	Yes
Q4	Was there a serious contest for the nomination of the incumbent party candidate?	Yes
Q5	Was the incumbent party candidate the sitting president?	Yes
Q6	Was the election year a time of recession or depression?	Yes
Q7	Was the yearly mean per capita rate of growth in real gross national product during the incumbent administration equal to or greater than the mean rate in the previous 8 years and equal or greater than 1%?	Yes
Q8	Did the incumbent president initiate major changes in national policy?	No
Q9	Was there major social unrest in the nation during the incumbent administration?	No
Q10	Was the incumbent administration tainted by a major scandal?	Yes
Q11	Is the incumbent party candidate charismatic or a national hero?	No
Q12	Is the challenging party candidate charismatic or a national hero?	Yes

In the rightmost column we show the authors' answers (provided in [140]) for the election of 1980 that gave the presidency to Ronald Reagan

Predicting who can win an election indeed seems to be an impossible task and the same can be said about predicting earthquakes. The story is a good example of the concept of *feature engineering* and how it can bring a new perspective to analyse problems for which it is seemingly impossible to come up with a solution.

Table 1.3 The table shows the dataset contributed by Lichtman and Keilis-Borok, with each answer for each election represented in binary (1=Yes, 0=No) in [140]

Year	Q1	Q2	Q3	**Q4**	**Q5**	Q6	Q7	**Q8**	**Q9**	Q10	Q11	**Q12**	Winner (target)
1864	0	0	0	**0**	**1**	0	0	**1**	**1**	0	0	**0**	Incumbent
1868	1	1	0	**0**	**0**	0	1	**1**	**1**	0	1	**0**	Incumbent
1872	1	1	0	**0**	**1**	0	1	**0**	**0**	0	1	**0**	Incumbent
1880	1	0	0	**1**	**0**	0	1	**1**	**0**	0	0	**0**	Incumbent
1888	0	0	0	**0**	**1**	0	0	**0**	**0**	0	0	**0**	Incumbent
1900	0	1	0	**0**	**1**	0	1	**0**	**0**	0	0	**1**	Incumbent
1904	1	1	0	**0**	**1**	0	1	**0**	**0**	0	1	**0**	Incumbent
1908	1	1	0	**0**	**0**	1	0	**1**	**0**	0	0	**0**	Incumbent
1916	0	0	0	**0**	**1**	0	0	**1**	**0**	0	0	**0**	Incumbent
1924	0	1	1	**0**	**1**	0	1	**1**	**0**	1	0	**0**	Incumbent
1928	1	1	0	**0**	**0**	0	1	**0**	**0**	0	0	**0**	Incumbent
1936	0	1	0	**0**	**1**	1	1	**1**	**0**	0	1	**0**	Incumbent
1940	1	1	0	**0**	**1**	1	1	**1**	**0**	0	1	**0**	Incumbent
1944	1	1	0	**0**	**1**	0	1	**1**	**0**	0	1	**0**	Incumbent
1948	1	1	1	**0**	**1**	0	0	**1**	**0**	0	0	**0**	Incumbent
1956	0	1	0	**0**	**1**	0	1	**0**	**0**	0	1	**0**	Incumbent
1964	0	0	0	**0**	**1**	0	1	**0**	**0**	0	0	**0**	Incumbent
1972	0	0	0	**0**	**1**	0	0	**1**	**1**	0	0	**0**	Incumbent
1860	1	0	1	**1**	**0**	0	1	**0**	**1**	0	0	**0**	Challenger
1876	1	1	0	**1**	**0**	1	0	**0**	**0**	1	0	**0**	Challenger
1884	1	0	0	**1**	**0**	0	1	**0**	**1**	0	1	**0**	Challenger
1892	0	0	1	**0**	**1**	0	0	**1**	**1**	0	0	**1**	Challenger
1896	0	0	0	**1**	**0**	1	0	**1**	**1**	0	1	**0**	Challenger
1912	1	1	1	**1**	**1**	0	1	**0**	**0**	0	0	**0**	Challenger
1920	1	0	0	**1**	**0**	1	0	**1**	**1**	0	0	**0**	Challenger
1932	1	1	0	**0**	**1**	1	0	**0**	**1**	0	0	**1**	Challenger
1952	1	0	0	**1**	**0**	0	0	**0**	**0**	1	0	**1**	Challenger
1960	1	1	0	**0**	**0**	1	0	**0**	**0**	0	0	**1**	Challenger
1968	1	1	1	**1**	**0**	0	1	**1**	**1**	0	0	**0**	Challenger
1976	1	1	0	**1**	**1**	0	0	**0**	**0**	1	0	**0**	Challenger
1980	0	0	1	**1**	**1**	1	1	**0**	**0**	1	0	**1**	Challenger

The final column (i.e., the "target" feature) indicates the winner party of the popular vote, with the "incumbent" party victory indicating some stability, or the "challenger" party victory (indicating the presence of some sort of upheaval in the society at the time). These "key" 12 features were expanded to a set of 13 in [52] with an extra one that also helped to accurately "predict" the election of 1912 (this feature was not present in [140])

The central idea they had was to look at the problem from a "geological perspective". *"Stability"* was then linked to the fact that a political party that currently runs the White House would continue there, while *"Upheaval"* was then linked to the case in which the party that challenges the current control of the country wins the election. This radically changes the perspective on the prediction problem. The question swifts. It is no longer whether Republicans or Democrats would win but instead they worked in a complete new representation of the core issue, namely: *"Upheaval or Stability"?*

Keilis-Borok and Lichtman did their "feature engineering" according to this new perspective which, in turn, needed a new representation of the knowledge database. They looked at features whose responses were either "true" or "false" and that the answers had some correlation with the outcome (Upheaval or not) in each election since 1860 until 1980. Features with weak association like *"Has the election occurred during a time of war?"* were omitted. They finally identified 12 features that, together with an algorithm, "retrodicted" the popular vote winners of every election from 1860 to 1980 except the 1912 election.

Table 1.2 has the 12 "key" features. (We note that Lichtman continues working and publishing in this subject and later, with the collaboration of statistician Harry N. Davey at the U.S. Bureau of Labor Statistics, they were expanded to it to 13 features [142] and they slightly differ from these. We choose to illustrate our take on this problem by referring to the 12 features in their original publication [140].) The reader would note that these features reveal the interest on building models that would help to understand politics. Questions related to issues like social unrest, foreign policy, etc., as well as opinion polls were not considered. In [140] we can read:

> The full set of favourable circumstances for continuity in party control of the White House has not changed significantly in the past 100 years.

and later they say

> Our 12 questions seem to be close to a minimally necessary set, given the information included in this study. Of course, our questionnaire could be superseded by other questions not considered here.

These open research questions motivated us to study this problem with a different perspective. Is it the case that these questions are a *minimal* necessary set? Is it possible that a subset of the questions has the same discriminatory power? We note that the comment of Lichtman and Keilis-Borok relates to the nature of the dataset *and* the algorithm they have devised to classify samples (a technique that could be classified as closely linked to kernel discriminant function analysis [195]). Would it be possible that another technique would prove that there is discrimination between the pairs of samples and that a small feature set could achieve this objective? There is obviously no single feature, or not even a pair of features that can do that, so the problem of identifying k features out of those 12 would require some computation (Table 1.3). Table 1.4 shows that a subset of the whole set of questions is a *5-feature set*, so it is time to give a proper formal definition to this problem.

Table 1.4 A reduced dataset from the one presented in Table 1.3

Year(s) of same election pattern	Q4	Q5	Q8	Q9	Q12	Winner (Target)
1864, 1972	**0**	**1**	**1**	**1**	**0**	Incumbent
1868	**0**	**0**	**1**	**1**	**0**	Incumbent
1872, 1888, 1904, 1956, 1964	**0**	**1**	**0**	**0**	**0**	Incumbent
1880	**1**	**0**	**1**	**0**	**0**	Incumbent
1900	**0**	**1**	**0**	**0**	**1**	Incumbent
1908	**0**	**0**	**1**	**0**	**0**	Incumbent
1916, 1924, 1936, 1940, 1944, 1948	**0**	**1**	**1**	**0**	**0**	Incumbent
1928	**0**	**0**	**0**	**0**	**0**	Incumbent
1860, 1884	**1**	**0**	**0**	**1**	**0**	Challenger
1876	**1**	**0**	**0**	**0**	**0**	Challenger
1892	**0**	**1**	**1**	**1**	**1**	Challenger
1896, 1920, 1968	**1**	**0**	**1**	**1**	**0**	Challenger
1912, 1976	**1**	**1**	**0**	**0**	**0**	Challenger
1932	**0**	**1**	**0**	**1**	**1**	Challenger
1952	**1**	**0**	**0**	**0**	**1**	Challenger
1960	**0**	**0**	**0**	**0**	**1**	Challenger
1980	**1**	**1**	**0**	**0**	**1**	Challenger

We highlight the presence of a subset of features that constitute a *5-feature set*. These questions alone can help generate a predictive model for the outcome of the election. Identifying these feature sets in data allows to "compress" information as well, uncovering a common set of characteristics which are jointly present. In turn, political scientists can elaborate theories about their formation and influence in the electorate behaviour

1.5.1 *Let's Get More Formal: The k*-FEATURE SET

We can now return to our marketing motivation. In the general case, a company may have information of several products or services that have been rejected by consumers and, hopefully, the company would have others being purchased and accepted. We then say that the particular case of having only one sample of one class, while all the others are of the different class, is a special case of the problem the k-FEATURE SET problem known as the ONE-OUT k-FEATURE SET (one of the classes has only one sample from it). Following Davies and Russell [50], the k-FEATURE SET is the following decision problem:

k-FEATURE SET

Instance Given a set X of examples (which are composed of a binary value specifying the value of the *target feature* and a vector of n binary values specifying the values of the other features) and a positive integer $k > 0$.

Question Does a set S of k *non-target* features exist such that:

- $S \subseteq \{1, \cdots, n\}$ (i.e., the set S has the indexes of the features in the feature set),

- $|S| = k$ (i.e., the set S has cardinality equal to k),
- no two examples in X that have identical values for all the features in S have different values for the target feature?

The reader can now check that the subset of five features presented in Table 1.4 is rightfully called a 5-feature set for the US Presidential elections data. We now know that feature sets of size $k \geq 5$ exist for this problem, but do we have still open to know if there are feature sets of smaller cardinality? These decision problems (one for each value of k) can be conveniently reformulated as follows:

k-Feature Set

Instance A set of m examples $X = \{x^{(1)}, x^{(2)}, \dots, x^{(m)}\}$, such that for all i, $x^{(i)} = \{x_1^{(i)}, x_2^{(i)}, \dots, x_n^{(i)}, t^{(i)}\} \in \{0, 1\}^{n+1}$, and an integer $k > 0$.

Question Does a *feature set* S, $S \subseteq \{1, \cdots, n\}$, with $|S| = k$ exist and such that for all pairs of examples $i \neq j$, if $t^{(i)} \neq t^{(j)}$ there exists $l \in S$ such that $x_l^{(i)} \neq x_l^{(j)}$?

Table 1.5 An instance of the *k*-Feature Set problem from a marketing scenario in which the first four samples (rows) correspond to objects of one class ("purchased") and the other three samples have not been purchased by consumers

1	1	1	1	1	Sold
0	1	0	0	0	Sold
0	0	0	0	0	Sold
1	0	0	1	1	Sold
0	0	1	1	1	Unsold
1	0	1	0	0	Unsold
0	1	0	1	1	Unsold

Each of the five columns indicate a characteristic that is either present or absent in those products. Each column indicates a different "feature"

Table 1.5 shows a simple example of this problem from a marketing application. For the mathematical formalization, we now mapped also the target feature to binary values (e.g., "*sold*" is represented by a value of "1" and "*unsold*" by a "0" for the data of Table 1.5, or "*incumbent*" is represented with a "1" and "*challenger*" with a "0" for the data of Tables 1.3 and 1.4).

1.5.2 Minimal Feature Sets Help to Identify Patterns in Data

In 2004, which was an election year in the USA, one of the authors of this chapter addressed the issue of identifying if Lichtman and Keilis-Borok's dataset contains a minimum feature set. A manuscript was produced in which the authors studied the application of exact methods to find k-Feature Sets of minimum cardinality for this problem [161].

From the previous section, recapitulating a bit, in Table 1.3 we present the original data and a 5-Feature Set is shown by highlighting the values of five columns in boldface. In Table 1.4 we show how this feature set "compresses" the information available and that several elections have the same pattern of feature state values and, by definition, no pattern of feature set values that we observe in one of the two classes of outcomes is observed in the other one. Some elections differ by only one feature, for instance, the election of 1892, that brought challenger candidate (Grover Cleveland, Democrat) to the White House, only differs in a single value in one of the five features (Q12) from the elections of 1864 (Lincoln, Republican) and 1972 (Nixon, Republican). Indeed, Q12 seems to be a decisive factor (*Q12: Is the challenging party candidate charismatic or a national hero?*), and it is worth noting that Cleveland was one of only two people to win the popular vote in three U.S. presidential elections and the first person in the USA who was elected to a second, non-consecutive presidential term.

The elections of 1980 and 1900 only differ in the value attributed to feature Q4 (*Q4: Was there a serious contest for the nomination of the incumbent party candidate?*). In 1980, the election that brought Ronald Reagan to the presidency, the incumbent was Jimmy Carter who had a troubled presidency (with double-digits figures for inflation, interest rates and an energy crisis). Carter also faced the Iran hostage crisis, the misfortune of an attempted rescue that failed and a perceived deteriorated leadership. Senator Ted Kennedy appeared as a serious contender to be the leader of the incumbent party, with a leadership in the polls of 58–25 against Carter in August 1979, then going to 49–39 3 months later. Kennedy lacked a clear will to confront Carter and he failed to articulate his motivation for the top job[18]; these factors clearly debilitated the Democrat campaigners. In contrast, in 1900 the incumbent was William McKinley who was unanimously nominated for his re-election with no candidate running against him in the 1900 Republican National Convention, and New York Governor Theodore Roosevelt was selected as nominated candidate for the Vice-Presidency by a vote of 925 with only one delegate abstaining. This shows the markedly different outcomes for Q4 for these two elections.

In June of 2004, our team at The University of Newcastle submitted our manuscript on the application of k-Feature Set techniques for the US Presidential Election to the Thirtieth Australasian Computer Science Conference. At the time of submission, George W. Bush was the incumbent candidate and the challenger was John Kerry. Our paper was accepted and the final version was resubmitted before the election took place and in it we predicted Bush's victory. At the Republican National Convention Bush received 2508 votes (with only one abstention, a 99.96% of the votes), so we can safely assume that the answer to Q4 is "No" in this

[18]http://archive.boston.com/news/nation/articles/2009/02/18/chapter_4_sailing_into_the_wind/.

case. The answer to Q5 is obviously “Yes” as Bush was seeking re-election, the answer for Q8 is also “Yes” as during the first Bush administration there were major changes in national policy. In our paper we assumed that, in spite of the war in Iraq and Afghanistan, there had been no major social unrest inside the USA. We acknowledged that protests have occurred in relation to the nation’s foreign policies but that it does not seem to be an unusual position for the USA and we did not consider this was a *major* unrest *within* the nation. This said, our answer for Q9 was “No”. Finally, we considered that the answer of Q12 was “Yes”. We highlighted the fact that John Kerry was decorated 5 times, including 3 Purple Hearts, in the Vietnam conflict and we considered reasonably charismatic (although we acknowledge the use of the term “reasonably” as he was far from the charisma of some other successful challengers of other elections like John F. Kennedy (1960), Ronald Reagan (1980), Dwight Eisenhower (1952) or Franklin D. Roosvelt (1932) as well as the previously cited Grover Cleveland (1892)). This said, the pattern in question for 2004, regarding our 5-feature set was ($Q4 = 0, Q5 = 1, Q8 = 1, Q9 = 0, Q12 = 1$). This is very close to the most frequent pattern that can be observed as associated with an incumbent victory, present in the elections of 1916, 1924, 1936, 1940, 1944 and 1948. Based on a number of further analytical studies, including the generation of other types of feature sets that will be described in other chapters of this book, we made the final prediction: *“Bush will win the election”*. We waited for our fate, as presenting the manuscript in an International conference in Computer Science with a failed prediction would be quite a challenge and a lesson in humbleness.

Luckily for our self-esteem and the confidence in our new mathematical models, Bush won the popular vote for a very close margin. However, we did follow the marketing campaign closely during 2004. Clearly, Q12 was indeed a key question. If the perception of the public can be directed to make a five-times decorated John Kerry, former prisoner of war in Vietnam for 5 years, with 3 Purple Hearts, not look as a “war hero”, perhaps that is where there is some point in converting to the pattern that has been so frequently observed to be associated with an incumbent victory. In fact, John McCain (Republican) had denounced the ads attacking John Kerry as “dishonest and dishonorable” and urged the White House to condemn it as well.[19] Fast-forward 11 years, in July 2015 it was the turn of John Kerry to defend the war hero record of John McCain when he went under attack from Donald Trump.[20] Kerry said: “If anyone doesn’t know that John McCain is a war hero, it only proves they know nothing about war and even less about heroism”.

It is comforting that we still have people who are supporting different political parties yet they do not allow some tactics to be employed in the media and that see each other as citizens full of respect and admiration, if this is something genuine and not also a political tactic. Now from a purely mathematical and data analytics perspective, this dataset allowed us to present a very simple, yet illustrative, example of how a combinatorial approach can give new insights on a complex predictive

[19] http://www.nbcnews.com/id/5612836/ns/politics/t/mccain-deplores-anti-kerry-ad/.

[20] http://www.businessinsider.com.au/john-kerry-donald-trump-john-mccain-2015-7.

problem that spans more than a century. We thus recommend the reader to read the original publication [140] and to compare it with [161] in which several k-Feature Sets of minimum cardinality are found for this dataset. The method presented here is indeed non-parametric and can give important insights. In future sections we will revisit the dataset to present some other techniques and again it will help us to illustrate some of the main concepts.

Donald Trump's Victory in 2016: Could It Have Been Predicted?
Many things happened in 2016, and indeed we witnessed a highly uncommon presidential election. This said, this year may have "broken the mould" of what an election is and how it is planned, but it is illustrative to see what our analysis based on feature sets can deliver.

We refer the reader to Fig. 1 of [161] that presents a decision tree. The top question at the root of the tree is $Q4$ (*'Was there a serious contest for the nomination of the incumbent party candidate?'*). In our view it can be answered as "Yes" as indeed the contest of the nomination of the Democratic party was very serious, with Clinton defeating Sanders and without the party giving any impression of coming in unity to the election. The next question in the decision tree is $Q8$ (*"Did the incumbent president initiate major changes in national policy?"*), again we argue the answer is a clear "Yes" as Obama introduced many major national policy changes during his 8 years as President. Finally, $Q9$ (*'Was there major social unrest in the nation during the incumbent administration?'*) is a matter of discussion. Answering "No" indicates that the decision tree predicts an Incumbent victory for Clinton. We have, however, very little previous evidence of a similar pattern (only the election of 1880). A "Yes" answer leads to three elections that had the three-questions pattern (1896, 1920 and 1968). We can argue that a large and latent "social unrest" existed in the society, and that may have made him "unstoppable" [212]. Film-maker Michael Moore was among the few that pointed at the unrest in the "rust belt" of the Great Lakes (Michigan, Ohio, Pennsylvania and Wisconsin), see his post *"5 reasons why Trump will win"*.[21]

One of Moore's five reasons indirectly points to $Q4$. He mentions the "Depressed Sanders Vote", a backer who will be a voter for Hillary, if obliged by the circumstances but that "doesn't bring five people to vote with her". He doesn't volunteer 10 h in the month leading up to the election. She never talks in an excited voice when asked why she's voting for Hillary". The view is then that in a democracy that does not have compulsory attendance to elections, these voters would not draw in followers and incumbent parties that have serious contests for the nomination will always suffer. $Q4$ is indeed

(continued)

[21] http://michaelmoore.com/trumpwillwin/.

very important, only the election of 1880 can be answered "Yes" (won by Garfield) yet it lead to an Incumbent party victory and in 10 other elections we have victories of the Challenger party. In fact, Allan Lichtman himself pointed at the risks for the Democrats in May 2016 (in "Why President Obama must endorse Hillary Clinton" [141]). In that article he also highlights that Garfield's exceptional victory in 1880 was a marginal one he *"eked out a general election victory by one-tenth of 1 percent of the popular vote in 1880"*.

What does the US Presidency dataset teach us? The application of a method to identify a 5-feature set followed by the generation of decision trees based on these features is an example of the "filter" approach to feature selection and then training of a classifier. However, this simple dataset gives the opportunity to discuss some other issues in marketing and data science. The next sections give connections with techniques normally use to the analysis of data.

1.5.3 Association Rules

In early 2016, Lichtman recommended Obama to endorse Hillary based on what we call an *association rule*. This is basically a statement of the form: "If P then Q". This basically reads "if P is true, then Q is also true". Mathematical logic is the basis of association rule mining [85, 86].

In data analytics, algorithms for "discovering" association rules are in high demand to "mine" the databases (like the one of US Presidents) to discover regularities in the collected evidence [10]. They have traditionally been used by supermarkets to identify regularities at transaction data recorded by point-of-sale systems (an example from the field of *market basket analysis* has been recently popularized by recommender systems like those of Amazon and other e-commerce providers).

Quantitative association rules also exist; for instance, "ten percent of married people between age 50 and 60 have at least 2 cars" (an example from [210]). Litchman's advice to Obama in The Hill's article could have thus been written simply as:

Ten out of the 11 times in which $(Q4 = True)$ AND $(Q8 = True)$ AND $(Q9 = True)$ the Challenger Party won the election.

which is a quantitative association rule involving the target feature and which clearly conveys a very powerful message.

Finding these association rules has then been the quest of many researchers using exact algorithms like *Apriori, Eclart* and *FP-Growth* [90] or based on dynamic programming [238] and heuristics [14, 107, 115, 153]. Chapter 6 discusses advanced techniques in this area.

Identifying small feature sets then leads to a dimensionality reduction and a guarantee that all the possible cases of outcomes are "covered" by an association rule. The one presented to President Obama covered some of the outcomes, but others can be extracted from the decision tree (by starting in the root and working downwards towards a leaf).

1.5.4 The Need for Including a Problem Domain When Mining Rules

Finding association rules is probably not enough to derive useful knowledge. In Marketing, "Market Basket Analysis" refers to techniques that try to identify the products that are more likely to be purchased together with another group of products. Vindevogel, van den Poel and Wets alert that, when applied to market basket analysis, association rules can bring together products that are both *substitutes* and *complementary* so care has to be taken by marketers who need to use domain knowledge [226].

Complementary products have a negative cross-price elasticity: when the demand for one product decreases, the price of the other one increases. Analogously, supplementary products have a positive cross-price elasticity: when the demand of a product increases, the price of the other product increases. Could this have some links to the US President Election? Well, the association rule is a *logical conjunction*, $Q4$, $Q8$ and $Q9$ should all be answered "True" for predicting the victory. None of them are products or "goods". As a marketing campaigners for the Challenger Party candidate, though, we can perceive them areas in which we need to jointly invest as a leverage for victory. The campaign may steer resources so that marketing can influence "customers" to perceive all of them as being "True", trying to reduce the "cost" on consumers to perceive them as being valid. Then, $Q4$ can be perceived to be true if there is a division in the Incumbent party. To create this perceived vision, one strategy is to invest in attacking the most prominent opponent of the Incumbent party before the primaries (and largely ignore the rest). The key idea is to choose the early opponents in a strategic way so that the other party naturally divides itself. $Q8$ is about the past, so little can be done, but a simple message on one clear issue (e.g., "Obamacare") can help to cement that there are nation-wide changes in policy, but they need to be corrected. Finally feature $Q9$ is about "social unrest"; once again, a marketer can capitalize in the opportunities to make this the major issue of your campaign. It is hard to think that this has not been exactly what has happened in the US elections of 2016. Perhaps a variant of Vindevogel, van den Poel and Wets alert to marketing may be needed for some cases in political science.

1.5.5 Alternative Scenarios and Algorithmic Bias

The original paper by Lichtman and Keilis Borok [140] and the analysis using k-Feature sets [161] both employ a mechanism for exploring models. The "learning material" is composed of 19 elections and 12 features and we can explore what can happen if some, or just one, of these samples or features are deleted. In [140] the authors state that they "removed each of the 12 features to obtain five indeterminate projections, instead of one, and up to two wrong projections, instead of one". They concluded: "Our 12 questions seem to be close to a minimally necessary set, given the information included in this study". In contrast, in [161], the authors identify 23 different 5-feature sets, which show that the previous conclusion in [161] is wrong. With each of these 5-feature sets we can construct 23 possible models that perfectly explain the separation of victories into the two groups.

How can this be explained? For Lichtman and Keilis Borok these 12 features are probably the minimal number they require *given the classification algorithm* they use. Feature set reduction, using a formal model like the k-FEATURE SET can lead to alternative models to consider. In a marketing and business environment, a group can then consider all of them without the bias of an algorithm "in the loop" and adjust intervention policies based in all of them. Considerations of alternative scenarios may help for the interpretation of models and even allow the use of multiple learning algorithms (these matters are covered in more depth in Chaps. 18 and 20).

1.5.6 Ensemble Learning and Decision Forests

Although both [140] and [161] use the same "learning material" they differ in the way they produce a call for a new election. In [140] the prediction outcome is the result of a single classifier using all features. In [161] (Sect. 5.3) the prediction for 2004 was made on the basis of discussing four decision trees and two heuristics called PART and PRISM (which were part of a Java-based Open Source package called WEKA of great versatility and popularity [27]).

For a given dataset, the feature set of small cardinality does not need to be unique. In the US presidency election dataset we have 23 different 5-feature sets [161]. It is then possible to generate an equal number of different decision trees (e.g., such a set of trees could be called a *decision forest*) and then, given a new upcoming election, return the result that the majority of the decision trees have selected. Indeed, this approach is very close to one nowadays very popular technique for classification and data mining which is called *random decision forests* [1, 29, 91].

Contrary to a k-FEATURE SET problem-based approach, which aims at identifying the smallest feature set in a dataset (and thus need to address an NP-complete problem), a random forest approach can be seen as a heuristic to obtain several feature sets (which are not necessary of the smallest cardinality). Due to the speed of the basic decision tree generating mechanism of the basic random forest implementation, and the possibility of parallelizing the search, they have become an interesting heuristic for *wrapper-based* feature selection [96, 170, 182, 228].

The technique of creating a "forest" and then to base decisions in the majority of them (or some other function of their output taken as a consensus measure) is an example of a *ensemble learning* approach (i.e., multiple learners classify the data and the outputs are combined for new data in order to improve the performance). In *ensemble averaging* multiple models are created to produce a desired output model. An ensemble-learning method that can be considered a "meta-analytic" method to a traditional problem in business and marketing analytics (i.e., churn prediction) is described in Chap. 20.

1.5.7 The Role of Logic and Asking the Right Questions

Another one of W. Edwards Deming's famous quotes is:

> If you do not know how to ask the right question, you discover nothing.

and it seems very right to cite here. In [140] features are binary (Yes/No, a "True"/"False" answer for each of them). While some of the questions are indeed subjective, taken them as features they still satisfy Wittgenstein's "Independence of facts propositions", quoting him: *"1.2 The world divides into facts. 1.2.1 Each item can be the case or not the case while everything else remains the same"* from his *Tractus Logico-philosophicus* from 1921 (see [230]). Indeed, we learn from[22] that "Out of nearly 200 questions, which were all binary ('Yes' or 'No') variables, the algorithm picked those that displayed the greatest difference between the proportion of the time the variable was 'Yes' for years when the incumbent party won and the corresponding proportion for years when the challenging party won, using all U.S. elections from 1860 through 1976 as the training set". While today many practitioners complain about the "data deluge", we hope they appreciate that these investigators also started with one order of magnitude more features than samples. However, via a useful yet slightly unclear preprocessing stage, they kept those features that had more predictive value in univariate statistical tests and that they were largely independent of each other.

It may be argued that perhaps the most important question was that of the target feature. When they asked *if the Incumbent party will remain*, this pushes the discussion to be about good governance and not about the candidate appeal (with the notable exception of $Q11$ and $Q12$ but note that the presence of both allows for this proposition, of subjective evaluation, still satisfy the independence of facts). Quantitative information that compares the two candidates (i.e., the folklore knowledge that "Americans tend to reject the shorter candidate") were disregarded of consideration even when only eight winning candidates have been shorter than

[22]http://analytics-magazine.org/13-keys-to-the-white-house/.

the opponents in the last 29 elections (from 1900 to 2016) [229]. Such a question, independent of the predictive value, does not match the context of the other ones and would not bring knowledge about the governance process and how it is viewed by the population. We observe that part of the "data deluge" is due to improper use of logic and domain knowledge and how to generate the set of features that would help to extract useful models.

1.5.8 Mutual-Information and Feature Independence

The 12 features of [140] seem to have low correlation across the set of samples. Without details of the preprocessing we can only speculate that this may have occurred (reducing the number of possible features from around 200 to just 12). The need to consider the mutual information of features has been proposed more than two decades ago in the area of learning in neural networks [22] (see also [56, 172]). Large datasets are now posing new challenges which are being addressed by efficient algorithms [235].

1.5.9 Majority-Based Classifiers, You Can Do Better

The 12 features introduced in [140] and the algorithm they originally presented in the same paper were later both modified. Lichtman moved to an alternative approach in which he considered 13 questions (an odd number now) and a majority rule, for a reason that we will make clear below. Following the convention that questions are posed in a way that when answered "True" this favours the Incumbent party candidate chances, the new classification algorithm says that when six or more of the questions are answered "False", then the Challenger party is predicted to win the popular vote. These majority-based classifiers can be seen as follows:

$$f(\mathbf{w}, \mathbf{q}) = sign(\sum_{i=1}^{13} w_i q_i) \tag{1.7}$$

with $q_i = 1$ if the answer to question i is "True" and -1 if the answer is "False", $sign(x)$ is a function that is 1 if the argument of the function x is positive and it is -1 if it is negative, and for all i is also valid that $w_i = 1$ (we will explain why we included this variable later if the reader bears a bit of suspense).

1.5.10 The k-Feature Set as a Sharp Bound for Your Aspirations

It may be the case that you would be interested in finding good association rules and you may consider that finding a proven discriminatory set of features, although of minimum cardinality, it would bring some sort of "overfitting". Consequently, investigating which is the minimum k-Feature Set for your dataset of interest can help you to bound the search for association rules. For instance, in the dataset of the US Presidential election we now know that five features are enough to discriminate between the two classes. This tell us that good association rules with four and, perhaps even three features, could be really interesting to "compress" most of the data in just a few patterns of interest. In fact, the reader can easily check that the association rule

$$IV = not(or(Q4, and(not(Q7), Q12))) \tag{1.8}$$

is quite powerful. We can formally write it using the logic symbols as:

$$IV = \neg(Q4 \vee (\neg Q7 \wedge Q12)) \tag{1.9}$$

It gives an almost perfect discrimination; this rule only predicts in an incorrect way the US Presidential Election of 1880. We know, as it was explained before, that it was a very small and marginal number of votes that gave a different result (Table 1.6).

Table 1.6 A subset of the questions and elections previously presented in Table 1.3

Year	Q4	Q7	Q12	Winner (target)
1864, 1888, 1908, 1916, 1948, 1972	0	0	0	Incumbent
1880	1	1	0	Incumbent
1900	0	1	1	Incumbent
1868, 1872, 1904, 1924, 1928, 1936, 1940, 1944, 1964, 1956	0	1	0	Incumbent
1860, 1884, 1912, 1968	1	1	0	Challenger
1876, 1896, 1920, 1976	1	0	0	Challenger
1892, 1932, 1960	0	0	1	Challenger
1952	1	0	1	Challenger
1980	1	1	1	Challenger

We have removed only one US election (the one of 1880) and 9 out of the 12 questions. The remaining three questions are able to separate all other US elections in the two different outcomes

1.6 Linear Separability

We have seen before that Lichtman moved in later years to a classification scheme of the US elections which was based on a majority rule [52], a particular type of a procedure called "linear separability". We have introduced it by referring to a set of variables $\{w_1, w_2, \ldots, w_{13}\}$ which are all set to be equal to the value of 1 (which look to be a bit of an overkill in the previous section). But what if we allow them to vary, thus becoming "continuous/real variables" then?

In fact, a set of samples in Euclidean n-dimensional space that belong to one of two classes (called $Class_1$ and $Class_2$) is *linearly separable* if an array of real values exists $(w_1, w_2, \ldots, w_n)$ together with a real-valued constant b such that we have

$$b + \sum_{i=1}^{n} w_i x_i > 0 \qquad \forall x \in Class_1 \tag{1.10}$$

$$b + \sum_{i=1}^{n} w_i x_i < 0 \qquad \forall x \in Class_2 \tag{1.11}$$

Then it is clear that functions of the form

$$f(\mathbf{w}, \mathbf{x}, b) = sign(b + \sum_{i=1}^{n} w_i x_i) \tag{1.12}$$

can be used to provide a linear separation that helps to discriminate between the samples. When the sign is positive (i.e., $f()$ returns 1), then we will say that the sample $\mathbf{x}$ is from $Class_1$ (and, of course, of $Class_2$ when $f()$ returns -1).

We leave to the reader the task of identifying if such a function exists or not for the whole dataset of 12 questions of the original paper of Lichtman and Keilis-Borok [140]. We refer to [62] for several of the mathematical methods that can be used.

We now turn our attention to the problem of identifying if such linear separations exist for subsets of the questions. The following example will illustrate that there might be some hope for the case of 12 questions (before doing the maths) as there is 3-feature set that is very good at discriminating elections. We have seen before that the association rule $IV = not(or(Q4, and(not(Q7), Q12)))$ which we can write as $not(or(Q4, and(not(Q7), Q12))) \rightarrow IV$ is very good. In fact, we will now turn our attention to these three questions and, for a moment, we will forget about the US Presidential Election of 1880 which was won by a very marginal difference.

With three binary features, we could potentially observe a maximum of $2^3 = 8$ different binary patterns on these features. Interestingly, all these have been observed at least once in the data. Three have now been associated with an Incumbent party victory and the other five with a Challenger party victory. We can then associate each of these patterns to a vertex of a cube of edge length equal to one in three dimensions. For instance, the origin of coordinates $(0, 0, 0)$ would correspond to the pattern $Q4 = 0, Q7 = 0, Q12 = 0$ (an Incumbent Victory), $Q4 = 0, Q7 = 1, Q12 = 0$ (another Incumbent victory) and the pattern $Q4 = 0, Q7 = 0, Q12 = 1$ corresponding to a Challenger victory. Given this geometric interpretation of our patterns: *Would it be possible to separate the two types of outcomes in a different mathematical way?*

We first note that an arithmetic function of the form:

$$f(Q4, Q7, Q12) = 1 + 2Q7 - 2Q12 - 4Q4 \tag{1.13}$$

takes positive values for patterns that corresponds to the Incumbent class and for the Challenger class takes negative values. In fact, it is greater or equal than 1 for the first group and lower or equal to -1 for the other (see Table 1.7 for all the values). The set of points in three dimensions for which the function evaluates to 0 defines a two dimensional plane. In n-dimensions, if we have points that belong to two different classes and there is an $(n - 1)$-dimensional hyperplane that discriminates between the two classes we thus say the classes are "linearly separable".

Table 1.7 With the exception of the election of 1880, all the US Presidential Elections are linearly separable using only three of the 12 original questions proposed by Lichtman and Keilis-Borok in [140]

Q4	Q7	Q12	Class	$f(Q4, Q7, Q12)$
0	0	0	Incumbent	1
0	1	1	Incumbent	1
0	1	0	Incumbent	3
1	1	0	Challenger	−1
1	0	0	Challenger	−3
0	0	1	Challenger	−1
1	0	1	Challenger	−5
1	1	1	Challenger	−3

The separating function values are those computed following Eq. (1.13)

1.6.1 *Linear Separability and the Identification of "Maximum Margins"*

"Linear separability" is one central computational problem. Wouldn't it be great if everything is linearly separable? A simple function would discriminate samples in two groups. You can hire a data analytics service anywhere in the world, send your

data and a linear function would be sent to you (and it would be even better if the service is fast). All what you would need is then, when the function arrives in your email Inbox, to compute it for all samples (as we did in Table 1.7). This has the same characteristics as other NP problems, if somebody gives you a solution, you can check it is a solution in polynomial-time in the size of the instance. However is "Linear separability" an NP-complete problem?

Well, we have good news here. In fact "Linear separability" is a special case of a problem known as "Linear Programming" which is known to be polynomial-time solvable. However, the first algorithms for "Linear separability" goes back to 1958, with the pioneering "perceptron algorithm" of Rosenblatt [190]. These two are alternative methods to show that a problem is separable. When Rosenblatt presented the perceptron, the New York Times got attracted to the news and the headline was: "New Navy Device Learns by Doing" [216]. The New Yorker titled its coverage of the news with an interesting title: "Rival"[23] and with this simple algorithm humans now recognized that non-human entities were able to perform classification and categorization tasks. Being afraid of the performance of machines is clearly not a new trend of this millennium. Currently, perceptron-like algorithms are a known part of an area called "on-line learning" [32, 92] which is a distinguishable approach to "batch learning" [117, 145].

1.6.2 Slabs, Hyperslabs and Learning the Margins

When we can linearly separate a set of points belonging to two classes, this leaves us with a very interesting question. If we expect that our samples are coming from some process that generates them according to some distribution, we may have several ways by which we can linearly separate them. A perceptron-type algorithm or a variant gives a function that separates the samples (such as those of Fig. 1.13a or b which shows lines that separate the red and blue samples in two groups). Which one would we prefer?

Without entering in statistical arguments about which type of process may have generated samples with a particular observed distribution of occurrences, we are certain that there are well-defined regions in which the red and blue samples have been observed. This said, with no further mathematical arguments to bring to the table and just with a pure intuition, we may expect that a new sample (one which we have not yet observed and which may be either be red or blue in type) should "most likely" be observed somewhere in the smallest convex set of points that contains all the points of samples of that colour. In Fig. 1.13c we show two different sets of points and their enclosing *convex hulls*, the two polygons that define the limits of these two regions of points belonging to only one class. Obviously this generalizes

[23]http://www.newyorker.com/magazine/1958/12/06/rival-2.

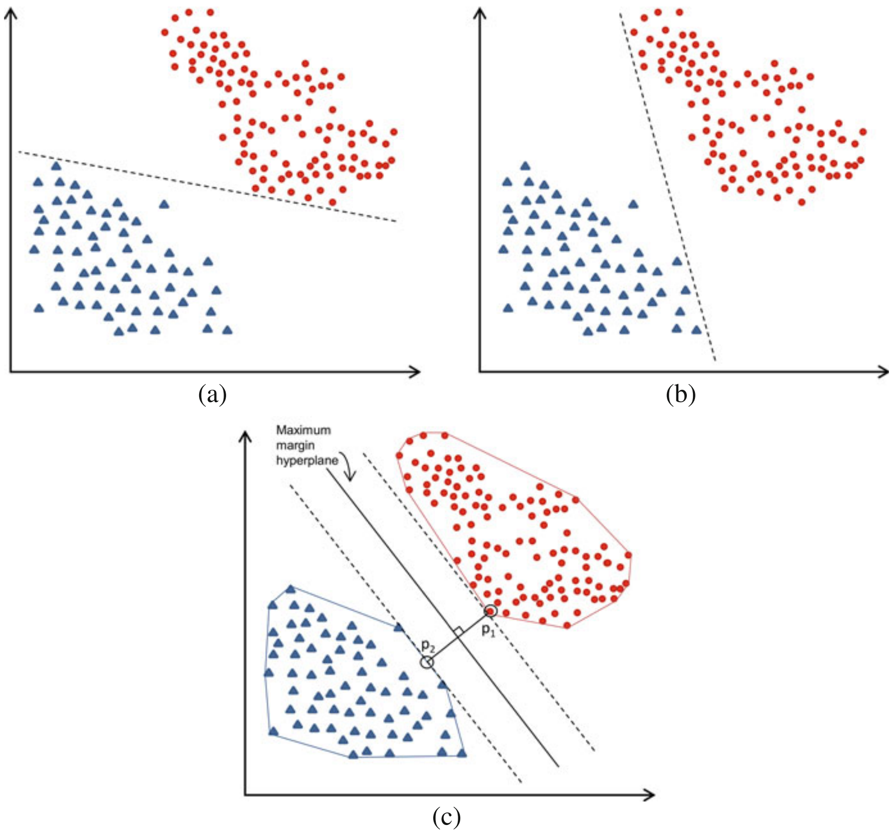


Fig. 1.13 If a set of samples is linearly separable, then there might be other criteria to select which linear function we would like to prefer. In many dimensions, selecting the *maximum margin hyperplane* is a reasonable alternative. The figure shows such a situation when the points are in the plane. (**a**) A possible separating plane. (**b**) Another possible separating plane. (**c**) Maximum margin hyperplane

to n-dimensions, in that case we will have *"facets"* that limit the regions instead of edges/segments.

In Fig. 1.13c we have identified as p_1 and p_2 the two points of each convex hull that are the closest to each other. One of them (p_2) lies on the edge of the convex hull that connects two blue samples in the convex hull. The other one (p_1) is actually one of the red samples, so it is a vertex of the "red" convex hull. For any unseen point on the line that connects p_1 and p_2, if is closer to p_1 we will intuitively associate it with the red group (and the opposite can be said for those which are closer to p_2). Consequently, any hypothetical new observed point that lies in the line which is perpendicular to the mid-point of the edge connecting p_1 and p_2 we could either attribute a red or blue label. In n-dimensions, we do not have a line, we have a separating hyperplane. Two parallel hyperplanes can then be traced and containing the points p_1 and p_2. The resulting hyperplane is the one with a *maximum margin* [24].

Maximum margins could be useful for decision-making. For instance, assume that the blue dots in Fig. 1.13 correspond to customers of a bank that manage to pay a home mortgage in less than 5 years. In contrast, let's assume that the red is a group of customers that could not pay the mortgage. The "slab" that contains the hyperplane of maximum margin separates the two-dimensional space in three regions. The one that contains the blue dots is then assumed to be a safer option for a bank, while the region that contains the red dots involves a higher risk. The slab may then indicate a region where the bank may actually take the risk of financing the customers and, in addition, it may contain the most profitable segment for them (as the institution may expect that they take more time to pay the mortgage than those in the blue group). Depending on the bank risk appetite, more or less funds could be allocated to customers that appear in the slab, thus diversifying the credit given could lead to a good strategy for the company.

1.6.3 What to Do When Linear Separability Is Not Possible? Escape to Higher Dimensions!

Many classification problems are defined as having a set of samples in an n-dimensional space that does not allow to have a separating hyperplane that separates the set of samples in two classes. What to do in these cases?

As a matter of fact, one simple idea is "a trick of escapology". If we have a problem that is not linearly separable in n-dimensions, perhaps it is linearly separable in m-dimensions when $m > n$. Can it be possible to add at least one extra dimension to the problem and make it linearly separable? We can resort to two basic alternatives: either we collect more information from the problem domain (and more piece of data could be added) or we do some sort of "trick" that creates these new dimensions from the existing data. In fact, this second approach has been very popular and a simple example can help understand the main concepts.

Consider the set of points in Fig. 1.14a. It is clear that there is no straight line that can separate these points in the two classes (red and blue). However, the red points are all "circumscribed" to be interior to an ellipse. This is quite revealing, and allows us to give a very simple example of the "escapology trick". It is known that an ellipse on a two-dimensional Cartesian Plane can be defined as the set of points $P = \{(x, y)\}$ that satisfy the following equation:

$$ax^2 + bxy + cy^2 + dx + ey + f = 0 \tag{1.14}$$

under the condition that $b^2 - 4ac < 0$, where a, b, c, d, e and f are all real valued coefficients.

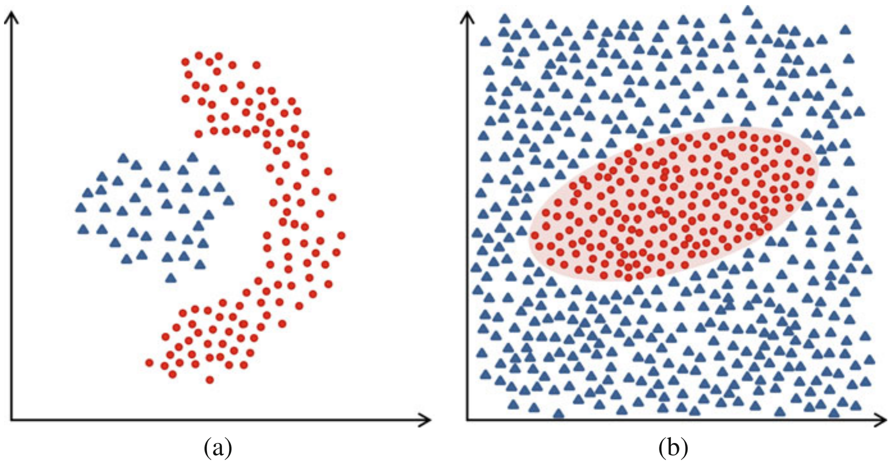


Fig. 1.14 Two instances in which the two classes of interest are not linearly separable in two dimensions. (**a**) Two classes of samples (red and blue) which are not linearly separable data by a line in a two-dimensional Euclidean plane. (**b**) Another set of samples which are not linearly separable data but in which all samples of one type are within the interior area of an ellipse

Just knowing this bit of mathematics has suddenly become quite empowering. What this means is that for the set of red samples $S_R = \{(x_i, y_i)\}$ it is always the case that:

$$ax_i^2 + bx_i y_i + cy_i^2 + dx_i + ey_i + f < 0 \tag{1.15}$$

and for all blue samples $S_B = \{(x_j, y_j)\}$ it is also true that

$$ax_j^2 + bx_j y_j + cy_j^2 + dx_j + ey_j + f > 0 \tag{1.16}$$

We can obviously draw a parallel with Eq. (1.11). We would not say that the set of blue and red points are "ellipse separable"... but indeed an ellipse is a kind of geometrical figure called a "quadric", also known as "quadric hypersurfaces" in higher dimensions. This is a powerful set of "new tools" to separate samples. Just in three dimensions there are 10 non-degenerate and 6 degenerate real quadric surfaces. They add quite a new battery of separating surfaces to our set of mathematical tools (for a visual tour of them check[24]).

Interestingly, degenerate forms of quadrics include planes, so we have something here that nicely generalizes on our previously known abilities to discriminate samples with mathematical equations.

[24]https://en.wikipedia.org/wiki/Quadric.

This new power comes at a price, the trick of escapology has a cost. If we have been given a problem in two dimensions, like the ones of Fig. 1.14 , then we now need to "create" new "meta-features" from our data corresponding to the values that our samples would have if three new features corresponding would have been measured. This means that for any sample $\mathbf{x}$ in our set of samples S (composed of the union of the blue and red samples in two-dimensions), we now have samples in a five-dimensional space $\mathbf{x}' = (x_i, y_i, x_i^2, y_i^2, x_i y_i)$ (note that we are writing the coordinates of $\mathbf{x}'$ as a function of the coordinates of its associated point in two dimensions $\mathbf{x}$). As an illustrative example, if we have the sample point $(-1, 5)$, then in this five-dimensional space it will be represented as $(-1, 5, 1, 25, -5)$. Then the problem of finding "a good quadric" that separates the set of points becomes the problem of identifying a separating hyperplane in five-dimensions. On a practical side, the mathematical method that finds a maximum separating hyperplane in this new higher-dimensional space is the same. No need for a new project in software coding, just a bit to "lift" some of our data points with this non-linear transformation. Isn't it wonderful?

1.6.4 A First Few Notes on Generalization

And now, a caveat, remember that old quote: "With great power comes great responsibility".[25] Assume that for a given dataset of 20 red and 20 blue samples in two dimensions you cannot find a separation in five dimensions. For instance, when you draw the convex hull of the red and blue samples they intersect, so there is no plane of maximum margin (and obviously the perceptron learning algorithm does not converge [62]). Then you may be tempted to continue the process and "lift it more" by using your newly discovered power again and by creating new dimensions. After all, such a separation may now be possible. For instance, you may now have your points lying in a nine-dimensional Euclidean space $\mathbf{x}'' = (x_i, y_i, x_i y_i, x_i^2, y_i^2, x_i^3, x_i^2 y_i, x_i y_i^2, y_i^3)$. While the problem of separating your 40 samples may become easier in this space, the use of the resulting separating maximum margin classifier may become less and less useful in terms of its *generalization* capabilities.

As we have seen in the case of the presidents dataset, three features were given the possibility of creating a linear separable function that almost perfectly classified all the US elections results with the exception of one. Then, there is a clear trade-off between reducing the dimensionality of the original dataset (without losing the discrimination potential on the observed), and increasing the discriminative capacity of the model built for the future elections. After all, it is generally attributed to William of Ockham (c. 1287–1347) the principle that says

Among competing hypotheses, the one with the fewest assumptions should be selected.

[25] http://quoteinvestigator.com/2015/07/23/great-power/.

This principle is generally known as "Occam's razor" and it then has to be considered as heuristic design rule to develop theoretical models of phenomena and not as a kind of a "judge" between alternative models. This said, and we will return to Occam's razor later, if we expand to higher dimensions we may pay a price in terms of the generalization to cases not yet seen (and, intuitively, the problem will be worse if we use too many dimensions). In fact, Table 1.6 has eight rows, indicating that all possible assignments/answers of "True" and "False" to these three questions have already occurred until 1980 (informally speaking, we "have covered all corners with elections", sometimes several times). There is then some strong evidence, with only one exception, of the power of these three questions to discriminate. We also know from [161] that neither a set of three nor a set of four features can perfectly separate the two types of outcomes. We know that only a 5-feature set can do the job. However, having a representation in five dimensions, to "cover all corners/possibilities" we should have at least $2^5 = 32$ different election patterns (and this is impossible to achieve as we have only 31 samples in our original dataset until 1980). In addition, we also know from [161] that there are 23 5-feature sets and there are 9 (α, β)-k-feature sets with $\alpha = 2$ and $\beta = 2$. It is then an open problem to identify which are the conditions that would make one of these many discriminating feature sets the "best" in terms of its expected generalization capabilities. For the US presidents dataset, with a new sample generated every 4 years, it is unlikely we could answer the question anytime soon!

This issue of generalization is closely linked with the use of a classifier. In our experience, we have found that there is a good synergy between using the (α, β)-k-feature set approach (to reduce the dimensionality of the dataset, something that can also be done with heuristics means [187]) followed by relatively simple classification methods (like obtaining a maximum margin hyperplane or by employing a Support Vector Machine algorithm [79]). It may be possible that, in the near future, a combined multi-objective optimization approach combining these approaches would then be developed.

1.6.5 Piling Things Up: From Simple Neural Networks to Deep Learning

A lot can be done with linear discrimination and proper transformation of features. Quoting from the Epilogue of [9]

> The linear model can be used as a building block for other popular techniques. A cascade of linear models, mostly with soft thresholds, creates a neural network. A robust algorithm for linear models, based on quadratic programming, creates support vector machines. An efficient approach to non-linear transformation in support vector machines creates kernel methods. A combination of different models in a principled way creates boosting and ensemble learning. There are other successful models and techniques, and more to come for sure.

Indeed, “piling things up”, using some of the techniques as building blocks of something more complex would add power to our machine learning techniques. Generally referred as *connexionism*, this approach has been followed by many researchers and practitioners since the 1940s but it may come at a cost. Minsky and Papert’s book on perceptrons (published in 1969 [158]) was credited at one that pointed at some limitations of linear separability and some gains that can be obtained by using layered networks. The book also pointed at limitations of these networks; these results created pessimism but also a dynamic response to overcome the obstacles that emerged [168].

We will now return to our guiding example, the one of the 12 questions for the US Presidential elections. We can now go back to a geometrical view. A training sample can be thought as a corner of a 12-dimensional hypercube. Assuming that no two elections have been equal, each corner will be “labelled” as IV (if the Incumbent party won that particular election) or CV (when the Challenger prevailed). We have asked the reader in Sect. 1.6 to investigate if there is a separating hyperplane that can separate these corners in two groups (such that those having the same label are in the same half-space only).

We can now extend our previous challenge to the reader and to consider the two questions:

Does *a single* hyperplane exist that separates the “IV corners” from the “CV corners”?

a reiteration of linear separability question, if the answer is “No”, then the reader should now consider this new question:

Do *two hyperplanes* exist that can partition the corners so that either one quadrant contains all and only “IV corners” or one quadrant contains only “CV corners”?

Why is this relevant? Assume that the answer to the first question is “No”, we cannot find linear separability. Suppose we have now the following “architecture” which we have built by “piling up” our basic “building blocks” based on linear separable functions. On the bottom, we have one vertex for each question input, which like in Sect. 1.6, can we assume “receives” or “holds” the value of the variables $\{q_1, q_2, \ldots, q_{12}\}$. Let us also assume that there is a weight associated with each arc. We will then use the following notation, let $w_1^{(1)}, w_2^{(1)}, \ldots, w_{12}^{(1)}$ be the weights of the arcs coming from the inputs such that $w_i^{(1)}$ “weights in” the input q_i for vertex H_1. Assume that we want our new “system” to be such that when we have an Incumbent victory sample the vertex at the top (H_3) should return the value +1 for any “positive sample” (which without loss of generality can say they are the Incumbent victories), and H_3 should return -1 for a sample that corresponds to a Challenger Victory. This could be done, in principle, if H_3 is some sort of “logical AND function” (returns 1 if and only if it receives an input which is 1 coming from both H_1 and H_2. If we call h_1 and h_2 the inputs for H_3 it is easy to see that if $h_3 = H_3(h_1, h_2) = h_1 + h_2 - 1$, $H_3(-1, 1) = H_3(1, -1) = -1$ and $H_3(-1, -1) = -3$, while $H_3 = (1, 1) = 1$, which is not surprising as a Boolean AND is a linear separable function.

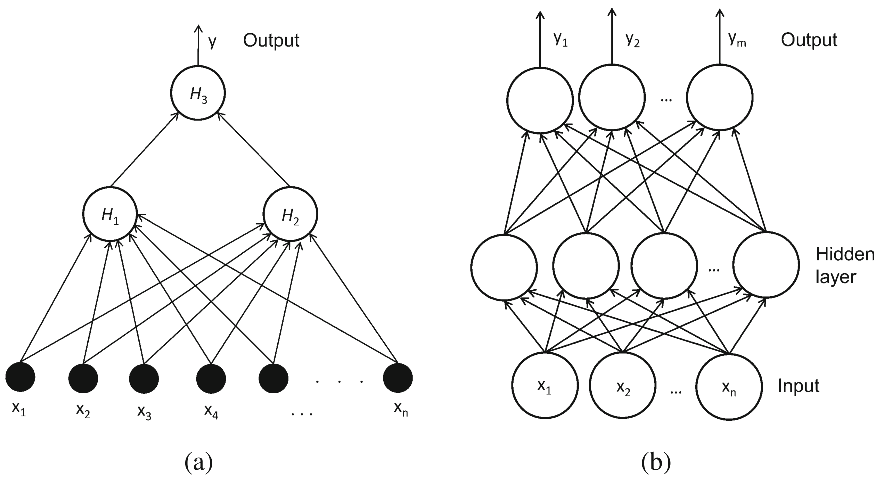


Fig. 1.15 Two basic architectures for artificial neural networks. (**a**) A 3-node neural network with n-inputs. (**b**) A two-layered neural network with four nodes in the hidden layer shape

This said, since we know that the problem is not linear separable, without loss of generality we can assume that H_1 is a linear discriminating function that returns the value 1 for many samples of the IV type but makes some "mistakes", returning 1 for some samples of the CV type. If H_2 is one function that returns -1 on those CV samples for which H_1 has returned the undesired output but returns 1 for all IV-type samples, then H_1 and H_2 together define the intersection of two half-spaces (a "quadrant") that contains samples of only class IV. This was our original question to the reader. Consequently, this pyramidal structure can separate the samples in two classes if and only if we can partition the corners with samples of different labels with *two* hyperplanes so that one quadrant contains all and only "IV corners" (or, alternatively, if one quadrant contains only "CV corners").

This proposed architecture to "pile up" our basic classifiers H_1, H_2 and H_3 (shown in Fig. 1.15) is what it is known as a 3-node artificial neural network. We can think of three units that "compute" according to

$$h_1 = sign\left(b_1 + \sum_{i=1}^{n} w_i^{(1)} x_i\right) \tag{1.17}$$

$$h_2 = sign\left(b_2 + \sum_{i=1}^{n} w_i^{(2)} x_i\right) \tag{1.18}$$

$$h_3 = sign\left(-1 + h_1 + h_2\right) \tag{1.19}$$

Then the "learning problem" for the US presidents data corresponds to answer if it is possible to find two sets of values of $\{w_1^{(1)}, w_2^{(1)}, \ldots, w_{12}^{(1)}\}$ and

$\{w_1^{(2)}, w_2^{(2)}, \ldots, w_{12}^{(2)}\}$, as well as two constants b_1 and b_2, that, together, can make it possible to have all the examples of the IV-type in the same quadrant defined by the linear discriminant functions hyperplanes h_1, h_2 and h_3 (as shown in Eqs. (1.17)–(1.18)).

What happens with the computational complexity of this problem? It has been attributed to Megiddo the honour to have been the first who has shown that this problem is NP-complete in a quite general case [156]. Meggido's result is for an *arbitrary* collection of points in n-dimensional Euclidean space. The problem when the set of points are in the vertices of a hypercube is a particular case thus to prove its computational complexity it would demand a different proof. This means that another mathematical proof is necessary (a particular case of an NP-complete problem may indeed be solvable in polynomial-time, that is, with an efficient algorithm like the perceptron algorithm, or others based on linear programming). Unfortunately the problem is also NP-complete...Blum and Rivest proved that *"training a 3-node neural network is NP-complete"* (see [25] and [26]). Other models of neural networks also lead to NP-completeness results [102, 104], while other models and architectures had led to efficient algorithms [103–105].

The idea of "piling things up" leads to a plethora of different computational solutions for classification problems. Kunihiko Fukushima in [73] reviews several such contributions before 1980 in an article in which he proposes an architecture called the "Neocognitron", for the purpose of delivering the right classification of images without being affected by shifts in the position of the object of interest [72, 74]. The Neocognitron approach[26] is considered a pioneer proposal in the area and, for many years, its full potential was anticipated but it was not entirely revealed. Several decades later, thanks to some new heuristics for learning and a new types of hardware (in particular the GPUs [236], an acronym for "Graphic Processing Units"), some of the ideas of pioneers from the 1980s have now become a disruptive reality[27] in every aspect of our lives. Known as "Deep Learning" this area has been recently reviewed by several authors [133, 197, 198, 227].

Here is perhaps an important lesson for our practitioners. Although all these NP-completeness results exist for learning and classification, many of these problems can be addressed with heuristics in practice. An important caveat exists, these methods currently are exhibiting a fantastic performance in many problems but at the cost of requiring a large number of examples. Undoubtedly, the fields of applications of finance [89], sharing behaviour social media [97] and even the US Central Intelligence Agency is reportedly using these technologies to predict collective human behaviour and social unrest [193]. Machine learning and artificial intelligence methods are showing that connexionism is alive and one of the strongest tools for many large-scale problems in business and consumer analytics. This time, it seems they are here to stay since the field is now so vibrant that it is pushing the development of specialized hardware for it.

[26]http://www.scholarpedia.org/article/Neocognitron.

[27]http://fortune.com/ai-artificial-intelligence-deep-machine-learning/.

1.6.6 A Few Words Regarding Model Complexity

One of the most common mistakes in real-life situations involving machine learning or the use of artificial intelligence solutions is the "unrecognized preference" towards "complex models" that fit the data better. There has been some "obsession" with the performance of a method on the known data, but increasingly performing well on it would not necessarily bring equal improvements for data not yet seen. The *generalization* capacity of a complex model would be then smaller than the one of a simpler model.

Without entering the discussion of how to measure "complexity", which is an open subject with many mathematical innuendos and ramifications,[28] we choose to illustrate our discussion with an example modified from Fig. 3.7 in [9], which we have slightly adapted for the purpose of our discussion. Let us suppose that we have collected information on two continuous feature variables of a set of samples/points in the Euclidean two-dimensional plane defined by these two features. We have been given class labels (red and blue) for all of them. We have drawn them with circles and Triangles, respectively, in Fig. 1.16a depending to the class they belong to. It is perhaps intuitive to the reader that these two sets of points are not linearly separable and that the line shown is perhaps one that minimizes the error you can have in this training set (or at least, one that does a reasonable good job at separating the points). In Fig. 1.16b we show that a higher dimensional surface can be calculated by the application of non-linear transformations to the original features (you can follow the discussion of in [9] to see how it can be calculated). The process is basically the "escape to higher dimensions" trick we have discussed before in Sect. 1.6.3. The intersection of this surface with the two-dimensional plane acts as a discriminant function on the set of red and blue nodes and achieves the separation.

What is then the catch? Of course we have been able to separate the two sets of points by going into higher dimensions, but do we have enough samples to expect a good generalization performance? One "rule of thumb" is that the number of samples that you would need to achieve good generalization should be proportional to the so-called *Vapnik–Chervonenkis dimension* (VC-dimension) [143]. The VC-dimension for a linear discriminant function on n-dimensions (a perceptron) is $n + 1$ [9, 61], so the VC-dimension for the first case is 3. In contrast, if we aim to use a higher-order surface, such as that of Fig. 1.16b, we would need to have at least an order of magnitude more samples to have some sort of reliability that the discrimination using this surface is better than the discrimination using a separating line [9]. We have seen a similar situation before, when we discussed that three questions can be used to predict the presidency of the USA (making a single mistake and "covering" all possible patterns you can have with three features).

Would this really be the *"3 Keys to the White House plus a linear classifier solution"* to the US Presidential prediction problem? Well, the VC-dimension for

[28] http://bactra.org/notebooks/complexity-measures.html.

the 5-dimensional case would be as high as 20, and for the 12-dimensional case of using all questions it is 12*15/2=90, indicating we would need to be collecting data for at least 200 years to reach the required values that gives us some confidence. Since there was no linear separability with 12 questions in the existing elections until 1980, we may possibly consider that we need to have other types of models indeed, while the linear one is still a good approximation.

We now turn our attention to Fig. 1.16b. Contrary to Fig. 3.7b of [9], we have decided not to colour a region discriminated by the higher-order function with the colour red on the background (and we have left it as white instead). The reason is that we would like to highlight an interesting fact. The reader would notice that the red points can be covered by two relatively thin *"slabs"*. Again, according to a generic role of preferring "simplicity" to "complexity" we could argue that there is a *piecewise linear model* that "explains" the location of the red samples in the plane. It does seem that not all the space associated with "not blue" could be "naturally" associated with be "red" given the evidence that all the samples of class red are restricted to be within two covering slabs.

In n-dimensions we can talk about *hyperslabs*. If we assume that we have to find two hyperslabs in our dataset that would correspond to find sets of coefficients which are the solutions of

$$b_1 - \epsilon_1 \geq \sum_{i=1}^{n} w_i^{(1)} x_i \geq b_1 + \epsilon_1 \qquad \forall x \in Slab_1 \tag{1.20}$$

$$b_2 - \epsilon_2 \geq \sum_{i=1}^{n} w_i^{(2)} x_i \geq b_2 + \epsilon_2 \qquad \forall x \in Slab_2 \tag{1.21}$$

given some tolerances $\epsilon_1 > 0$ and $\epsilon_2 > 0$ which correspond to half the length of the width of these hyperslabs (and we are assuming that in principle they are different).

The problem of finding the minimum number of hyperslabs that cover a set of points has been called the MINIMUM PARTITION INTO CONSISTENT SUBSYSTEMS (MIN PCS) by Mattavelli and Amaldi in [154] and in that work the authors present a greedy approach for this combinatorial problem. The strategy is based on "breaking" an instance of the MIN PCS into smaller problems which require the solution of another problem (the identification of consistent subsystems with the maximum number of equations [16]). We refer the reader to the expanded discussions of the topic in [17].

We note that in MIN PCS the objective is to cover the samples with the minimum number of hyperslabs, and once a point is covered we do not take into account the distance between the point and the central hyperplane of the hyperslab. There are similarities with *support vector regression* [59] which has business analytics applications in the stock market [139] and short-term load forecasting [94]. Other applications have been reported in customer demand forecasting [136], tourism demand forecasting [34], response modelling and direct marketing [113] and more recently in sales forecasting of computer products [146]. Even more recently, a very interesting new integrated problem has been proposed called k-PIECEWISE AFFINE MODEL FITTING WITH PIECEWISE LINEAR SEPARABILITY PROBLEM [15]. In many cases, there is the need to have a feature selection step together with the support vector regression approach which then more complex models like the one in [15] would perhaps deal with in a formal way. In any case, there is clear potential for these approaches in interpreting data.

Why we make these comments? Why we are concerned with linear regression in a world that is known to be highly non-linear? Well, we have actually observed in our practice a situation exactly like the one in Fig. 1.16 when analysing a high-dimensional dataset. It took us a while to recognize that indeed, some pairs of variables in our domain of study were easily separating the two classes of interest

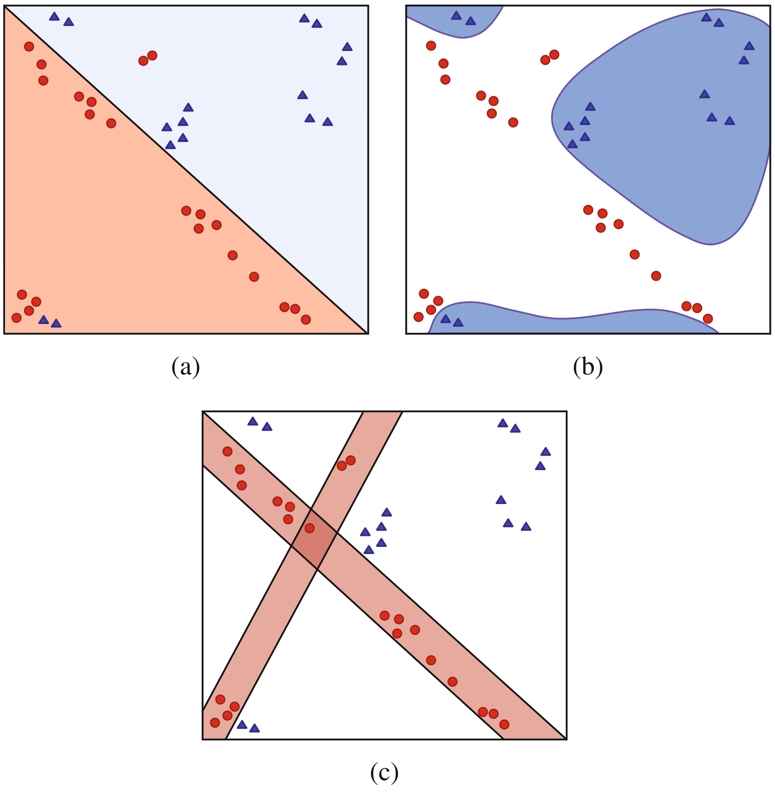


Fig. 1.16 If a set of samples is *not* linearly separable in a certain space of n-dimensions, then there might be other criteria to select which linear function we would like to prefer. Again, selecting the *maximum margin hyperplane* in a higher n'-dimensional space continues to be alternative. *Piecewise linear model estimation* of samples of one or both classes is another possibility to consider to separate the groups. (**a**) A possible separating plane that makes four "mistakes". (**b**) The intersection of a higher-dimensional surface that perfectly separates the points in two groups as suggested in [9]. (**c**) Piecewise linear model estimation of red samples

in one for which a k-piecewise affine model (with $k \leq 2$) was very good at "explaining" the variables of one class (but there was no equivalent "fitting" via k-piecewise affine models in the other class). Then we recognized that a large number of pairs of variables had the same behaviour. The domain in question was that of bioinformatics, and the problem was related to the disease called multiple sclerosis. The class that was behaving as the class with a "low k" good fitting was that of healthy control samples, while the samples from disease patients were not having that behaviour. In fact, multiple sclerosis has at least four clinical subtypes and we were sure that our samples belong to at least three of them.

This said, we draw some conclusions: (a) in some problem domains, some variables may be "tightly regulated" via some unknown mechanism that is "governed" by a linear model, or a piecewise linear one, (b) under some assumptions (i.e., natural evolution in biology, perhaps in this case; or by engineered design in others), a system exhibits this behaviour, (c) while other variables may be having good linear models that explain them, as this is in a disease or disease subtype, they may not necessarily be the same as the one that are in the other class.

1.7 From Features to Patterns

We have mentioned the word "patterns" a few times in this chapter but, so far, most of the attention was related to "features" and the role they play in discrimination. We discussed the problem of mining association rules, which can be seen as finding rules involving features which are in particular states, thus naturally leading to a type of pattern. Also, the presentation of the concept of k-feature sets in Sect. 1.3.2 rather explicitly refers to the concept of patterns.

We can argue that "patterns" are what we observe; we are in some sense "hardwired" to try to identify patterns and then act accordingly. Coalescing these patterns into some sort of "higher-order" set of patterns may help to establish a reduced set of rules for actions. This section explores these new possibilities.

1.7.1 On Including Problem Domain and Tall Men: Feature Engineering

In general, given an analytic task, it is difficult, if not entirely futile, to give generic arguments about what is the most relevant set of features. In some sense, it is only after a subset of discriminative features has been identified, and a mathematical model or a classifier algorithm based on them has been created, that we could recognize the usefulness of such representation. Prediction and interpretability are key.

In the highly successful TV show called "House of Cards", we can hear the central character Frank Underwood stating: "Tall men make great presidents". Abraham Lincoln was 6 foot 4 inches (193 cm), Barack Obama is 6 foot 1 inch (185 cm). The height of elected US presidents in our dataset has a growing trend in the past 150 years (as perhaps the average height of a US citizen has been increasing as well). Is this a form of problem domain knowledge that can be useful for prediction?

There is an entrenched belief that in modern US elections taller candidates have "the upper hand" (pun intended) and that they are more successful than their opponents. Is this a trend in leadership roles? We read in *"Blink: The Power of Thinking Without Thinking"*[76]

> In the U.S. population, about 14.5% of all men are six feet or over. Among CEOs of Fortune 500 companies, that number is 58%. Even more strikingly, in the general American population, 3.9% of adult men are 6 feet 2 inches or taller. Among my CEO sample, 30% were 6 feet 2 inches or taller.

Is that it? Are we selecting presidents by "Thinking Without Thinking"? Also from Gladwell's book [76] we quote:

> Of the tens of millions of American men below 5 foot 6 inches, a grand total of ten in my sample have reached the level of CEO, which says that being short is probably as much, or more, of a handicap to corporate success as being a woman or an African-American.

Departing from this special case, and from a data science perspective, we should concentrate in the core concept: are we missing essential features that can even reduce the dimensionality of models that can predict the data even more? After all, if this is a critical feature, it may "interact" well with other features and give predictive models with less variables.

We can generalize this a little. Given a problem domain area of interest, the first attempt should always be to identify which traits in the data can be measured with confidence. This mapping from the "qualitative characteristics" to the "quantitative features" to be measured is a decisive step. Are the 12 questions from Keilis-Borok relevant to understand the election process? If we consider a binary feature as a test on a sample, what are its individual *sensitivity* and *specificity*? Is it robust to the type of "noises" we can have at data collection? Are the features covering all aspects of the problem of predicting the US presidency election result? Should more be added? For instance, one of them could be: Q_h: "*Is the Incumbent party candidate taller than the Challenger candidate?*" That is clearly a quantitative feature with no margin for error and perhaps preferable to others more subjective (like the question involving "charisma"). Observed patterns may inform which features to include.

Researchers and data scientists use the denomination of *"feature engineering"* in machine learning for this process of using domain knowledge to create new features. This is a very important step in the extraction of knowledge from databases. For an interesting example of the craftsmanship behind current feature engineering techniques we refer to [138] in which the authors address one of the KDD Cup 2013 challenges of which they have been the winning team. The task was to identify whether a paper is truly written by a particular author. The authors transform different types of text information into 97 features and they also use ensemble learning techniques. In another study,[29] the authors address the problem of predicting student algebraic problem performance when past performance and feature engineering also plays a central role in their success.[30]

This area is more art than science as there are still no definite and generic guidelines about how to identify a good set of features. In reality, the process of identifying what are the good features involve several steps: (a) ideally, they should be easy to compute, (b) they should have a reduced cardinality of possible outcomes, (c) ideally, the outcome of computing the feature states should, if possible, correlate with the target feature of interest (the outcome we want to explain), (d) they may weakly correlate with each other, showing some degree of mutual independence in our sample dataset, (e) they should help to understand the properties of the discrimination task you aim to sort out, (f) they should lead to some sort of useful intervention (assuming the knowledge generated would be followed by some actions from our part).

The list above may not be complete. In some cases, the "right" set of features can only be identified after some initial experimentation with a dataset. In our practice we have observed that, even if a set of features satisfies our requirements (a–f) detailed in the previous paragraph, it is often the *combination* of pairs of features that brings more information for building classification models. We thus generally refer to them as *metafeatures* [51]. One simple way by which we have captured the idea of introducing combination (for instance, when we are dealing with numerical values) is to compute all four simple arithmetical operations (difference, addition, multiplication, and when possible, division). This takes the original number of variables from n to $4n(n-1)2 + n$ variables (assuming, of course, that ratios of variables make sense in the context of the problem domain, so we are here including division into account). We have noticed that in several applications, we have benefited from using "meta-features" [19, 51, 101, 188]. These metafeatures are somehow analogous to the outputs of the first hidden layer of a "connexionist machine" that has a constant weight of 1 for each edge that connects an input with a node in the hidden layer, and has $4n(n-1)/2 + n$ nodes in the hidden layer (where n of them just replicate the input in the hidden layer). In addition, they may be the simplest of the operations with variables in a tree that represents solutions in a genetic programming approach.

The creation of new features from the existing ones can be seen as a natural extensions of the approaches described in Sect. 1.6.3 for problems in Euclidean spaces. We have seen before that these approaches help to extend the power of Support vector machines while benefiting of efficient (polynomial-time) algorithms for finding a discriminating hyperplane of maximum margin. A central step is then how to identify a non-linear transformation that, given inputs which are vectors in an n-dimensional Euclidean space $\mathbf{x}$, is transformed into other non-linear combinations

[29]http://www.csie.ntu.edu.tw/~htlin/paper/doc/wskdd10cup.pdf.

[30]http://www.csie.ntu.edu.tw/~cjlin/courses/dmcase2010/slide.pdf.

via a function $\phi(\mathbf{x})$ such that the features for the SVM would be the similarities between the sample to be categorized and the training examples. A process in which a part of the training data is used to learn the appropriate function is needed, so this is a semi-automatic procedure that requires the identification of the so-called *kernel functions* [209]. Clearly, a machine-independent automation of these processes is pretty much in need for an area which is obviously called *automated feature engineering* or *feature synthesis* [109] and that has an association with *propositionalization* (the transformation of a relational dataset into a propositional dataset) [126].

1.7.2 Feature Engineering Should Help You to Understand

Now, before you decide to define the set of features, you need to consider what are you going to obtain when the analysis is completed. For once, we already know that increasing the dimensionality would probably lead to better fitting your training data (even at the risk of *overfitting* it), and if you have some sort dimensionality reduction technique you need to consider the *interpretability* of your final mathematical models and the classifier you have obtained. Would the set of features tell you something about the process you are trying to understand?

To wrap-up our this discussion about the US presidential election illustrative case, and to satisfy the curiosity of the readers, we indeed tested the hypothesis that the height of the candidates would be important for the outcome of the election. Using the data available online [229], and the Q_h presented before in this section, we tested if there was a k-feature set with $k \leq 4$ in this dataset. While recognizing that there is a clear trend for the answer to be positive between 1904 and 1980, and it could be considered a very correlated feature with the outcome, it does not allow to bring any new k-feature set solution with $k \leq 4$.

And if you asked or you are curious about 2016, Donald Trump is no Abraham Lincoln but he is pretty tall, 6 feet and 2 inches to be exact. In contrast, Hillary Clinton is the second shortest US presidential candidate in history at 5 feet 5 inches (165 cm, both James Madison and Stephen A. Douglas were 163 cm tall). The 23 cm of difference between the two candidates of 2016 was very clear during the second debate in which Trump loomed behind Clinton [55]. Is this a coincidence or a clever marketing tactic to play with the minds of the voters? Time will tell.

You may ask then, is it really important to introduce a particular feature when it is highly correlated with the outcome? Would it be able to bring new insights about the problem when it appears together with the other features in a solution? Would the patterns that are present help you to enact some intervention? These are questions that are highly dependent on the problem. Take a “return-on-investment” approach for feature engineering, sometimes working with less gives more.

1.7.3 Wine Connoisseurs and the Pattern Identification Problem

In the example given in Sect. 1.3.2 we have seen that the problem of feature selection is central for data analytics. Through this process of selection we are able to identify patterns which are present in one class and not in the other. These patterns have all the features in particular values. Is it possible to have other types of identification of patterns that give other insights?

This question is of particular interest for *"extreme case analysis"*. If we are studying consumer behaviour of a particular product, then some of these products may be highly rejected and others highly appreciated and we would like first to understand the motives for that. Consumers would pay significantly more for those products which, in turn, may become the ones in "high demand". Knowing the qualities that make them so required is then very important. That is where substantial financial gains may exist. Also, those highly rejected products (items that companies thought may really hit the market well but did not) may reflect some conceptual design issues that need to be revisited and addressed. We also note that the reasons for high acceptance or rejection may not necessarily be the same. Would or would not a feature set-based approach be able to bring some light here?

We have said before that a set of features can induce a set of patterns. We showed in Table 1.4 how a 5-feature set helps us to "compress" all the elections into a set of just 17 patterns that discriminate between the two types of outcomes. These patterns are somewhat "particular" in the sense that they correspond to fully specified set of feature values. This compression may not be enough for decision makers. If we are discussing a problem in the business and consumer analytics scenario, the marketing personnel may require an even higher degree of data compression. Marketers would prefer a smaller cardinality set or patterns so that they could take decisions from a more limited set of observations. This motivates the search for those. For that we note that now the objective is somewhat different. These "new type of patterns" are actually grouping samples.

Like we did before, we present this new combinatorial optimization problem by first pointing to an application. We could have done it with the US elections dataset, but we feel that a smaller "toy example" different from the one of the already can be illustrative here.

Let's consider this hypothetical situation. After the results of a large study with wine connoisseurs are finished, we concentrate the attention on four wines that have been highly acclaimed (all of them in the "award winning" class after a competition and three that have been "strongly rejected" by the specialists). Let us then assume we have a dataset of Table 1.8 in which we have information on five features: f_1 the free-sulphur dioxide, f_2 the level of alcohol content, f_3 is the sugar content, f_4 is the density, f_5 is the volatile acidity. We first note that this dataset can be an instance of the k-FEATURE SET Problem. This may be a surprise since in this case the dataset does not have binary values as entries. There is a reason in computer science for first presenting the problems with their "naked core". There is nothing really in the requirements definition of what a k-feature set is that says that the only variables that are allowed are binary, it actually requires recognizing if a feature is in the same value or not in from a finite set of computable checks. This set of different states a feature can have is called an "alphabet", so here we have a five characters alphabet, while in the original definition of feature set we have only either "one" or "zero" (coding for "True" or "False").

Table 1.8 A toy example of the problem of identifying patterns in data from a wine study

Sample	Free-sulphur dioxide	Alcohol	Sugar	Density	Acidity	Class
1	VH	H	L	M	VH	Highly rejected
2	L	H	VL	M	L	Highly rejected
3	VH	M	M	M	VH	Highly rejected
4	VL	H	VH	H	VH	Award winning
5	L	M	VH	M	L	Award winning
6	L	H	VH	H	VL	Award winning
7	VH	H	VH	M	L	Award winning

The rows corresponds to seven different samples of two types of quality (the "Class" label given by specialists which is of two types in this case). "VH" stands for "Very High", "H" for "High", "M" for "Medium", "L" for "Low" and "VL" for "Very Low". These values have been given to a set of five features that help to characterize these different wines

We thus refer to feature f_3, which corresponds to the amount of sugar in the wine. The award winning wines are all having a "VH" (for "Very High") sugar content amount. Then feature f_3 alone is a 1-feature set, making this instance a "Yes" instance for all $k \geq 1$. This shows a problem for modelling questions of interest with "just" a feature set perspective, as we know that a high sugary drink is not necessary a good wine…We need to "bring the context" and grouping the samples in a particular way. We need to do so by tightening the state of some features to values (when there is evidence for that) and find patterns that allow some "uncertainty" on the values that a feature can have. Towards building this new mathematical framework for this, we will need to include an extra character in our patterns, the "$*$" symbol (the "wild card" that can represent any symbol in our alphabet).

Now let's go to what it is a possible feasible solution, which we do not claim to be optimal, is the following:

$$Good = \{*\ H\ VH\ H\ *, \quad *\ *\ VH\ *\ L\} \tag{1.22}$$

$$Bad = \{VH\ *\ *\ M\ VH, \quad L\ \ H\ VL\ M\ L\} \tag{1.23}$$

which shows that the dataset of "award winning" samples can be covered by at least one of the two patterns in the set called "*Good*" (obviously, referring to "good" and observed patterns) and any "strongly rejected" sample can be covered by at least one of the patterns in the set called "*Bad*".

The purpose is to find some patterns that are common to only "award winning" or to only to a "highly rejected" wines. The patterns may not necessarily be related. Using these patterns we can build mathematical models; a model for understanding rejection would then not necessarily relate with a model for understanding high acceptance. Finally, a pattern corresponds to a "kind of test", or a different type of metafeature than the ones we described before. It will be of the type "if the pattern is observed then the class is X" (where X is one of the two classes of interest). In terms of logic thinking, it could not be possible for the pattern not to be present and the sample interrogated not to be of that class. The ultimate goal, for the marketers demanding analytics for the study, is to identify the minimum number of patterns such that, given any new sample of another wine, we can say if it will be "strongly rejected" or "award winning" as soon it passes one of the tests, and it is guaranteed that *for at least one test* the result must be true.

*1.7.4 Mathematical Formulation of the l-*PATTERN IDENTIFICATION *Problem*

We now give a more formal definition of the problem. A *string* is a concatenation of symbols of a given *alphabet*. Let Σ be one such alphabet. We define a *pattern* as a string s over an extended alphabet that now includes the "wild card" symbol and we write $\Sigma_* := \Sigma \cup \{*\}$.

In the example before we assumed the connoisseurs where giving marks for each feature in a 5-item Likert scale and we also have a "wild card" symbol to augment our alphabet Σ so $\Sigma_* = \{VL, L, M, H, VH, *\}$ (symbols corresponding to the answers $Very\,Low$, Low, $Medium$, $High$, $Very\,High$ (the levels the experts have given to each feature/attribute).

We denote $s[i]$ as the ith symbol of s. For instance, the first pattern in the set Bad before

$$p = VH\ *\ *\ M\ VH \tag{1.24}$$

specifies that feature $p[1]$ has value VH, $p[4]$ must be M and $p[5]$ must be VH. The symbol "$*$" is called *wild card* and can match any symbol in Σ. The wild cards for $p[2]$ and $p[3]$ indicate that "we don't care", or better, the pattern does not specify what the value at those two features should be.

A pattern is said to be *fully specified* when it has no wild cards, then the following nine fully specified patterns are compatible with $p' = VH\ *\ *\ *\ VH$

$$\begin{gathered}
VH\ H\ L\ M\ VH\\
VH\ M\ M\ L\ VH\\
VH\ H\ M\ L\ VH\\
VH\ L\ L\ L\ VH\\
VH\ M\ VL\ H\ VH\\
VH\ L\ L\ M\ VH\\
VH\ VH\ L\ L\ VH\\
VH\ M\ VH\ M\ VH\\
VH\ VL\ M\ M\ VH,
\end{gathered}$$

out of 125 fully specified patterns that are compatible with p'.

A pattern p is *compatible with a string* g, denoted $p \to g$, if for all i such that $p[i] \neq *$ we have $g[i] = p[i]$. Alternatively, a pattern is *not compatible* with the string g, and in that case we write $p \not\to g$.

For instance, in the example the pattern $*\ H\ VH\ H\ *$ is compatible with the strings *VL H VH H VH* and *L H VH H VL* corresponding to the samples 4 and 6 of the "award winning" wines. Pattern $*\ *\ VH\ *\ L$ is compatible with the strings *L M VH M L* and *VH H VH M L*, corresponding to the wines 5 and 7. The reader can easily check that these patterns are not compatible with any of the strings corresponding to the wines that have the other label.

We now extend this notation to sets of strings, writing $p \to Good$ to denote that for all strings g in the *Good* set of strings, $p \to g$ and $P \to Good$ if for all $g \in Good$ there exists at least one $p \in P$ such that $p \to g$.

A set P of patterns *Good-Bad-separates* an ordered pair $(Good, Bad)$ of two sets of strings, and we will write this as $P \to (Good, Bad)$, if satisfies two things: $P \to Good$, and for every $b \in Bad$ and $p \in P$ we have $p \not\to b$.

Thus we can state the central problem:

l-PATTERN IDENTIFICATION

Instance A finite alphabet Σ, two disjoint sets $Good, Bad \subseteq \Sigma^n$ of strings and an integer $l > 0$.

Question Is there a set P of patterns such that $|P| \leq l$ and $P \to (Good, Bad)$?

where $|P|$ is the cardinality of the set and Σ^n is the set of all strings that can be written using n characters from the alphabet Σ.

We note that we pose this problem as the task of separating "the good from the bad", one of the two groups of samples. We now turn into attention to the novelty of this different problem.

1.7.4.1 Are the l-PATTERN IDENTIFICATION and the k-FEATURE SET Problem Closely Related?

To answer this question, we first note that both problems have the same input, so they are already related in this way. But is there a deeper connection? They may be actually same problem. Let k-FS be the k-FEATURE SET problem, and l-PI the l-PATTERN IDENTIFICATION problem. For some readers it may be intuitive that they are different. We could understand l-PI as a problem of "covering" the fully specified samples with the minimum cardinality set of not compatible patterns (as rows/strings in the instance that are compatible with the pattern are grouped together). This seems already different than feature set but the differences need to be spelled out.

We assume that we have a valid input of the l-PATTERN IDENTIFICATION (this means that no pairs of samples are identical in all the features and belong to different classes). We first need to check if it is the case that, given such a valid input, it is possible to find k' and l' values for which the answer is "Yes" for both k'-FS and l'-PI. This is clearly the case, you just need to include all the features (for k'-FS being "Yes") and all the samples (for l'-PI being "Yes").

What is less trivial is to show that *each feature set induces a set of patterns and vice versa*. To make the notation simple, assume we have a k-feature set and we have reindexed the features so that the first k are a valid feature set. Let $F = \{1, \ldots, k\}$ be this feature set. The patterns made by the first k characters, followed by $(n-k)$ wild star characters "*", are a solution for the l-PATTERN IDENTIFICATION problem.

$$\begin{array}{ccc} L\,M \mid VH\,L\,VL & & L\,M * * * \\ M\,M \mid VH\,L\,H & \mapsto & M\,M * * * \\ - - - - - - - & & - - - - - - - \\ H\,M \mid VL\,L\,H & & H\,M * * * \\ M\,L \mid VH\,L\,VL & & M\,L * * * \end{array}$$

The opposite is also true, given a set of patterns which is a solution to the l-PATTERN IDENTIFICATION problem, we index the features so that all patterns have "*" from a particular value of $k+1 \leq n$ (with n being the total number of features), while for characters $\{1, \ldots, k\}$, at least one pattern has a non-"*" symbol of the alphabet Σ. It follows that the first k features are a k-feature set.

1.7.5 Are These Actually the Same Problem?

Intuitively, this suggests that to get small cardinality feature sets from patterns, it is best to use, when there is a choice, patterns that have as many wild card symbols as possible. However, let's consider the following example:

$$\begin{array}{ccc} H * L * & & H\,L\,L \mid H \\ & \mapsto & H\,H\,L \mid M \\ - - - - & & - - - - - \\ * M * * & & H\,M\,H \mid H \\ & & H\,M\,M \mid H \end{array}$$

This means that there is an $l = 2$ feasible solution for the 2-PATTERN IDENTIFICATION problem. We now highlight that column 3 is actually a 1-feature set (i.e., a feature set of cardinality one that feature is all what you need to discriminate between the two groups). Following the previous discussion about how to derive a set of patterns from a minimum cardinality feature set, while possible, it would not be optimal (it would have 3 patterns, namely $* * L *$, $* * H *$ and $* * M *$).

The optimality criteria then separates these two problems, as an optimal solution does not "translate" in an optimal solution of the other.

Suppose now we have another set of five wines (see Table 1.9). For these wines, the smallest cardinality feature set has size 1. A kind of problem for a data miner…each individual feature does a perfect discrimination (too much choice!). In a certain way, this makes exploring high dimensional combinations irrelevant. However, if we look for patterns, we find $H\ L\ L\ *\ **$ and $*\ *\ *\ H\ L\ M$ which would help model building in a different way.

We leave to the readers, as an exercise, to think of other situations in which we have m samples of n features and there is no $(n-1)$-feature set but there is an n-feature set. In addition, there are cases where the minimum cardinality is achieved by an n-feature set yet there is a set P of patterns such that $|P| < m$ and $P \rightarrow (Good, Bad)$?

Table 1.9 Another toy example for a wine study

Sample	Free-sulphur dioxide	Alcohol	Sugar	Density	Acidity	pH	Class
8	H	L	L	L	H	H	Strongly rejected
9	H	L	L	M	H	L	Strongly rejected
10	H	L	L	M	M	H	Strongly rejected
11	M	H	H	H	L	M	Award winning
12	L	M	M	H	L	M	Award winning

In this case we have five other wines for which the *pH* levels have also been measured

In conclusion, these problems are different and both are useful for data mining, each of them bringing their specific power to the analysis of data.

There are several variations of this basic problem. For instance, we refer to the more mathematically oriented readers to a paper that describes several of variants of the l-PATTERN IDENTIFICATION problem [127]. The paper describes a number of different problems, their computational complexity and we envision that this is a new exciting area for the application of combinatorial optimization methods to business analytics problems.

1.8 How to Fool Yourself

We hope to have attracted your attention with this title to this subsection. It was Nobel Prize winner Richard Feynman who at Caltech's Commencement address of 1974[31] said:

> The first principle is that you must not fool yourself and you are the easiest person to fool.

It is not as simple as it looks not to fool yourself. You are "with yourself all day", so probably you have found every trick in the book to do it. We will now check out a few pieces of advice that relate to the practice of data science, aiming at both sides of the bridge between computer science and marketing/business analytics. No side is immune.

We may start with the "Three Learning Principles" of his highly recommendable and very engaging publicly available lecture of Yaser Abu-Mostafa's from his online course,[32] which is also contained in Chap. 5 of "Learning from Data" [9]. The principles relate to three main topics: *Occam's Razor, Sampling Bias* and *Data Snooping*. We then refer to *Unbiasing*, and *The Nature of Measurements*.

1.8.1 Occam's Razor

We have previously discussed this principle (also known as *"Lex Parsimoniae"*,[33] or *"the Law of Briefness"*), in Sect. 1.6.6. There are several versions but the original quote in Latin from William of Ockham (1285–1347) supposedly is:

> Pluralitas non est ponenda sine neccesitate.

that translates as: *Plurality should not be posited without necessity*. An alternative translation often cited is: *"Entities are not to be multiplied beyond necessity"*. The second one goes directly to the heart of some of the data science as classifiers are being composed of many elements. The research questions arising from the need of defining of the number of layers to be used in an artificial neural network, the number of nodes in a hidden layers, the creation of high-dimensional spaces for Support vector machines, etc. are all examples of a direct need of the application of the *"Lex Parsimoniae"* as a useful heuristic guideline in the development of good models.

We can also quote the great mathematician, geometer and philosopher Ptolemy (85–165 AD):

> We consider it a good principle to explain the phenomena by the simplest hypothesis possible.

[31] http://calteches.library.caltech.edu/3043/1/CargoCult.pdf.

[32] https://work.caltech.edu/lecture.html.

[33] https://www.britannica.com/topic/Occams-razor.

Ockham’s quote:

Frustra fit per plura, quod potest fieri per pauciora.

which translates as: *“It is pointless to do with more what can be done with fewer”*.[34] also points in the same direction. For instance, if we have a pair of feature sets for US presidential elections data, one with 5 and another with 6 features, it seems reasonable to believe that the one with 5 (which we know is optimal) brings the simplest possible hypothesis.

Ptolemy’s quote brings another interesting point for discussion. What is “the phenomena” we are trying to explain? For instance, when we have defined the k-FEATURE SET problem, we were asked to identify a set of features such that for which every pair of samples that belong to different classes, at least one of the features should be in a different value. This basically is a *feasibility* requirement. We could have selected, in addition, a quantitative measure associated with a feature set. For each feature, we could count the number of pairs of samples of different classes for which the feature has different values, and then add up the values obtained for all the features. We have computed these numbers for the 23 5-feature sets, and one 5-feature set scores the highest (648) of sample pairs that these five features discriminate (see Fig. 2 of [161]). This means that the ratio of $648/5$ is the optimal value achievable for 5-feature sets, but if we open this new line of research, would it be possible for a 6-feature set to discriminate m pairs of such elections such that $m/6 > 648$? Wouldn’t it be reasonable to identify the k-feature set for which this new ratio score is maximum?

The reader may question our motives and argue that we are moving the goal posts with our redefinition of what “the phenomena” is. That is correct. We have done it so to entice the curiosity about this subject and the craftsmanship of “modelling” via a formal mathematical problem and its opportunities. The selection of an *objective function* leads to significantly different solutions, addressing different questions/phenomena that the data can answer.

1.8.2 Data Snooping

Again from [9]:

If the data set has affected any step in the learning process, its ability to assess the outcome has been compromised.

Snooping then perfectly fits our section on “fooling ourselves” because there are some imperceptible ways you may affect your procedures, but discipline in data handling may prevent this to happen.

[34] https://en.wikiquote.org/wiki/William_of_Ockham.

For instance, you must select your predictive analytics techniques *before working or even seeing the data*. It is all right to use some a priori knowledge about the problem domain, *but not the actual data you are using*. This is a serious trap and some current ad hoc practices make things worse. Consider the scenario of many online data mining competitions. You (and your team) start to select predictive analytic techniques for a challenge posed online. They give you a training set, and on it you develop your first machine learning method. You know your results on the training set (assuming your keep some tidy lab notes) but also you can *peek* a bit on the performance on the hidden test set (i.e., you may get a score and a rank). In some cases you have access to other people's codes which may help you to discard some machine learning models if yours are different and less competitive. In this case it is clear that, if this process is reiterated at least one time, you are adapting your predictive techniques to the data. You may be even entirely changing your predictive techniques. Guilty of snooping, you can bet you are.

Unfortunately, some of the challenges available online may allow some data snooping to occur. If this is the case, they are doing harm in a pedagogical sense to the newcomers of the global data science community. However, if another dataset is "locked in a safe" and the winner of a competition is only decided on performance on that *locked* dataset, we may have, perhaps, taught a better lesson to the overall community and snooping has been avoided.

People often refer to *training*, and *validation* and *test* datasets.

- **Training set:** The dataset of samples used for building your predictive models, this is the dataset which will allow to tune the adaptive parameters (i.e., weights of a neural network, coefficients of a symbolic regression model, etc.) and help you build your mathematical models.
- **Validation set:** A secondary dataset of samples that helps to identify the *"hyperparameters"*. For instance, this set may help you get some insight on the architecture of the neural network, the basic building blocks of the genetic programming technique of a classifier, the number of hidden units in each layer, etc.
- **Test set:** Another dataset which is only used to assess the performance of the model.

At the time of testing the performance with the test set, no further adaptation should be required. It thus takes self-discipline (if you are gathering your own data) to lock out a large part of it to have a useful test set and never "snoop". Better consider it "radioactive" and not use it for anything after it has been collected.

1.8.3 Sampling Bias

The key message here is that if there is some bias in the way the sample was collected, then the outcome of the data analytics procedure would also be biased. Some typical situations on which Sampling Bias occurs is when data is collected based on judgement or convenience or if the selection is related, even indirectly, to a subset of variables of interest to be measured. Sampling bias is different from other concepts like *sampling error*, *limited sampling bias*.[35]

Secondary selection bias can also occur when a dataset has already been made available, for instance, as a result of a previous study. Surveys to consumers may be a case in point. They may have been created to understand certain aspects and may be biased to avoid including other important features. A recent report of the US Federal Trade Commission highlights the potential of sampling bias to be a factor of exclusion to some parts of the community and give case examples [60].

Confirmation bias occurs when the data is somehow skewed to try to prove a particular hypothesis. Consequently the process was guided to find variables that would lead to confirm the intuition. Later, other datasets that could potentially do not align well with proving the hypothesis are neglected or not treated with the same standards.

1.8.4 Unbiasing

Some authors called the union of the training and validation sets as the *design set*. We bring this denomination to stress that we should be in control of the study, and design it accordingly. In marketing and customer behaviour analytics there are situations in which we should be interested in biasing our search for a predictive model, so "unbiasing" could be a way of fooling ourselves. For instance, we study later in the book a dataset that contains expert information (by wine connoisseurs) on the quality of wines. While the values given to the thousands of wines would have a bell-shaped distribution on a 1–10 scale, and the temptation to obtain good predictive models for all values in the range is strong, but we may reconsider if this is reasonable. For a client with a business perspective, predictive models that explain why some wines are highly scored by connoisseurs may be much more profitable than a predictive model over all the scale. If such a study would involve biasing or trimming the training and validation sets that should be part of a well-planned experimental design. A complementary study would then be conducted to understand why some wines are very poorly scored. It may be the case that the set of features occurring in these two studies are different. The knowledge derived by properly biasing the search of classifiers (given a design set) would probably be more relevant.

In addition, in marketing, many consumer and product segmentation techniques are well established and they are a useful way to create carefully planned bias. A group of consumers who are interested in some action movies may rank them according to some key characteristics, which are likely to be highly different if they are comedies or musicals. The dual is also true, another group of consumers may

[35]http://www.scholarpedia.org/article/Sampling_bias.

rank action movies by different features and thus different models can be derived. By “unbiasing” we may end up with a situation in which we are skewing the final predictive model by working well “on average” but seriously waste the true potential of the dataset to bring very useful insights for specific subsets of consumers and products.

It would also be relevant to have in mind *Simpson’s Paradox* or the *amalgamation paradox*, which refers to the existence of trends in data that either reverse or disappear when the individual datasets are combined.[36] The presence of *outliers*, samples or values that can present as “normal” but can skew results. They could also bring important information.

1.8.5 The Nature of Measurements

This is perhaps one interesting and most direct way of fooling yourself, just treat the data you are using without considering how it was generated. If you want to fool yourself, just ignore that the measurements you have come from the physical world or a real-world situations and merely pay attention to the known characteristics often cited in data mining books (i.e., categorical, ordinal, nominal, numerical). Curious at it may seem, we have seen this frequently in practice and it depends on the background education of the data scientist. In these studies, you must proactively need to understand your “input”, it cannot be just an “array of integers”. Even the scales of measurement need to be carefully scrutinized when building models [184, 191].

1.9 How Not to Fool Others

Going back to Feynman’s address in 1974, the quote continues with:

> After you have not fooled yourself, it is easy not to fool other scientists. You just have to be honest in a conventional way after that.

In some sense, statistics is the scientific area that helps us to be “honest in a conventional way” with others, but at the same time supports us not to fool ourselves. It is then perhaps prudent to just comment on other typical errors that we observe and can creep into the data analytical pipeline, particularly when working with large teams.

[36]https://en.wikipedia.org/wiki/Simpson’s_paradox.

1.9.1 Communication, Materials, Methods and Limitations of the Study

The first "others" not to fool are generally the members of your own team. Today, most large-scale data analytic activities and data science studies rely on team work. Communication is key for data science and it starts with your analytics team early in the process. In our own experience, we have learnt a few lessons by looking at other disciplines and by our academic endeavours.

Even if you are working as a service for others, or inside a company and you do not wish to publish your results, a recommendation is to start working with the data "as if" you are planning to publish your study in a highly prestigious journal. The requirements tend to be higher and here is an area where academic practice can help to improve your work in the commercial setting. In addition, start writing the "Materials and Methods" section of your hypothetical article from day one, as soon as you start working with the data. This could constitute a "log" of what is the dataset, the nature of the measurements, the techniques used to acquire the data, the potential sources of errors, the nature of errors, etc. It should be a well-drafted document, shared by all members of the team, with perhaps a couple of co-leaders but shared responsibility.

Assuming you have a plan, you can also start writing a section of this hypothetical paper on "Limitations of the Study". There are a number of issues that can hinder your work and having these documents available for the whole team from the very beginning is essential. Perhaps another member of the team should take responsibility of that document. It will help to catalyse new ideas and get everybody on the same page about possible sources of confusion and bring non-trivial issues up to the surface. It may also help to identify potential problems with your original plan (your data may not be useful for the question you would like to address and could be re-factored accordingly).

1.9.2 Avoid Using Pseudo-Scientific Measurements

There is a common theme here with the other advice about considering the physical world and not to fool yourself by disregarding the processes used to create the measured values that constitute your data and the scales used [152]. This is an area in which domain information is critical to understand what is "pseudo-science" and we cannot cover it in real depth and is particularly critical when human data is used [191, 203]. Michell describes the *"psychometricians' fallacy"* which concludes that an attribute is quantitative from the premise it is ordinal [157].

Here we briefly point at the collective disinformation we could be contributing by using unjustified measurements and perpetuating bad practices.

One leading example of a score that is criticized is the Body Mass Index (BMI), widely used by a variety of studies to quantitatively give some information about the characteristics of an individual. We could be saying that is an old "metafeature" that has been perpetuating until today. As a measure it was proposed and used by Lambert Adolphe Jacques Quetelet from 1830 to 1850, and the term BMI was later coined by Ancel Keys in 1972. From our perspective in this chapter we can consider a "mixed product-ratio metafeature" since the BMI of a person is defined as the ratio between the weight in kilograms divided by the square of the height measured in metres. It is thus difficult to accept this as an "index" of some sort as it has dimensions of kg/(m^2) and it looks as ad hoc as one of those generated by a genetic programming method while trying to fit some data. It is hard to understand what it really "measures" (some argue that it is the person's "thickness", as subjective and imprecise a concept can be).

It is hard to believe that what some researchers now consider a "numerical hack" from the early 1800s has come up to the twenty-first century and it is used to decide health policy and even insurance coverage. Other more recent measures that have been proposed are the *body adiposity index*, and the *waist-to-height ratio* (which seem to have an equal luck in describing something of relevance, the latter makes Beyoncé and Kate Moss differing by only 0.0001 on this index)[37] and it is categorizing Marilyn Monroe as "extremely slim" in comparison.

We thus support that, with current predictive analytic techniques, it makes more sense to work with the individual features as these compound measures are highly deceiving. Worse, continuous use of these simple to measure but non-scientific measures take out the attention of researchers from the more sound methods to measure obesity levels.[38]

This said, one possible alternative, whenever we need to relate our results with current understanding that involves some of these measurements, is to plan the analysis with and without these aggregated or questionable scores. This would allow readers and users to the data to put your results in context of the current literature and understand the value of the raw data per se.

1.9.3 Falsifiability and the Importance of Complete Search Processes

Albert Einstein is quoted as saying:

> No amount of experimentation can ever prove me right; a single experiment can prove me wrong.

There are many data analytics studies that they do not take into account that, simply, the results of the experiment cannot prove us wrong. Finding a single black swan can prove that the statement *"All swans are white"* is false. But black-necked swans

[37]https://en.wikipedia.org/wiki/Waist-to-height_ratio.

[38]http://www.npr.org/templates/story/story.php?storyId=106268439.

(the *Cygnus melancoryphus*) also exist... You may ask yourself: "is the experiment prepared to bring all possible outcomes of 'not white' for a swan?"

The important concept of falsifiability took strength after publication in Karl Popper's *"The Logic of Scientific Discovery"* in 1959 (for re-edition see [174]). In the context of what we are discussing here we echo: *Data should have some a priori chance to falsify a hypothesis*, or quoting from Problem 5.1. in [9].

> If the outcome of an experiment has no chance of falsifying a particular proposition, then the result of that experiment does not provide evidence one way or another toward the truth of that proposition.

It is interesting then to reiterate one of the benefits of complete search techniques for optimization. We can illustrate this by referring to the US Presidential elections dataset again. An optimization algorithm for the MIN *k*-FEATURE SET problem gave us a 5-feature set as a solution. This is a strong result, a complete one, it shows that no 4-feature set exists. No association rule with four features can explain *all* the elections results with just four features *using the information provided by that dataset*. Optimization also allowed us to find all the other 23 (optimal) 5-feature sets that exist. This also means that any hypothetical model involving more variables than five would not necessarily be more explanatory of the observed data than any of the optimal 5-feature sets. If, however, our search technique would have not been complete, we could have potentially left out one of the 23 optimal solutions. Optimization then plays an important role in not leaving outside any potentially explanatory set of variables, which in turn could lead to new interesting hypotheses about the nature of the process that generates the data. In [174] the author starts with an interesting quote: *"Hypotheses are nets: only he who casts will catch. (Novalis)"*. Here we are actually "catching hypotheses" from the data (and we got 23 interesting ones). Whether they are predictive or not remains to be seen, regularly with one new US election every 4 years, on their performance on independent sets. If, however, our search would have not been complete we would have not had a very good "casting" and we could have missed at least one of them.

1.10 To Know More: Further Bibliographic Sources to Consider When "Bridging the Gap"

There are many approaches to the analysis of data that, obviously, cannot be covered in a single chapter. In this section, we try to give some pointers to other sources of information that can help "bridge the gap" between the fields. We also provide links to some applications that could be useful for people interested in discussion of "real-life" problems. This literature, in turn, can be useful to understand how practitioners embed a particular problem instance into representation domains that allow the analytic techniques to be deployed.

1.10.1 Support Vector Machines and Their Application in Business Analytics and Marketing

We start with a brief discussion on *Support vector machines*, so this section can be seen as a brief appendix on applications that would naturally follow the discussion of Sects. 1.6.2 and 1.6.3. Introductory surveys to mention include [35] which presents the subject and may be studied in connection with the greater context provided by Kotsiantis et al. [121] and Lin et al. [233]. Burges' 1998 tutorial on SVMs [31] and Christianini and Schölkopf's article in AI Magazine [46] are other highly cited references providing easy access to the topic.

The Wikipedia article on Support vector machines[39] is also a useful initial source of historical information and variants of the technique started by Vapnik in 1963 with the proposal of classification via maximum-margin hyperplanes. The subject is also well covered in Battiti and Brunato's 2015 book on machine learning and optimization [23].

We thus turn our discussion to some applications in business analytics and marketing since the number of applications in all areas of data science is really huge. Chang and Lin's article introducing a library for Support Vector Machines (SVMs) times [33], and Burges' tutorial [31] have already been cited more than 30,000 and 17,000 times, respectively, so it is not a big surprise we will find many applications in business analytics. These two articles alone can give an idea of the impact of these methods. Many books have been dedicated to the topic [2, 47, 163].

Solutions based on SVMs have been widely used in many different disciplines of research, including business and marketing applications. SVMs have started to be used by marketing researchers more commonly in the last two decades. For instance, Cheung et al.'s study investigating the use of SVMs improving recommendations for direct marketing campaigns in 2003 [36]. However, adoption of SVMs in marketing literature in general was slow [48]. Cui and Curry aimed to introduce SVMs to the Marketing literature in more detail and advocate for their usefulness in their 2005 publication in the quantitative-driven journal *Marketing Science* [48]. They investigated the success of using SVMs compared to other methods to predict consumer choice and found that SVMs are very promising for conducting "pure prediction" tasks as well as remain accurate when the consumer choice set was significantly increased by number of possible consumer choices.

A common business and marketing application in which SVMs are applied more recently is *customer churn* [44, 63, 234, 237]. Customer churn is a problem for business and marketing professionals we address in this book in Chap. 20. For instance, Coussement and Van Den Poel [44] used SVMs in a marketing application to continue bridging the gap where marketing applications of SVMs are scarce (previously outlined by Cui and Curry [48]). Coussement and Van Den Poel investigate customer churn (customers leaving a company or service subscription)

[39] https://en.wikipedia.org/wiki/Support_vector_machine.

in the context of a newspaper subscription. They find that “Support vector machines show good generalization performance when applied to noisy marketing data” but they highlight that SVMs only outperform traditional regression approaches when an optimal parameter selection method had been applied to the process.

SVMs can be a useful tool for more than just churn prediction (as cited above) and marketing segmentation [13]. In a study of wine quality (using the same dataset that Chaps. 16 and 3 of this book look at) Cortez et al. [42] found that a SVM outperformed regression models in predicting human wine taste preferences based on chemical properties of the wine. Methodological details of this dataset (e.g., features that are included) are outlined in Part VII (the Appendix) Chap. 26.

1.10.2 Neural Networks and Marketing Applications

Work on Artificial Neural Networks (NNs) have been more commonly used in marketing applications since the early 1990s but some researchers believe the number of applications have not been fully explored as it should [54].

A typical example in which NNs can be used to explain how they work in a marketing setting is that of a market response function [93] with sales as a dependent variable of four other variables: retail price, current and lagged advertising budgets and average monthly temperatures (explained by Mazanec [155]). The approach is relatively standard: the nodes at the bottom of Fig. 1.15a would represent the four variables. They would then each link to the “hidden layer” with weights that are adapted by a learning procedure (also as shown in Fig. 1.15a) which would change the weights. Outputs are defined in the interval [0,1]. Ultimately, this comes to an outcome predicting sales, which corresponds to the top perceptron as in the same Figure.

More recently, NNs are making useful contributions classifying consumers in tourism applications. One such example is that of Mazanec’s [155] work on NNs in segmenting the market to “classify tourists”. The author explains that this would require a more complex network including four layers. Specifically the input layer has to have a number of units corresponding to the number of variables to be taken into consideration for the proposed segmentation. Then, the classification algorithm implemented in the “hidden layers” needs to ensure an unequivocal mapping of each input data vector onto the output edge. This process makes use of the concept of “competitive learning”. Finally, as explained in [155], the output layer may consist of as many elements as there are segments in the consumer segmentation output.

Since then, there are many other applications in marketing and business using NNs to classify consumers into segments, to investigate and classify consumer relationships [231], forecasting consumer demand for certain products, services or for travel demand to a specific location [132], guiding managers’ new product development decisions [215], to segment consumers for better direct marketing strategies [108] or for repeat purchase modelling in direct marketing activities [21], among many more.

Using NNs to segment consumers in particular has been used quite commonly in tourism applications. As stated above, Mazanec [155] classified tourists into market segments using NNs. More recently Kim et al. [114] conducted a more specific study on tourism investigating the market of West Australian senior tourists and segmenting using a NN approach. Disegna et al. used NNs to segment Italian tourists [57].

Another important application in marketing in which NNs have been used is the popular problem of customer *churn prediction* (as with SVMs). Chapter 20 of this book covering customer churn investigates the same dataset as Sharma and Panigrahi [202] who use NNs to predict customer churn outcomes. Predicting customer churn (i.e., whether a customer is going to leave the company or service provider) can be a very worthwhile task for companies as it is generally known that acquiring new customers is more expensive than retaining existing ones. If a company (for instance, a telecom services company like the one of the churn dataset) is able to identify which customers are going to churn, it is highly beneficial to them as they can act upon this information and hopefully retain those customers through enticing offers or deals. This is just one of the many applications, advancements in these methods are making a difference for businesses and their success.

In Table 1.10 we provide a very brief and introductory survey on NNs and some selected applications in business and consumer analytics applications. This survey provides the reader with further reading material in this area and a partial view of the vast current literature on NNs in business analytics.

1.11 A Note About the Contributions of This Book and Further Reading

This introduction helps us to put the contributions in the context of current developments in computer science for marketing and business analytics. Needless to say, we are perfectly aware that it is not possible to cover all areas in one or two volume of work. We have selected some "new ideas" that can possibly lead to novel approaches.

We also recommend a number of textbooks that can now help readers to "cross the gap" and study the basics of computer science having in mind their intended destiny in the consumer analytics and business fronts.

- **Discrete Mathematics**—A number of textbooks can help to "bridge the gap" by understanding basic mathematical notation and fundamental concepts normally covered in courses in the area of discrete mathematics. Richard Johnsonbaugh's *Discrete Mathematics*, now in its eight edition since 2009 [99], is a book that covers the basics notions: *Sets, Logic, Introduction to Algorithms, Graph Theory, etc.* The book has an appendix on the necessary concepts from Algebra and Matrices and even discusses issues related to the computation of the convex hull of a set of points, so it is an interesting reading for those who want to delve

Table 1.10 A brief survey review of some applications of artificial neural networks in marketing and consumer analytics

Application area and paper	Technique	Key characteristics
Product sales forecasting (of television sets) [232]	Neural networks	Turned an artificial NN into a useful marketing decision support system
Customer grouping in E-commerce [171]	Fuzzy ART neural network	A company's internal data is used to cluster customers into groups for better marketing approaches
Brand scanner forecasting using time-series data [98]	Bayesian vector autoregression (BVAR) neural network	Generalized autoregressive conditional heteroscedasticity (GARCH) models are outperformed by the BVAR NN methods
Customer credit analytics [134]	Neural networks	Use of multivariate adaptive regression splines
Customer satisfaction of fragrances assessment [112]	Neural networks	Method is validated using a case study dataset on customer fragrance note preferences
E-supermarket customer classification [213]	Modified fuzzy neural networks	A fuzzy NN is used for better customer classification and personalization possibilities
Assigning discounts in a marketing campaign [78]	Neural networks (self-organizing map (SOM) and multi-layer perceptron (MLP))	The SOM approach allows an intuitive interpretation of the results, and the MLP approach yields robust generalizable results
Churn prediction [219]	Hybrid neural networks and SOMs	Two hybrid models are compared (NN+NN and NN+SOM). Both hybrid models outperform single NN models and the NN+NN outperforms the NN+SOM approach
Dental services marketing-recognizing profitable customers [135]	Multi-layer feed-forward neural networks	The NN recognition model was successfully used to recognize profitable customers for dental services
Target marketing online shopping consumers [165]	Neural networks and logistic regression	NNs are compared with logistic regression and found to be comparable
Modelling and forecasting the impact of sales promotions [180]	Neural networks (use of additive and multiplicative models)	The NNs approach outperforms all other methods in forecasting and predicting sales promotion outcomes
Customer churn dataset of a telecom company [214]	Neural networks compared to standard statistical approaches	Improving profitability in direct marketing campaigns
Direct mail marketing response rates [83]	Neural networks compared to multiple regression analysis and logistic regression analysis	NNs can be used more effectively than the other two methods for targeting marketing campaigns

(continued)

Table 1.10 (continued)

Application area and paper	Technique	Key characteristics
Prediction of bank deposit subscription in telemarketing campaigns [159]	Neural networks	Predicting customer lifetime value had better results when using NNs than with baseline approaches
Mobile customers behaviour and usage patterns [75]	Self-organizing map neural networks	The output of the SOMs provides interpretable results for market segmentation strategies
Future customer purchase prediction using Twitter data [120]	Recurrent neural networks	The dataset was annotated by a human annotator for customer purchase relevance and analysed by a NN approach
Customer behaviour analysis in the financial industry [12]	Hybrid data mining and neural networks method	The hybrid model developed is new and is effective at segmenting customers regarding consumer behaviour and credit scoring
Customer satisfaction in survey collection [164]	3D convolutional neural networks	Customer satisfaction is estimated using sensory perceptors and analysed using NNs to predict and find out customer satisfaction (and dissatisfaction) during survey responses in market research
Conversion prediction of an online education comparison service software [192]	Deep learning NNs	This is a masters' thesis and would provide the reader with a good overview of deep learning methods, its advantages and disadvantages and further information
Financial market prediction using a large-scale dataset of limit order book events [220]	Deep learning NNs	Price change indication predictions are made and are more successful than linear SVMs and MLPs

more into concepts discussed in this introduction. Kenneth H. Rosen's *Discrete Mathematics and Its Applications* [189] is also another excellent introduction to the basics of these topics.

- **Algorithms**—There are many books that can be used as an introduction to the study of algorithms. Another work from Richard Johnsonbaugh (together with Marcus Schaefer) [100] can complement introductory study (or even self-study) relatively well, as they can be employed in tandem. The book provides examples on several techniques discussed in this introductory chapter (e.g., greedy/constructive algorithms, polynomial-time and NP-complete problems) as well as others discussed in other chapters of the book (e.g., an introduction to fixed-parameter tractability and the vertex cover problem). For those who are interested to have an introduction to algorithms but linked to some computer programming language, the collection created by Bruno R. Preiss could be of interest. Originally intended in 2000 to introduce object-oriented design and the methodology of *design patterns* [81, 176] using the Java computer language early in the computer science curricula [177], the series now also includes versions in several languages (Python, C#, C++, Java).

- **Logic**—Undoubtedly, the area of Logic can help to "bridge the gap" of an understanding between people trained in mathematics and computer science with those coming from the areas of business and marketing. Translation of findings from data would, most likely, need to be communicated to others in natural language and also by some formal language. This said, this is an avenue that would need mutual cooperation to transit. Tracey Bowel and Gary Kemp's *Critical Thinking: A Concise Guide* [28] aims at understanding the rules of argumentation, differentiate "knowledge" vs. "opinion" or "truth", and other topics to help with the study of natural language. It should be probably complemented with Copi, Cohen and McMahon's *Introduction to Logic* [41]. Evangelos Triantaphyllou's *Data Mining and Knowledge Discovery via Logic-Based Methods: Theory, Algorithms, and Applications* [218] would help readers to "connect the dots" between the areas of data mining, logic and the resulting optimization problems in graphs.
- **Neural Networks**—A large number of books have been written on the topic of artificial neural networks. We followed here a historical approach which is also, in our humble opinion, quite pedagogical at the same time. Issues like linear separability, maximum margin classifiers and the creation of systems composed of "layers" of individual units follow a coherent thread, leading to the *Deep Learning* architectures of today. We thus recommend a number of textbooks which can complement these discussions. The previously cited *Learning from Data* [9] of Abu-Mostafa, Magdon-Ismail and Lin is essential reading to go further in depth on the basics of machine learning. In addition, Battiti and Brunato's 2015 book in learning and optimization [23] helps to understand the strong link between these two topics, as clever optimization algorithms enable fundamentally more relevant machine learning methods. Another introductory textbook which may be of interest for business and marketing practitioners is the one of Kelleher, MacNamee and D'Arcy [111]. Michael Nielsen's free online book[40] aims at presenting deep learning in a way that naturally flows with the style of our chapter and may be interesting for some readers to check out. Da Costa Lewis in [49] presents deep learning and application methods using the R programming language in the areas of modelling customer brand choice, customer churn, product demand forecasting, prediction of credit card expenditure and automobile value forecasting.
- **Evolutionary Computation and Genetic Programming**—A large number of articles have been dedicated to evolutionary computation techniques in the area of business analytics.

  A number of other works could be recommended, including a relatively recent work on genetic programming in business forecasting [118], part of the series in *Genetic Programming Theory and Practice* (published by Springer with

[40] http://neuralnetworksanddeeplearning.com.

more than twelve volumes dedicated to the topic). The *Handbook of Genetic Programming* contains a number of applications of the technique to classification of streaming data [223], production of a semi-passive index portfolio in the stock market [119], data mining [201] and mining of association rules [148].

Finally, the *Handbook of Memetic Algorithms* [166] together with the surveys and reviews recommended in the introductory chapter of the corresponding section on the topic in this book can help readers to approach this subject area. It contains many case studies and several combinations of the basic techniques.

Acknowledgements The authors would like to thank Luke Mathieson, Ivana Ljubic and Shannon Fenn for helpful comments on early drafts of this chapter. Pablo Moscato acknowledges previous support from the Australian Research Council Future Fellowship FT120100060 and Australian Research Council Discovery Projects DP120102576 and DP140104183. He acknowledges a fruitful discussion with Prof. Keilis-Borok in 1991 at the International Centre for Theoretical Physics, in Trieste, Italy, and also thanks the organizers of the Workshop on Non-Linear Dynamics and Earthquake Prediction that facilitated his participation by funding his visit and attendance to the event.

References

1. *Third International Conference on Document Analysis and Recognition, ICDAR 1995, August 14–15, 1995, Montreal, Canada. Volume I.* IEEE Computer Society, 1995.
2. Shigeo Abe. *Support Vector Machines for Pattern Classification*. Advances in Pattern Recognition. Springer, 2010.
3. Yaser S. Abu-Mostafa. Learning from hints in neural networks. *J. Complexity*, 6(2):192–198, 1990.
4. Yaser S. Abu-Mostafa. A method for learning from hints. In Stephen Jose Hanson, Jack D. Cowan, and C. Lee Giles, editors, *Advances in Neural Information Processing Systems 5, [NIPS Conference, Denver, Colorado, USA, November 30 - December 3, 1992]*, pages 73–80. Morgan Kaufmann, 1992.
5. Yaser S. Abu-Mostafa. Hints and the VC dimension. *Neural Computation*, 5(2):278–288, 1993.
6. Yaser S. Abu-Mostafa. Learning from hints. *J. Complexity*, 10(1):165–178, 1994.
7. Yaser S. Abu-Mostafa. Hints. *Neural Computation*, 7(4):639–671, 1995.
8. Yaser S. Abu-Mostafa. Financial model calibration using consistency hints. *IEEE Trans. Neural Networks*, 12(4):791–808, 2001.
9. Yaser S. Abu-Mostafa, Malik Magdon-Ismail, and Hsuan-Tien Lin. *Learning From Data*. AMLBook, 2012.
10. Rakesh Agrawal, Tomasz Imielinski, and Arun N. Swami. Mining association rules between sets of items in large databases. In Peter Buneman and Sushil Jajodia, editors, *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data, Washington, D.C., May 26–28, 1993*, pages 207–216. ACM Press, 1993.
11. Ravindra K. Ahuja, Thomas L. Magnanti, and James B. Orlin. *Network flows - theory, algorithms and applications*. Prentice Hall, 1993.
12. Mahmood Alborzi and Mohammad Khanbabaei. Using data mining and neural networks techniques to propose a new hybrid customer behaviour analysis and credit scoring model in banking services based on a developed RFM analysis method. *IJBIS*, 23(1):1–22, 2016.
13. Pedro Albuquerque, Solange Alfinito, and Claudio V. Torres. Support vector clustering for customer segmentation on mobile tv service. *Communications in Statistics - Simulation and Computation*, 44(6):1453–1464, 2015.

14. Fawaz Alsolami, Talha Amin, Mikhail Moshkov, and Beata Zielosko. Comparison of heuristics for optimization of association rules. In Zbigniew Suraj and Ludwik Czaja, editors, *Proceedings of the 24th International Workshop on Concurrency, Specification and Programming, Rzeszow, Poland, September 28–30, 2015*, volume 1492 of *CEUR Workshop Proceedings*, pages 4–11. CEUR-WS.org, 2015.
15. Edoardo Amaldi, Stefano Coniglio, and Leonardo Taccari. Discrete optimization methods to fit piecewise affine models to data points. *Computers & OR*, 75:214–230, 2016.
16. Edoardo Amaldi and Viggo Kann. The complexity and approximability of finding maximum feasible subsystems of linear relations. *Theor. Comput. Sci.*, 147(1&2):181–210, 1995.
17. Edoardo Amaldi and Marco Mattavelli. The MIN PFS problem and piecewise linear model estimation. *Discrete Applied Mathematics*, 118(1-2):115–143, 2002.
18. Tom Anderson. Pittsburgh area mall sells for $100 at a foreclosure auction. http://goo.gl/CQptpN, January 2017. (Accessed on 12/13/2017).
19. Ahmed Shamsul Arefin, Luke Mathieson, Daniel Johnstone, Regina Berretta, and Pablo Moscato. Unveiling clusters of RNA transcript pairs associated with markers of Alzheimer's disease progression. *PLOS ONE*, 7(9):1–25, 09 2012.
20. Marcelo G. Armentano, Jie Tang, and Virginia Yannibelli, editors. *Proceedings of the 3rd International Workshop on Social Influence Analysis co-located with 26th International Joint Conference on Artificial Intelligence (IJCAI) 2017, Melbourne, Australia, August 19, 2017*, volume 1893 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2017.
21. Bart Baesens, Stijn Viaene, Dirk Van den Poel, Jan Vanthienen, and Guido Dedene. Bayesian neural network learning for repeat purchase modelling in direct marketing. *European Journal of Operational Research*, 138(1):191–211, 2002.
22. Roberto Battiti. Using mutual information for selecting features in supervised neural net learning. *IEEE Trans. Neural Networks*, 5(4):537–550, 1994.
23. Roberto Battiti and Mauro Brunato. *The LION way. Machine Learning plus Intelligent Optimization*. LIONlab, University of Trento, Italy, 2014.
24. Kristin P. Bennett and Erin J. Bredensteiner. Duality and geometry in SVM classifiers. In Pat Langley, editor, *Proceedings of the Seventeenth International Conference on Machine Learning (ICML 2000), Stanford University, Stanford, CA, USA, June 29 - July 2, 2000*, pages 57–64. Morgan Kaufmann, 2000.
25. Avrim Blum and Ronald L. Rivest. Training a 3-node neural network is NP-complete. *Neural Networks*, 5(1):117–127, 1992.
26. Avrim Blum and Ronald L. Rivest. Training a 3-node neural network is NP-complete. In Stephen Jose Hanson, Werner Remmele, and Ronald L. Rivest, editors, *Machine Learning: From Theory to Applications - Cooperative Research at Siemens and MIT*, volume 661 of *Lecture Notes in Computer Science*, pages 9–28. Springer, 1993.
27. Remco R. Bouckaert, Eibe Frank, Mark A. Hall, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. WEKA - experiences with a java open-source project. *Journal of Machine Learning Research*, 11:2533–2541, 2010.
28. T. Bowell and G. Kemp. *Critical Thinking: A Concise Guide*. Taylor & Francis, 2014.
29. Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.
30. Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual web search engine. *Computer Networks*, 30(1–7):107–117, 1998.
31. Christopher J. C. Burges. A tutorial on support vector machines for pattern recognition. *Data Min. Knowl. Discov.*, 2(2):121–167, 1998.
32. Nicolò Cesa-Bianchi and Gábor Lugosi. *Prediction, learning, and games*. Cambridge University Press, 2006.
33. Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A library for support vector machines. *ACM TIST*, 2(3):27:1–27:27, 2011.
34. Kuan-Yu Chen and Cheng-Hua Wang. Support vector regression with genetic algorithms in forecasting tourism demand. *Tourism Management*, 28(1):215–226, 2007.
35. Yiling Chen and Isaac G. Councill. An introduction to support vector machines: A review. *AI Magazine*, 24(2):105–107, 2003.

36. Kwok-Wai Cheung, James T. Kwok, Martin H. Law, and Kwok-Ching Tsui. Mining customer product ratings for personalized marketing. *Decision Support Systems*, 35(2):231–243, 2003. Web Data Mining.
37. Pradeep Chintagunta, Dominique M. Hanssens, and John R. Hauser. Editorial - marketing science and big data. *Marketing Science*, 35(3):341–342, 2016.
38. Darren M. Chitty. Improving the performance of GPU-based genetic programming through exploitation of on-chip memory. *Soft Comput.*, 20(2):661–680, 2016.
39. Andrzej Cichocki. Tensor networks for big data analytics and large-scale optimization problems. *CoRR*, abs/1407.3124, 2014.
40. Brendan Cody-Kenny, Edgar Galván López, and Stephen Barrett. locoGP: Improving performance by genetic programming java source code. In Sara Silva and Anna Isabel Esparcia-Alcázar, editors, *Genetic and Evolutionary Computation Conference, GECCO 2015, Madrid, Spain, July 11–15, 2015, Companion Material Proceedings*, pages 811–818. ACM, 2015.
41. I.M. Copi, C. Cohen, and K. McMahon. *Introduction to Logic*. Pearson, 2013.
42. Paulo Cortez, António Cerdeira, Fernando Almeida, Telmo Matos, and José Reis. Modeling wine preferences by data mining from physicochemical properties. *Decision Support Systems*, 47(4):547–553, 2009. Smart Business Networks: Concepts and Empirical Evidence.
43. Carlos Cotta, Antonio José Fernández Leiva, and José E. Gallardo. Memetic algorithms and complete techniques. In Neri et al. [166], pages 189–200.
44. Kristof Coussement and Dirk Van den Poel. Churn prediction in subscription services: An application of support vector machines while comparing two parameter-selection techniques. *Expert Systems with Applications*, 34(1):313–327, 2008.
45. Nichael Lynn Cramer. A representation for the adaptive generation of simple sequential programs. In John J. Grefenstette, editor, *Proceedings of the 1st International Conference on Genetic Algorithms, Pittsburgh, PA, USA, July 1985*, pages 183–187. Lawrence Erlbaum Associates, 1985.
46. Nello Cristianini and Bernhard Schölkopf. Support vector machines and kernel methods: The new generation of learning machines. *AI Magazine*, 23(3):31–42, 2002.
47. Nello Cristianini and John Shawe-Taylor. *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*. Cambridge University Press, 2010.
48. Dapeng Cui and David Curry. Prediction in marketing using the support vector machine. *Marketing Science*, 24(4):595–615, 2005.
49. N. Da Costa Lewis. *Deep Learning for Business with R: A Very Gentle Introduction to Business Analytics Using Deep Neural Networks*. CreateSpace Independent Publishing Platform, 2016.
50. S. Davies and S. Russell. NP-completeness of searches for smallest possible feature sets. In *Proceedings of the AAAI Symposium on Relevance*, pages 41–43, 1994.
51. Natalie Jane de Vries, Ahmed Shamsul Arefin, and Pablo Moscato. Gauging heterogeneity in online consumer behaviour data: A proximity graph approach. In *2014 IEEE Fourth International Conference on Big Data and Cloud Computing, BDCloud 2014, Sydney, Australia, December 3–5, 2014*, pages 485–492. IEEE Computer Society, 2014.
52. K. DeCell and A.J. Lichtman. *The Thirteen Keys to the Presidency*. Madison Books, 1990.
53. Erik D. Demaine, Fedor V. Fomin, Mohammad Taghi Hajiaghayi, and Dimitrios M. Thilikos. Fixed-parameter algorithms for (k, r)-center in planar graphs and map graphs. *ACM Transactions on Algorithms*, 1(1):33–47, 2005.
54. Kristen Bell DeTienne and David H. DeTienne. Neural networks in strategic marketing: exploring the possibilities. *Journal of Strategic Marketing*, 25(4):289–300, 2017.
55. Daniella Diaz. Donald trump looms behind Hillary Clinton at the debate - CNN Politics. https://goo.gl/6dduaN, October 2016. (Accessed on 12/13/2017).
56. Chris H. Q. Ding and Hanchuan Peng. Minimum redundancy feature selection from microarray gene expression data. *J. Bioinformatics and Computational Biology*, 3(2):185–206, 2005.

57. Marta Disegna, Isabella Procidano, and Silio R. Luchini. Segmenting the Italian tourists: A neural network approach. *Faculty of Tourism and Hospitality Management in Opatija.Biennial International Congress. Tourism and Hospitality Industry*, pages 60–73, 2010. Copyright - Copyright University of Rijeka, Faculty of Tourism and Hospitality Management 2010; Last updated - 2012-07-05; SubjectsTermNotLitGenreText - Italy.
58. Rodney G. Downey, Michael R. Fellows, and Ulrike Stege. Computational tractability: The view from mars. *Bulletin of the EATCS*, 69:73–97, 1999.
59. Harris Drucker, Christopher J. C. Burges, Linda Kaufman, Alexander J. Smola, and Vladimir Vapnik. Support vector regression machines. In *Advances in Neural Information Processing Systems 9, NIPS, Denver, CO, USA, December 2–5, 1996*, pages 155–161, 1996.
60. Maureen K. Ohlhausen Terrell McSweeny Edith Ramirez, Julie Brill. Big data: A tool for inclusion or exclusion?: Understanding the issues. https://goo.gl/WQqnaQ. (Accessed on 12/13/2017).
61. Andrzej Ehrenfeucht, David Haussler, Michael J. Kearns, and Leslie G. Valiant. A general lower bound on the number of examples needed for learning. *Inf. Comput.*, 82(3):247–261, 1989.
62. David A. Elizondo. The linear separability problem: some testing methods. *IEEE Trans. Neural Networks*, 17(2):330–344, 2006.
63. Guo en XIA and Wei dong JIN. Model of customer churn prediction on support vector machine. *Systems Engineering - Theory & Practice*, 28(1):71–77, 2008.
64. S. Ervelles, N. Fukawa, and L. Swayne. Big data consumer analytics and the transformation of marketing. *Journal of Business Research*, 69(2):897–904, 2016.
65. Thomas A. Feo and Mauricio G. C. Resende. Greedy randomized adaptive search procedures. *J. Global Optimization*, 6(2):109–133, 1995.
66. Thomas A. Feo, Mauricio G. C. Resende, and Stuart H. Smith. A greedy randomized adaptive search procedure for maximum independent set. *Operations Research*, 42(5):860–878, 1994.
67. Resende M, Ribeiro C (2016) Optimization by GRASP. Springer-Verlag New York.
68. Robyn Ffrancon and Marc Schoenauer. Memetic semantic genetic programming. In *Proceedings of the Genetic and Evolutionary Computation Conference, GECCO 2015, Madrid, Spain, July 11–15, 2015*, pages 1023–1030, 2015.
69. Richard Forsyth. BEAGLE A Darwinian approach to pattern recognition. *Kybernetes*, 10(3):159–166, 1981.
70. Les R. Foulds. *Combinatorial Optimization for Undergraduates*. Springer-Verlag New York, 1984.
71. Aviezri S. Fraenkel and David Lichtenstein. Computing a perfect strategy for n x n chess requires time exponential in n. *J. Comb. Theory, Ser. A*, 31(2):199–214, 1981.
72. Kunihiko Fukushima. Neocognitron: A hierarchical neural network capable of visual pattern recognition. *Neural Networks*, 1(2):119–130, 1988.
73. Kunihiko Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, 36(4):193–202, April 1980.
74. Kunihiko Fukushima, Sei Miyake, and Takayuki Ito. Neocognitron: A neural network model for a mechanism of visual pattern recognition. *IEEE Trans. Systems, Man, and Cybernetics*, 13(5):826–834, 1983.
75. Rawan Ghnemat and Edward Jaser. Classification of mobile customers behavior and usage patterns using self-organizing neural networks. *iJIM*, 9(4):4–11, 2015.
76. Malcom Gladwell. *Blink: The Power of Thinking without Thinking*. Bay Back Books, Little, Brown, 2005.
77. F. Glover and M. Laguna. *Tabu Search*. Kluwer Academic Publishers, Boston, MA, 1997.
78. Gabriel Gómez-Pérez, José David Martín-Guerrero, Emilio Soria-Olivas, Emili Balaguer-Ballester, Alberto Palomares, and Nicolás Casariego. Assigning discounts in a marketing campaign by using reinforcement learning and neural networks. *Expert Syst. Appl.*, 36(4):8022–8031, 2009.

79. Martin Gomez Ravetti and Pablo Moscato. Identification of a 5-protein biomarker molecular signature for predicting Alzheimer's disease. *PLOS ONE*, 3(9):1–12, 09 2008.
80. Maoguo Gong, Chao Song, Chao Duan, Lijia Ma, and Bo Shen. An efficient memetic algorithm for influence maximization in social networks. *IEEE Comp. Int. Mag.*, 11(3):22–33, 2016.
81. Dhrubajyoti Goswami, Ajit Singh, and Bruno R. Preiss. From design patterns to parallel architectural skeletons. *J. Parallel Distrib. Comput.*, 62(4):669–695, 2002.
82. J. L Gross and J Yellen. *Handbook of Graph Theory*. Discrete Mathematics and its Applications. CRC Press LLC, Boca Raton, Florida, 2004.
83. Gianluigi Guido, M. Irene Prete, Stefano Miraglia, and Irma De Mare. Targeting direct marketing campaigns by neural networks. *Journal of Marketing Management*, 27(9–10):992–1006, 2011.
84. Christoph Haase and Stefan Kiefer. The odds of staying on budget. In Magnús M. Halldórsson, Kazuo Iwama, Naoki Kobayashi, and Bettina Speckmann, editors, *Automata, Languages, and Programming - 42nd International Colloquium, ICALP 2015, Kyoto, Japan, July 6–10, 2015, Proceedings, Part II*, volume 9135 of *Lecture Notes in Computer Science*, pages 234–246. Springer, 2015.
85. Petr Hájek, Ivan Havel, and Metodej K. Chytil. The GUHA method of automatic hypotheses determination. *Computing*, 1(4):293–308, 1966.
86. Petr Hájek, Martin Holena, and Jan Rauch. The GUHA method and its meaning for data mining. *J. Comput. Syst. Sci.*, 76(1):34–48, 2010.
87. Adriana Hansberg, Dirk Meierling, and Lutz Volkmann. Distance domination and distance irredundance in graphs. *Electr. J. Comb.*, 14(1), 2007.
88. P. Hansen and N. Mladenović. Variable neighborhood search: Principles and applications. *European Journal of Operational Research*, 130(3):449–467, 2001.
89. J. B. Heaton, N. G. Polson, and J. H. Witte. Deep learning for finance: deep portfolios. *Applied Stochastic Models in Business and Industry*, 33(1):3–12, 2017. asmb.2209.
90. Jeff Heaton. Comparing dataset characteristics that favor the apriori, Eclat or FP-growth frequent itemset mining algorithms. *CoRR*, abs/1701.09042, 2017.
91. Tin Kam Ho. The random subspace method for constructing decision forests. *IEEE Trans. Pattern Anal. Mach. Intell.*, 20(8):832–844, 1998.
92. Steven C. H. Hoi, Jialei Wang, and Peilin Zhao. LIBOL: a library for online learning algorithms. *Journal of Machine Learning Research*, 15(1):495–499, 2014.
93. Harald Hruschka. Determining market response functions by neural network modeling: A comparison to econometric techniques. *European Journal of Operational Research*, 66(1):27–35, 1993.
94. Zhongyi Hu, Yukun Bao, and Tao Xiong. Comprehensive learning particle swarm optimization based memetic algorithm for model selection in short-term load forecasting using support vector regression. *Appl. Soft Comput.*, 25:15–25, 2014.
95. Laurent Hyafil and Ronald L. Rivest. Constructing optimal binary decision trees is NP-complete. *Inf. Process. Lett.*, 5(1):15–17, 1976.
96. Adnan Idris, Muhammad Rizwan, and Asifullah Khan. Churn prediction in telecom using random forest and PSO based data balancing in combination with various feature selection strategies. *Computers and Electrical Engineering*, 38(6):1808–1819, 2012.
97. Alexander G. Ororbia II, David Reitter, Jian Wu, and C. Lee Giles. Online learning of deep hybrid architectures for semi-supervised categorization. In Annalisa Appice, Pedro Pereira Rodrigues, Vítor Santos Costa, Carlos Soares, João Gama, and Alípio Jorge, editors, *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2015, Porto, Portugal, September 7–11, 2015, Proceedings, Part I*, volume 9284 of *Lecture Notes in Computer Science*, pages 516–532. Springer, 2015.
98. James J. Jiang, Maosen Zhong, and Gary Klein. Marketing category forecasting: An alternative of BVAR-artificial neural networks. *Decision Sciences*, 31(4):789–812, 2000.
99. R. Johnsonbaugh. *Discrete Mathematics*. Pearson Education, 2017.

100. R. Johnsonbaugh and M. Schaefer. *Algorithms*. JK computer science and mathematics series. Pearson Education, 2004.
101. Daniel Johnstone, Elizabeth A. Milward, Regina Berretta, Pablo Moscato, and for the Alzheimer's Disease Neuroimaging Initiative. Multivariate protein signatures of pre-clinical Alzheimer's disease in the Alzheimer's disease neuroimaging initiative (ADNI) plasma proteome dataset. *PLOS ONE*, 7(4):1–17, 04 2012.
102. J. Stephen Judd. Learning in neural networks. In David Haussler and Leonard Pitt, editors, *Proceedings of the First Annual Workshop on Computational Learning Theory, COLT '88, Cambridge, MA, USA, August 3–5, 1988.*, pages 2–8. ACM/MIT, 1988.
103. J. Stephen Judd. On the complexity of loading shallow neural networks. *J. Complexity*, 4(3):177–192, 1988.
104. J. Stephen Judd. *Neural network design and the complexity of learning*. Neural network modeling and connectionism. MIT Press, 1990.
105. J. Stephen Judd. Constant-time loading of shallow 1-dimensional networks. In John E. Moody, Stephen Jose Hanson, and Richard Lippmann, editors, *Advances in Neural Information Processing Systems 4, [NIPS Conference, Denver, Colorado, USA, December 2–5, 1991]*, pages 863–870. Morgan Kaufmann, 1991.
106. Giuseppe Jurman, Samantha Riccadonna, and Cesare Furlanello. A comparison of MCC and CEN error measures in multi-class prediction. *PLOS ONE*, 7(8):1–8, 08 2012.
107. Mir Md Jahangir Kabir, Shuxiang Xu, Byeong Ho Kang, and Zongyuan Zhao. A new multiple seeds based genetic algorithm for discovering a set of interesting Boolean association rules. *Expert Syst. Appl.*, 74:55–69, 2017.
108. Frederick Kaefer, Carrie M. Heilman, and Samuel D. Ramenofsky. A neural network application to consumer classification to improve the timing of direct marketing activities. *Computers & Operations Research*, 32(10):2595–2615, 2005. Applications of Neural Networks.
109. James Max Kanter and Kalyan Veeramachaneni. Deep feature synthesis: Towards automating data science endeavors. In *2015 IEEE International Conference on Data Science and Advanced Analytics, DSAA 2015, Campus des Cordeliers, Paris, France, October 19–21, 2015*, pages 1–10. IEEE, 2015.
110. S. Kase and N. Nishiyama. An Industrial Engineering Game Model for Factory Layout. *The Journal of Industrial Engineering*, XV(3):148–150, 1964.
111. J.D. Kelleher, B.M. Namee, and A. D'Arcy. *Fundamentals of Machine Learning for Predictive Data Analytics: Algorithms, Worked Examples, and Case Studies*. MIT Press, 2015.
112. Athakorn Kengpol and Worrapon Wangananon. The expert system for assessing customer satisfaction on fragrance notes: Using artificial neural networks. *Computers & Industrial Engineering*, 51(4):567–584, 2006.
113. Dongil Kim, Hyoung joo Lee, and Sungzoon Cho. Response modeling with support vector regression. *Expert Systems with Applications*, 34(2):1102–1108, 2008.
114. Jaesoo Kim, Sherrie Wei, and Hein Ruys. Segmenting the market of west Australian senior tourists using an artificial neural network. *Tourism Management*, 24(1):25–34, 2003.
115. Jun Woo Kim. Construction and evaluation of structured association map for visual exploration of association rules. *Expert Syst. Appl.*, 74:70–81, 2017.
116. S. Kirkpatrick, C.D. Gelatt Jr., and M.P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598):671–680, 1983.
117. Norbert Klasner and Hans Ulrich Simon. From noise-free to noise-tolerant and from on-line to batch learning. In Wolfgang Maass, editor, *Proceedings of the Eight Annual Conference on Computational Learning Theory, COLT 1995, Santa Cruz, California, USA, July 5–8, 1995*, pages 250–257. ACM, 1995.
118. Arthur K. Kordon. *Applying Genetic Programming in Business Forecasting*, pages 101–117. Springer New York, New York, NY, 2014.
119. Michael F. Korns. *Trading Volatility Using Highly Accurate Symbolic Regression*, pages 531–547. Springer International Publishing, Cham, 2015.
120. Mandy Korpusik, Shigeyuki Sakaki, Francine Chen, and Yan-Ying Chen. Recurrent neural networks for customer purchase prediction on twitter. In Toine Bogers, Marijn Koolen,

Cataldo Musto, Pasquale Lops, and Giovanni Semeraro, editors, *Proceedings of the 3rd Workshop on New Trends in Content-Based Recommender Systems co-located with ACM Conference on Recommender Systems (RecSys 2016), Boston, MA, USA, September 16, 2016.*, volume 1673 of *CEUR Workshop Proceedings*, pages 47–50. CEUR-WS.org, 2016.

121. S. B. Kotsiantis, I. D. Zaharakis, and P. E. Pintelas. Machine learning: A review of classification and combining techniques. *Artif. Intell. Rev.*, 26(3):159–190, November 2006.
122. John R. Koza. *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. MIT Press, Cambridge, MA, USA, 1992.
123. John R. Koza. *Genetic Programming II: Automatic Discovery of Reusable Programs*. MIT Press, Cambridge Massachusetts, May 1994.
124. John R. Koza. Human-competitive results produced by genetic programming. *Genetic Programming and Evolvable Machines*, 11(3/4):251–284, September 2010. Tenth Anniversary Issue: Progress in Genetic Programming and Evolvable Machines.
125. John R. Koza, David Andre, Forrest H Bennett III, and Martin Keane. *Genetic Programming III: Darwinian Invention and Problem Solving*. Morgan Kaufman, April 1999.
126. Nicolas Lachiche. Propositionalization. In Claude Gammut and Geoffrey I. Webb, editors, *Encyclopedia of Machine Learning*, pages 812–817. Boston, MA, 2010.
127. Giuseppe Lancia, Luke Mathieson, and Pablo Moscato. Separating sets of strings by finding matching patterns is almost always hard. *Theor. Comput. Sci.*, 665:73–86, 2017.
128. William B. Langdon and Mark Harman. Optimizing existing software with genetic programming. *IEEE Trans. Evolutionary Computation*, 19(1):118–135, 2015.
129. William B. Langdon, Brian Yee Hong Lam, Marc Modat, Justyna Petke, and Mark Harman. Genetic improvement of GPU software. *Genetic Programming and Evolvable Machines*, 18(1):5–44, 2017.
130. William B. Langdon, Brian Yee Hong Lam, Justyna Petke, and Mark Harman. Improving CUDA DNA analysis software with genetic programming. In *Proceedings of the Genetic and Evolutionary Computation Conference, GECCO 2015, Madrid, Spain, July 11–15, 2015*, pages 1063–1070, 2015.
131. William B. Langdon, Marc Modat, Justyna Petke, and Mark Harman. Improving 3d medical image registration CUDA software with genetic programming. In Dirk V. Arnold, editor, *Genetic and Evolutionary Computation Conference, GECCO '14, Vancouver, BC, Canada, July 12–16, 2014*, pages 951–958. ACM, 2014.
132. Rob Law and Norman Au. A neural network model to forecast Japanese demand for travel to Hong Kong. *Tourism Management*, 20(1):89–97, 1999.
133. Yann LeCun, Yoshua Bengio, and Geoffrey E. Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.
134. Tian-Shyug Lee and I-Fei Chen. Mining the customer credit using artificial neural networks and multivariate adaptive regression splines. In Hamid R. Arabnia, editor, *Proceedings of the International Conference on Artificial Intelligence, IC-AI '04, June 21–24, 2004, Las Vegas, Nevada, USA, Volume 1*, pages 133–139. CSREA Press, 2004.
135. Wan-I Lee and Bih-Yaw Shih. Application of neural networks to recognize profitable customers for dental services marketing-a case of dental clinics in Taiwan. *Expert Syst. Appl.*, 36(1):199–208, 2009.
136. A.A. Levis and L.G. Papageorgiou. Customer demand forecasting via support vector regression analysis. *Chemical Engineering Research and Design*, 83(8):1009–1018, 2005.
137. Theodore Levitt. Marketing myopia. *Harvard Business Review*, pages 45–56, 1960.
138. Chun-Liang Li, Yu-Chuan Su, Ting-Wei Lin, Cheng-Hao Tsai, Wei-Cheng Chang, Kuan-Hao Huang, Tzu-Ming Kuo, Shan-Wei Lin, Young-San Lin, Yu-Chen Lu, Chun-Pai Yang, Cheng-Xia Chang, Wei-Sheng Chin, Yu-Chin Juan, Hsiao-Yu Tung, Jui-Pin Wang, Cheng-Kuang Wei, Felix Wu, Tu-Chun Yin, Tong Yu, Yong Zhuang, Shou-De Lin, Hsuan-Tien Lin, and Chih-Jen Lin. Combination of feature engineering and ranking models for paper-author identification in KDD cup 2013. *Journal of Machine Learning Research*, 16:2921–2947, 2015.

139. Xun Liang, Rong-Chang Chen, Yang Bo He, and Ying Chen. Associating stock prices with web financial information time series based on support vector regression. *Neurocomputing*, 115:142–149, 2013.
140. A.J. Lichtman and V.I. Keilis-Borok. Pattern recognition applied to presidential elections in the united state 1860–1980: role of integral, social, economic and political traits. *Proceedings of the National Academy of Sciences USA*, 78:7230–7234, 1981.
141. Allan J. Lichtman. Why president Obama must endorse Hillary Clinton. https://goo.gl/exJh3e, May 2016. (Accessed on 12/13/2017).
142. Allan J. Lichtman and Ken DeCell. *The Thirteen Keys to the Presidency*. Madison Books, 1990.
143. Nathan Linial, Yishay Mansour, and Ronald L. Rivest. Results on learnability and the Vapnik-Chervonenkis dimension. *Inf. Comput.*, 90(1):33–49, 1991.
144. Paulo J. G. Lisboa. Interpretability in machine learning - principles and practice. In Francesco Masulli, Gabriella Pasi, and Ronald R. Yager, editors, *Fuzzy Logic and Applications - 10th International Workshop, WILF 2013, Genoa, Italy, November 19–22, 2013. Proceedings*, volume 8256 of *Lecture Notes in Computer Science*, pages 15–21. Springer, 2013.
145. Nick Littlestone. From on-line to batch learning. In Ronald L. Rivest, David Haussler, and Manfred K. Warmuth, editors, *Proceedings of the Second Annual Workshop on Computational Learning Theory, COLT 1989, Santa Cruz, CA, USA, July 31–August 2, 1989.*, pages 269–284. Morgan Kaufmann, 1989.
146. Chi-Jie Lu. Sales forecasting of computer products based on variable selection scheme and support vector regression. *Neurocomputing*, 128:491–499, 2014.
147. Alessandro Prudêncio Lukosevicius, Gustavo Guimarães Marchisotti, and Carlos Alberto Pereira Soares. Overview of complexity: main currents, definitions and constructs. *Sistemas & Gestão*, 11(4):455–465, 2017.
148. J. M. Luna, A. Cano, and S. Ventura. *Genetic Programming for Mining Association Rules in Relational Database Environments*, pages 431–450. Springer International Publishing, Cham, 2015.
149. Emad Mabrouk, Abdel-Rahman Hedar, and Masao Fukushima. Memetic programming with adaptive local search using tree data structures. In Richard Chbeir, Youakim Badr, Ajith Abraham, Dominique Laurent, Mario Köppen, Fernando Ferri, Lotfi A. Zadeh, and Yukio Ohsawa, editors, *CSTST 2008: Proceedings of the 5th International Conference on Soft Computing as Transdisciplinary Science and Technology, Cergy-Pontoise, France, October 28–31, 2008*, pages 258–264. ACM, 2008.
150. Emad Mabrouk, Julio César Hernández-Castro, and Masao Fukushima. Prime number generation using memetic programming. *Artificial Life and Robotics*, 16(1):53–56, Jun 2011.
151. Malik Magdon-Ismail, Hung-Ching Chen, and Yaser S. Abu-Mostafa. The multilevel classification problem and a monotonicity hint. In Hujun Yin, Nigel M. Allinson, Richard T. Freeman, John A. Keane, and Simon J. Hubbard, editors, *Intelligent Data Engineering and Automated Learning - IDEAL 2002, Third International Conference, Manchester, UK, August 12–14, Proceedings*, volume 2412 of *Lecture Notes in Computer Science*, pages 410–415. Springer, 2002.
152. Helen M. Marcus-Roberts and Fred S. Roberts. Meaningless statistics. *Journal of Educational Statistics*, 12(4):383–394, 1987.
153. D. Martín, Jesús Alcalá-Fdez, Alejandro Rosete, and F. Herrera. NICGAR: A niching genetic algorithm to mine a diverse set of interesting quantitative association rules. *Inf. Sci.*, 355–356:208–228, 2016.
154. Marco Mattavelli and Edoardo Amaldi. Estimating piecewise linear models using combinatorial optimization techniques. In *8th European Signal Processing Conference, EUSIPCO 1996, Trieste, Italy, 10–13 September, 1996*, pages 1–4. IEEE, 1996.
155. Josef A. Mazanec. Classifying tourists into market segments. *Journal of Travel & Tourism Marketing*, 1(1):39–60, 1992.
156. Nimrod Megiddo. On the complexity of polyhedral separability. *Discrete & Computational Geometry*, 3:325–337, 1988.

157. Joel Michell. The psychometricians' fallacy: Too clever by half? *British Journal of Mathematical and Statistical Psychology*, 62(1):41–55, 2009.
158. Marvin Minsky and Seymour Papert. *Perceptrons - an introduction to computational geometry*. MIT Press, 1987.
159. Sérgio Moro, Paulo Cortez, and Paulo Rita. Using customer lifetime value and neural networks to improve the prediction of bank deposit subscription in telemarketing campaigns. *Neural Computing and Applications*, 26(1):131–139, 2015.
160. Pablo Moscato. Memetic algorithms: The untold story. In *Handbook of Memetic Algorithms*, pages 275–309. Springer Berlin Heidelberg, 2012.
161. Pablo Moscato, Luke Mathieson, Alexandre Mendes, and Regina Berretta. The electronic primaries: Predicting the U.S. presidency using feature selection with safe data reduction. In Vladimir Estivill-Castro, editor, *Computer Science 2005, Twenty-Eighth Australasian Computer Science Conference (ACSC2005), Newcastle, NSW, Australia, January/February 2005*, volume 38 of *CRPIT*, pages 371–380. Australian Computer Society, 2005.
162. Pablo Moscato and Michael G. Norman. A "memetic" approach for the travelling salesman problem: Implementation of a computational ecology for combinatorial optimization on message-passing systems. In M. Valero, E. Onate, M. Jane, J. L. Larriba, and B. Suarez, editors, *Proceedings of PACTA '92 - International Conference on Parallel Computing and Transputer Applications, Sep. 21–25, Barcelona*, pages 177–186. IOS Press, Amsterdam, 1992.
163. M. N. Murty and Rashmi Raghava. *Support Vector Machines and Perceptrons - Learning, Optimization, Classification, and Application to Social Networks*. Springer Briefs in Computer Science. Springer, 2016.
164. Tomofumi Nakano and Shohei Kato. Validation of 3d convolutional neural networks for customer satisfaction estimation using videos. In Leonard Barolli, Tomoya Enokido, and Makoto Takizawa, editors, *Advances in Network-Based Information Systems, The 20th International Conference on Network-Based Information Systems, NBiS 2017, Ryerson University, Toronto, ON, Canada, August 24–26, 2017.*, volume 7 of *Lecture Notes on Data Engineering and Communications Technologies*, pages 453–462. Springer, 2017.
165. Mohammad B. Naseri and Greg Elliott. Targeting online customers. A comparative analysis of artificial neural networks and logistic regression. *Journal of Decision Systems*, 19(3):291–312, 2010.
166. Ferrante Neri, Carlos Cotta, and Pablo Moscato, editors. *Handbook of Memetic Algorithms*, volume 379 of *Studies in Computational Intelligence*. Springer, 2012.
167. Fabrice Talla Nobibon. *Algorithms for selection and graph-coloring problems with applications in marketing and micro-economics*. PhD thesis, Faculteit Economie En Bedrijfswetenschappen, KU Leuven, Katholieke Universiteit Leuven, 9 2010.
168. Mikel Olazaran. A sociological study of the official history of the perceptrons controversy. *Social Studies of Science*, 26(3):611–659, 1996.
169. Andrea Ovans. That Mad Men Computer, Explained by HBR in 1969. https://hbr.org/2014/05/that-mad-men-computer-explained-by-hbr-in-1969, May 2014. (Accessed on 4/24/2018).
170. Herbert Pang, Stephen L. George, Ken Hui, and Tiejun Tong. Gene selection using iterative feature elimination random forests for survival outcomes. *IEEE/ACM Trans. Comput. Biology Bioinform.*, 9(5):1422–1431, 2012.
171. Sungjune Park. Neural networks and customer grouping in e-commerce: A framework using fuzzy ART. In *2000 Academia/Industry Working Conference on Research Challenges (AIWORC 2000), Next Generation Enterprises: Virtual Organizations and Mobile/Pervasive Technologies, 27–29 April 2000, Buffalo, NY, USA*, pages 331–336. IEEE Computer Society, 2000.
172. Hanchuan Peng, Fuhui Long, and Chris H. Q. Ding. Feature selection based on mutual information: Criteria of max-dependency, max-relevance, and min-redundancy. *IEEE Trans. Pattern Anal. Mach. Intell.*, 27(8):1226–1238, 2005.
173. M. Pirlot. General local search methods. *European Journal of Operational Research*, 92(3):493–511, 1996.

174. Karl Popper. *The logic of scientific discovery*. Routledge, London/New York, 2005.
175. Doina Precup and Yee Whye Teh, editors. *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6–11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*. PMLR, 2017.
176. Bruno R. Preiss. Design patterns for the data structures and algorithms course. In Jane Prey and Robert E. Noonan, editors, *Proceedings of the 30th SIGCSE Technical Symposium on Computer Science Education, 1999, New Orleans, Louisiana, USA, March 14–28, 1999*, pages 95–99. ACM, 1999.
177. Bruno R. Preiss. *Data Structures and Algorithms with Object-Oriented Design Patterns in Java*. Worldwide Series in Computer Science. Wiley, 2000.
178. P. Roushini Leely Pushpam and S. Padmapriea. Global roman domination in graphs. *Discrete Applied Mathematics*, 200:176–185, 2016.
179. Resende MGC, Ribeiro CC (2003) Greedy Randomized Adaptive Search. In: Glover F, Kochenberger G (ed) Handbook of Metaheuristics. Springer US, Boston, MA, pp 219–249
180. Ibrahim Zafar Qureshi, Marwan Khammash, and Konstantinos Nikolopoulos. Turning artificial neural networks into a marketing science tool - modelling and forecasting the impact of sales promotions. In Joaquim Filipe and Ana L. N. Fred, editors, *ICAART 2011 - Proceedings of the 3rd International Conference on Agents and Artificial Intelligence, Volume 1 - Artificial Intelligence, Rome, Italy, January 28–30, 2011*, pages 698–702. SciTePress, 2011.
181. C.G Raptis, C.I Siettos, C.T Kiranoudis, and G.V Bafas. Classification of aged wine distillates using fuzzy and neural network systems. *Journal of Food Engineering*, 46(4):267–275, 2000.
182. David M. Reif, Alison A. Motsinger, Brett A. McKinney, James E. Crowe Jr., and Jason H. Moore. Feature selection using a random forests classifier for the integrated analysis of multiple data types. In *Proceedings of the 2006 IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology, CIBCB 2006, Renaissance Hotel Downtown, Toronto, Ontario, Canada, September 28–29, 2006*, pages 1–8. IEEE, 2006.
183. Mauricio G. C. Resende and Thomas A. Feo. A GRASP for satisfiability. In David S. Johnson and Michael A. Trick, editors, *Cliques, Coloring, and Satisfiability, Proceedings of a DIMACS Workshop, New Brunswick, New Jersey, USA, October 11–13, 1993*, volume 26 of *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, pages 499–520. DIMACS/AMS, 1993.
184. Fred S. Roberts. Meaningfulness of conclusions from combinatorial optimization. *Discrete Applied Mathematics*, 29(2-3):221–241, 1990.
185. J. M. Robson. The complexity of go. In *IFIP Congress*, pages 413–417, 1983.
186. J. M. Robson. N by N checkers is exptime complete. *SIAM J. Comput.*, 13(2):252–267, 1984.
187. Mateus Rocha de Paula, Regina Berretta, and Pablo Moscato. A fast meta-heuristic approach for the alpha beta- k -feature set problem. *Journal of Heuristics*, 22(2):199–220, Apr 2016.
188. Mateus Rocha de Paula, Martín Gómez Ravetti, Regina Berretta, and Pablo Moscato. Differences in abundances of cell-signalling proteins in blood reveal novel biomarkers for early detection of clinical Alzheimer's disease. *PLOS ONE*, 6(3):1–14, 03 2011.
189. Kenneth H. Rosen. *Discrete Mathematics and Its Applications*. McGraw-Hill Higher Education, 5th edition, 2002.
190. Frank Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65(6), 1958.
191. Giovanni B. Rossi and Francesco Crenna. On ratio scales. *Measurement*, 46(8):2913–2920, 2013.
192. Rutger Ruizendaal. The potential of deep learning in marketing. Master's thesis, University of Twente/StudyPortals, September 2017.
193. Mary-Ann Russon. Cia using deep learning neural networks to predict social unrest before it occurs. http://goo.gl/bT3pyH, October 2016. (Accessed on 12/13/2017).
194. Nasser R. Sabar, Masri Ayob, Rong Qu, and Graham Kendall. A graph coloring constructive hyper-heuristic for examination timetabling problems. *Appl. Intell.*, 37(1):1–11, 2012.
195. Douglas A. Samuelson. Election 2012: The '13 keys' to the white house - informs. https://goo.gl/ewiK3c. (Accessed on 12/13/2017).

196. Sanjoy Sarkar, Tillman Weyde, Artur S. d'Avila Garcez, Gregory G. Slabaugh, Simo Dragicevic, and Chris Percy. Accuracy and interpretability trade-offs in machine learning applied to safer gambling. In Tarek Richard Besold, Antoine Bordes, Artur S. d'Avila Garcez, and Greg Wayne, editors, *Proceedings of the Workshop on Cognitive Computation: Integrating neural and symbolic approaches 2016 co-located with the 30th Annual Conference on Neural Information Processing Systems (NIPS 2016), Barcelona, Spain, December 9, 2016.*, volume 1773 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2016.
197. Jürgen Schmidhuber. Deep learning in neural networks: An overview. *Neural Networks*, 61:85–117, 2015.
198. Jürgen Schmidhuber. Deep learning. In Claude Sammut and Geoffrey I. Webb, editors, *Encyclopedia of Machine Learning and Data Mining*, pages 338–348. Springer, 2017.
199. Michael Schmidt and Hod Lipson. Distilling free-form natural laws from experimental data. *Science*, 324(5923):81–85, 2009.
200. Peter Schuster. Definitions of complexity are notoriously difficult. In Peter Schuster, editor, *43 Visions for Complexity*, pages 61–62. World Scientific, 2016.
201. Dominic P. Searson. *GPTIPS 2: An Open-Source Software Platform for Symbolic Data Mining*, pages 551–573. Springer International Publishing, Cham, 2015.
202. A. Sharma and P. K. Panigrahi, Dr. A Neural Network based Approach for Predicting Customer Churn in Cellular Network Services. *ArXiv e-prints*, September 2013.
203. Klaas Sijtsma. Introduction to the measurement of psychological attributes. *Measurement*, 44(7):1209–1219, 2011.
204. Joseph Sill and Yaser S. Abu-Mostafa. Monotonicity hints. In *Advances in Neural Information Processing Systems 9, NIPS, Denver, CO, USA, December 2–5, 1996*, pages 634–640, 1996.
205. D. Silver, T. Hubert, J. Schrittwieser, I. Antonoglou, M. Lai, A. Guez, M. Lanctot, L. Sifre, D. Kumaran, T. Graepel, T. Lillicrap, K. Simonyan, and D. Hassabis. Mastering Chess and Shogi by Self-Play with a General Reinforcement Learning Algorithm. *ArXiv e-prints*, December 2017.
206. David Silver, Aja Huang, Chris J. Maddison, Arthur Guez, Laurent Sifre, George van den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Vedavyas Panneershelvam, Marc Lanctot, Sander Dieleman, Dominik Grewe, John Nham, Nal Kalchbrenner, Ilya Sutskever, Timothy P. Lillicrap, Madeleine Leach, Koray Kavukcuoglu, Thore Graepel, and Demis Hassabis. Mastering the game of go with deep neural networks and tree search. *Nature*, 529(7587):484–489, 2016.
207. David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, Yutian Chen, Timothy Lillicrap, Fan Hui, Laurent Sifre, George van den Driessche, Thore Graepel, and Demis Hassabis. Mastering the game of go without human knowledge. *Nature*, 550:354–359, 2017.
208. Peter Sondergaard. The internet of things will give rise to the algorithm economy - Peter Sondergaard. http://goo.gl/fNFr6n, June 2015. (Accessed on 12/13/2017).
209. Arua De M. Sousa, Ana Carolina Lorena, and Márcio P. Basgalupp. GEEK: grammatical evolution for automatically evolving kernel functions. In *2017 IEEE Trustcom/BigDataSE/ICESS, Sydney, Australia, August 1–4, 2017*, pages 941–948. IEEE, 2017.
210. Ramakrishnan Srikant and Rakesh Agrawal. Mining quantitative association rules in large relational tables. In H. V. Jagadish and Inderpal Singh Mumick, editors, *Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data, Montreal, Quebec, Canada, June 4–6, 1996.*, pages 1–12. ACM Press, 1996.
211. Pål Roe Sundsøy, Johannes Bjelland, Asif M. Iqbal, Alex Pentland, and Yves-Alexandre de Montjoye. Big data-driven marketing: How machine learning outperforms marketers' gut-feeling. In William G. Kennedy, Nitin Agarwal, and Shanchieh Jay Yang, editors, *Social Computing, Behavioral-Cultural Modeling and Prediction - 7th International Conference, SBP 2014, Washington, DC, USA, April 1–4, 2014. Proceedings*, volume 8393 of *Lecture Notes in Computer Science*, pages 367–374. Springer, 2014.
212. Matt Taibbi. "How America made Donald Trump unstoppable". *Rolling Stone*, February 24, 2016.

213. Yu-An Tan, Zuo Wang, and Qi Luo. Research on customer classification in e-supermarket by using modified fuzzy neural networks. In Derong Liu, Shumin Fei, Zeng-Guang Hou, Huaguang Zhang, and Changyin Sun, editors, *Advances in Neural Networks - ISNN 2007, 4th International Symposium on Neural Networks, ISNN 2007, Nanjing, China, June 3–7, 2007, Proceedings, Part II*, volume 4492 of *Lecture Notes in Computer Science*, pages 301–306. Springer, 2007.
214. Zaiyong Tang. Improving direct marketing profitability with neural networks. *International Journal of Computer Applications (0975–8887)*, 29(5):13–18, 09 2011.
215. R. Jeffrey Thieme, Michael Song, and Roger J. Calantone. Artificial neural network decision support systems for new product development project selection. *Journal of Marketing Research*, 37(4):499–507, 2000.
216. The New York Times. New navy device learns by doing; psychologist shows embryo of computer designed to read and grow wiser. http://goo.gl/RcdkyF, July 1958. (Accessed on 12/13/2017).
217. Craig A. Tovey. Tutorial on computational complexity. *Interfaces*, 32(3):30–61, 2002.
218. E. Triantaphyllou. *Data Mining and Knowledge Discovery via Logic-Based Methods: Theory, Algorithms, and Applications*. Springer Optimization and Its Applications. Springer US, 2010.
219. Chih-Fong Tsai and Yu-Hsin Lu. Customer churn prediction by hybrid neural networks. *Expert Syst. Appl.*, 36(10):12547–12553, 2009.
220. Avraam Tsantekidis, Nikolaos Passalis, Anastasios Tefas, Juho Kanniainen, Moncef Gabbouj, and Alexandros Iosifidis. Using deep learning to detect price change indications in financial markets. In *25th European Signal Processing Conference, EUSIPCO 2017, Kos, Greece, August 28 - September 2, 2017*, pages 2511–2515. IEEE, 2017.
221. A. M. Turing. Computing machinery and intelligence. *Mind*, 49:433–460, January 01 1950.
222. Jeffrey D. Ullman. NP-complete scheduling problems. *J. Comput. Syst. Sci.*, 10(3):384–393, 1975.
223. Ali Vahdat, Jillian Morgan, Andrew R. McIntyre, Malcolm I. Heywood, and Nur Zincir-Heywood. *Evolving GP Classifiers for Streaming Data Tasks with Concept Change and Label Budgets: A Benchmarking Study*, pages 451–480. Springer International Publishing, Cham, 2015.
224. Kalyan Veeramachaneni, Katya Vladislavleva, Matt Burland, Jason Parcon, and Una-May O'Reilly. Evolutionary optimization of flavours. In Martin Pelikan and Jürgen Branke, editors, *Genetic and Evolutionary Computation Conference, GECCO 2010, Proceedings, Portland, Oregon, USA, July 7–11, 2010*, pages 1291–1298. ACM, 2010.
225. M.G.A. Verhoeven and E.H.L. Aarts. Parallel local search. *Journal of Heuristics*, 1:43–65, 1995.
226. Bernd Vindevogel, Dirk van den Poel, and Geert Wets. Why promotion strategies based on market basket analysis do not work. *Expert Systems with Applications*, 28(3):583–590, 2005.
227. Haohan Wang, Bhiksha Raj, and Eric P. Xing. On the origin of deep learning. *CoRR*, abs/1702.07800, 2017.
228. Yisen Wang and Shu-Tao Xia. A novel feature subspace selection method in random forests for high dimensional data. In *2016 International Joint Conference on Neural Networks, IJCNN 2016, Vancouver, BC, Canada, July 24–29, 2016*, pages 4383–4389. IEEE, 2016.
229. Wikipedia. Heights of presidents and presidential candidates of the united states. https://goo.gl/R9sme6. (Accessed on 12/13/2017).
230. L. Wittgenstein. *Tractatus Logico-philosophicus*. International library of psychology, philosophy, and scientific method. Routledge, 1990.
231. Barry Wray, Adrian Palmer, and David Bejou. Using neural network analysis to evaluate buyer-seller relationships. *European Journal of Marketing*, 28(10):32–48, 1994.
232. Jingtao Yao, Nicholas Teng, Hean-Lee Poh, and Chew Lim Tan. Forecasting and analysis of marketing data using neural networks. *J. Inf. Sci. Eng.*, 14(4):843–862, 1998.
233. Hsuan-Tien Lin, Yaser S. Abu Mostafa, Malik Magdon-Ismail. *Learning from Data*. AML Book, 2012.

234. Xiaobing Yu, Shunsheng Guo, Jun Guo, and Xiaorong Huang. An extended support vector machine forecasting framework for customer churn in e-commerce. *Expert Systems with Applications*, 38(3):1425–1430, 2011.
235. Yiteng Zhai, Yew-Soon Ong, and Ivor W. Tsang. Making trillion correlations feasible in feature grouping and selection. *IEEE Trans. Pattern Anal. Mach. Intell.*, 38(12):2472–2486, 2016.
236. Hao Zhang, Zhiting Hu, Jinliang Wei, Pengtao Xie, Gunhee Kim, Qirong Ho, and Eric P. Xing. Poseidon: A system architecture for efficient GPU-based deep learning on multiple machines. *CoRR*, abs/1512.06216, 2015.
237. Yu Zhao, Bing Li, Xiu Li, Wenhuang Liu, and Shouju Ren. Customer churn prediction using improved one-class support vector machine. In Xue Li, Shuliang Wang, and Zhao Yang Dong, editors, *Advanced Data Mining and Applications, First International Conference, ADMA 2005, Wuhan, China, July 22–24, 2005, Proceedings*, volume 3584 of *Lecture Notes in Computer Science*, pages 300–306. Springer, 2005.
238. Beata Zielosko. Application of dynamic programming approach to optimization of association rules relative to coverage and length. *Fundam. Inform.*, 148(1–2):87–105, 2016.

Chapter 2
Consumer Behaviour and Marketing Fundamentals for Business Data Analytics

Natalie Jane de Vries and Pablo Moscato

Abstract This chapter provides the reader with a brief introduction to the basics of marketing. The intention is to help a "non-marketer" to understand what is needed in business and consumer analytics from a marketing perspective and continue "bridging the gap" between data scientists and business thinkers. A brief introduction to the discipline of marketing is presented followed by several topics that are crucial for understanding marketing and computational applications within the field. A background of market segmentation and targeting strategies is followed by the description of typical bases for segmenting a market. Further, consumer behaviour literature and theory is discussed as well as the current trends for businesses regarding consumer behaviour.

Keywords Advertising · Behaviouristic segmentation · Consumer behaviour · Market segmentation · Social media marketing · Targeted marketing

2.1 Introduction

As Chap. 1 has already mentioned, it is time to *bridge the gap* between data science and business and marketing. As it becomes increasingly important for everyone in business to understand data analytics in more detail in order to let new technologies bring advantages, so too has it become important for data scientists to generate a level of understanding of the problem domains their technologies apply in. As Chap. 1 stated, this has to become a two-way conversation where each party to the table has the required knowledge of the other's domain to "keep the conversation flowing".

N. J. de Vries (✉) · P. Moscato
School of Electrical Engineering and Computing, The University of Newcastle, Callaghan, NSW, Australia
e-mail: natalie.devries@newcastle.edu.au; Pablo.Moscato@newcastle.edu.au

P. Moscato, N. J. de Vries (eds.), *Business and Consumer Analytics: New Ideas*,
https://doi.org/10.1007/978-3-030-06222-4_2

Marketing has become one of the most multi-faceted disciplines, both in academia and in the professional world and it evolves as fast as technology, trends and our surrounding world does. Although it is ever-changing, some basic fundamentals of Marketing still hold truth today and help understand the background of the field, Business and the "why" and "how" of consumer behaviour. We will cover the general background of Marketing as a discipline and slowly bring it forward to today, where business researchers and professionals increasingly rely on data analytics for success, and where data scientists increasingly find some of the most exciting applications of their technological advancements in the business world. We therefore aim to provide a brief introduction into the world of Marketing, the background marketers come from when engaging with data scientist and analytics experts in our quest to *bridging the gap*.

2.1.1 Marketing as Applied, Consumer-Centric Data-Rich Economics?

With roots in Economics, Psychology, Sociology, Science and more recently Computer Science and Mathematics, Marketing has become an extremely multi-faceted discipline. Much debate has been conducted about marketing as a discipline and the role it should play in organizations and businesses. As the marketing discipline continued to grow, a "push" was seen to bring marketing closer to the "hard sciences", the Science, Technology, Engineering and Mathematics (STEM) subjects. Particularly in the late 1950s to early 1960s marketing literature saw a surge in research advocating "marketing as a quantitative science" discipline [69]. However, the truth is that marketing co-evolved with businesses, organizations and consumers, and has been influenced by these other disciplines. Kumar [69] specifically shows the overarching trends in marketing research and practice from the 1930s, when marketing was first spawned as a "child discipline" of economics, described by Kumar using the metaphor "Marketing as Applied Economics", to the present day of marketing's omnipresence in both business and life in general.

Today, marketing continues to evolve as we continuously try to understand consumers, their behaviour, their reasons and motivations for their actions and subsequently try to serve all of these consumers with goods and services that please them. What marketing is has changed over the years and cannot be answered in a single sentence. Throughout this chapter we aim to give a good overview of the area, how marketing and business units intertwine with data analytics in today's world while also providing the necessary basics and fundamentals.

With the world-wide adoption of modern information and computing technologies and the ever increasingly interconnected world, the role of marketing in businesses is now evolving at an accelerated pace. One quote from Stephen Brown in 1996, that is still very relevant to us today and in the coming years, goes as follows:

> Surely, so the argument seems to go, if marketers try hard enough, if we crunch ever-larger data sets through our ever-faster computing facilities and develop ever-more sophisticated mathematical models, we will eventually break through to the bright uplands of absolute marketing understanding [11, p. 256].

In today's data-rich world, this statement becomes all the more pressing because, how do we know we are using our "ever-more sophisticated mathematical models" or the "ever-faster computing facilities" to truly generate new knowledge and actionable insights from the hordes of data produced by everything we do and touch in our completely inter-connected world? Will we ever truly achieve "absolute marketing understanding"? Can we compare the understanding of consumer behaviour to that of the behaviour of biological systems? Probably not, but the STEM subjects have now actually been completely intertwined with business and marketing applications as part of the never-ending plight to "marketing understanding". Marketing to, and serving today's consumers successfully, requires an understanding of many aspects of scientific modelling, technological applications and mathematical foundations for data analytics.

One of the reasons for this is that consumer behaviour is becoming more and more complex with the continued adoption and penetration of new technologies, communication and purchase channels and business models [34]. Online search engines, "search engine aggregation" sites, price and product comparison sites, social networking platforms, online stores, online music and tv-subscription sites, mobile phone providers, internet providers and any company with a database store every bit of information about their consumers that they can get their hands on. Unfortunately, a lot of this information is not used to its full potential as it sits unused in endless amounts of databases or is not used in conjunction with relevant data where true benefit can be derived from it.

One consumer, in a matter of minutes, can generate gigabytes of data that are full of potentially fruitful information about that consumer. For instance, while watching a subscription online television show that same consumer is browsing an online clothing store on their laptop, skipping through songs on a recommended music playlist service, receiving instant messages on their smartphone while their wearable device keeps track of their vital health stats. Separately, the datasets that are generated from these behaviours may not provide any of the service providers with a comprehensive understanding of that consumer. However, combined with his usual browsing, listening and online shopping behaviours, the behaviours of consumers similar to him and the inter-platform data available, the information may reveal that this person is an action-movie addict, a rocker and tends to use messaging services most between certain times of day and at various intervals.

This information can be used to target that consumers' next movie recommendation, song recommendation or better targeted advertisements for the online store through his instant messaging app at the time he is actually using it (rather than untargeted ads that are irrelevant to this consumer). This in turn would cause the consumer's experience of using the online platforms and recommender sites to be more pleasurable and efficient which makes it likely for that consumer to continue to use those services. The world is now moving towards one seamless experience between all devices and even between the "real-world" and the "online-world". In fact, it has been found that service quality improves with personalization and a customer's lifetime value (CLV) increases with more personalized and higher quality service [88]. We can logically imagine how much more important this is when we are connected at all times through a multitude of devices throughout our daily lives, tracking, saving and recommending almost everything we buy, do or share.

Furthermore, if targeted advertisements to this consumer using various websites are more accurate, the ads are less likely to be received as bothersome. It is for these reasons that consumer analytics has gained traction among many companies and even new business models have flourished because of this phenomenon. Companies are trying to come to grips with how to successfully derive use from the potentially very valuable information they have about their consumers and how to "monetize" their new online business models in various ways rather than the plain selling of advertisement space on their pages. Scholarly research aiming to better understand our world and the people in it is scrambling to keep up with the new technologies being generated at today's accelerating pace. Further, staying on top of knowing how to make sense of all the new information that is generated by new technologies continues to be relevant in the online world of today and the future.

In order to deal with these large amounts and different types of data coming from many sources (also referred to as Big Data), a shift has already started to occur from having a focus on fitting models to theorized ideas, to framing theories from data-driven research [34]. A multitude of strategies are now employed by organizations and marketing researchers to understand consumers to better serve their customers and ultimately, for companies to become more profitable. Although this interconnected world looks very different now than it did 50 years ago, the basics of marketing, consumer behaviour, business-consumer relationships and strategies are still valid today. This chapter provides you with a fundamental understanding of these basics concepts covering the topics of market segmentation and targeting, consumer behaviour and specifically online consumer behaviour that is relevant to business and consumer analytics.

2.1.2 *The New Bridge Between Computer Science and Marketing*

Some top scientists are recognizing the key role of computer science in the twenty-first century. It all comes, hand in hand, with the increasing central role of mathematics as an ordering force aimed at "quantifying" many other disciplines. As stated, Marketing evolves as people evolve, which means the "marriage" between Computer Science and Marketing is here to stay, and to make a difference. A few years ago, Michael Mackey, President of the Society for Mathematical Biology, was quoted saying:

> The conversion of biology into a more quantifiable science will continue to the extent that it might even become the main driving force behind innovation and development in mathematics.... For many years the inspiration for innovation in applied mathematics has come from physics, but in my opinion, in this century it will come from the biological sciences, broadly defined [57].

In some sense, this is already a trend that we also observe between marketing and computer science, and it is perhaps much more prominent and clear. It is increasingly common to see marketing scenarios providing the source of inspiration for the development of new problems. As we have explained in the previous chapter, computer scientists *thrive* on new problems. The identification of a clearly-cut data-rich domain for which no algorithm exists that satisfies the required need attracts their attention.

The trend is not new. In the early 1990s the attention of many computer scientists was attracted as a "new type of problem" was being introduced. For these problems, decisions must be made, and in some cases resources must be allocated, even if we do not completely know what the future holds [64]. The emergence of digital libraries [91] and video-on-demand [1] were motivating this area. This means that we could interpret this situation "as if" the input of the problem is not completely known in advance and only one part of it is known at the time of making decisions. Computer scientists define *online algorithms* as the methods that they have developed for these problems. They have also defined a measure called the *competitive ratio*: the worst-case ratio between the cost of applying the online algorithm (when only part of the input is known but decisions need to be made) and the one of the *offline* algorithm (a hypothetical situation in which the entire input is known in advance and an algorithm can make optimal decisions with perfect knowledge). This lead to the area of *online algorithms* and a fruitful dialogue between disciplines. Diverse applications now exist in the areas of business and consumer analytics: story scheduling in web advertising [2], display ad allocation [35], resource allocation under discounts in cloud networks [56], online pricing with impatient bidders [22], real-time optimization of personalized assortments [39] and large scale charging of electric vehicles [17] just to mention a few examples. The dialogue between computer science, marketing, business and consumer analytics is stronger than ever before and it is likely that this trend will exponentially increase with the possibilities of personalization of services and products.

In fact, returning to the second part of the quote cited above [57], it is, perhaps, a very narrow view of the world today. By the early 1980s, with the Personal Computing revolution, more and more data processing power has been coming towards the marketing, business and consumer analytics area. It is now clear that, with each of us having in our pockets what was easily considered a supercomputer in the 1980s, the time has come that these fields are going to be an inspiring muse for applied mathematics and computer science. It is probably safe to say that, when it comes to the "marriage" of Marketing and Computer Science, we are only just at the beginning!

2.1.3 The Blurring Lines Between Products and Services

To continue the dialogue of presenting common trends in business and marketing today, we talk about services. Services heavily dominate our economies. Almost everything we do online as consumers is part of us making use of a service. May it be shopping online, reading publicly available encyclopaedia information, chatting with friends or searching to book a holiday, they all mean we are using a service one way or another. Even before the widespread use of the internet, services made up a huge part of consumers' purchases (think for instance, visiting a doctor, hairdresser, going to a restaurant or a holiday). Furthermore, services have also become a lot more prominent in Business-to-Business (B2B) transactions and relationships. Companies are outsourcing business aspects such as finance, accounting, data analytics, storage and management among many others to service firms at greater levels than ever before. All of this means that services dominate a great part of the research and knowledge development in business and marketing fields. Due to the different natures of goods and services, a lot of research and theories have been conducted and developed to deal with them separately. However, the line between goods and services has continuously blurred in recent years. For example, would we call an app a good or a service? It is not physically tangible but it also is not quite a service that only is transferred from human to human. A mobile app is programmed and coded and subsequently this software is downloaded by a user. What about an online magazine or personalized training plan subscription? We cannot hold the actual software behind the app in our hands, like we can with a "typical" consumer good such as a carton of milk, however, it also does not classify as a "traditional" service such as a restaurant experience or a financial service.

In order to understand the complications of blurring lines between goods and services, we briefly cover the basic characteristics, theories and ideas about services, services marketing, how they differ from goods and how the marketing of each typically differs.

2.1.3.1 Services Marketing Fundamentals

We traditionally split products into two categories: goods and services. That means, physical products we can hold, and services provided to us that add a benefit (or *value*) to our lives. The difference between the two was described by four key attributes: tangibility, inseparability, heterogeneity and perish-ability [74]. In a pre-internet age, these four principles made a lot of sense and applied to almost every service. For example, being served in a restaurant is not a tangible object you can hold, for the service is *inseparable* from the moment you are consuming it and inseparable from the service provider. An experience in a restaurant is *heterogeneous*, as you may have a different experience every time you go into a restaurant, depending on the day, the different person serving you, etc. Finally, it is *perishable* as you cannot "save" it and consume the experience at a later time.

However, as goods and services have become increasingly intertwined, these four principles may not always be true nowadays. Today, services are “heterogeneous” on purpose. Meaning, they are personalized to your benefit, made to suit you and to generate as much *value* as possible for you personally. Further, some of them are not really perishable. For example, your mobile smartphone, your smart television device or simply your computer can remember exactly where you were up to in your latest Netflix “binge” or music tracking playlist.

In response to this, more recently, Lovelock and Gummesson have introduced a different view of services which posits that “services offer benefits through access or temporary possession, instead of ownership, with payments taking the form of rentals or access fees” [73]. This type of view is more generally applicable in today’s online world. For example, temporary online movie rentals, book rentals through e-Reader devices, online music and tv-show streaming services all fall under this way of viewing “services”.

Although the lines between “products” and “services” have blurred, there are still several characteristics of services that make them different from products that are relevant today, some of them highlighted by Lovelock and Gummesson in a more recent work [74]. Services cannot be inventoried. This has an impact on businesses and strategic marketing managers for forecasting demand. In terms of the online economy, this means, for example, enough computing power needs to be made available for enough customers to be able to use the software at once. For instance, if a website crashes when too many people are trying to use it one time, the company will likely have many dissatisfied customers. Likewise, if a company has already made surplus computing power or data storage space available for what they thought would be a high-demand time, those resources could be wasted resulting in financial losses for the firm.

The aspects that cannot be touched, seen or visualized are usually what creates value when it comes to services. These aspects are what impacts on consumers’ decision making processes, a company’s product and service development teams or the way it is marketed. This is one reason why customer online reviews, and other forms of Customer-to-Customer (C2C) interaction (see, e.g., Sect. 2.3.1), have exploded in popularity in the past decade. Reviews and consumer’s experience statements are nothing new in marketing and advertising. However, the level of online reviews available to consumers today is unprecedented. We all know *TripAdvisor*[1] or similar customer review websites reviewing anything ranging from travel services and restaurants, to hairdressers, transportation companies, video games, or even doctors and other professionals’ services. It is these kinds of online behaviours that have now become part of the customer service experience in many industries and for endless amounts of services (both online and offline).

[1]https://www.tripadvisor.com.

2.2 From Direct Marketing to Market Segmentation and Targeting Strategies and Back Again

Marketing and consumer behaviour textbooks have typically taught us that market segmentation and targeted marketing strategies are necessary for a successful relationship with your consumers. They have also taught us that "direct marketing" is usually too expensive and not usually a viable alternative. However, with almost every consumer in developed countries having access to the internet in the palm of their hands or on their wrist (or in the near future, maybe even under our skin!) and the capability of communicating with businesses and fellow consumers, this has changed. In the mobile and online market place, direct marketing and advertising is very real, is happening in real-time and is saving consumers a lot of time while making businesses more money.

However, before we discuss today's situation, we take a brief history lesson about segmentation in the marketing discipline and how and when market segmentation came about. Targeted marketing strategies are explained and their benefits to businesses and how targeting to customers is done now by online companies and brands. After, we return to today's setting of direct marketing capabilities, unprecedented levels of personalization and even prediction.

2.2.1 A Little History to Market Segmentation

Market segmentation is the process of the market being divided into smaller groups of consumers with similar wants, needs and characteristics and who may require different goods and services or alternative marketing and advertising strategies [40]. Market segmentation emerged in the literature and in business in the 1950s (see, for instance [26, 93] and [33]). Marketing as its own discipline started to gain more traction and marketers wanted to gain better insights into their customers. In fact, Smith [93] was one of the first to put market segmentation forward (in 1956) and it has since been defined in many different ways.

Although it may not previously have had a name or label, market segmentation has existed for as long as businesses and suppliers have used different methods to gain clients. However, in the last five decades, marketers have been able to segment better, thanks to the development of new economic theories and sophisticated analytical methods [28]. In the early 1960s, business and marketing literature began referring to market segmentation as a new actual business "strategy" as it gained more popularity and attention among scholars and professional marketers [10, 38, 76, 87]. Companies had realized that mass marketing (think: Ford's Model T in the 1920s) was not as successful as when Henry Ford first started selling his black Model T's through mass production. Marketers began to understand that each individual consumer is different and expects to be served according to their own needs. Scholars started analysing, discussing and explaining various approaches to segment a market. People have different purchase behaviours because they have different tastes, different ways of expressing themselves and varying expectations from businesses and brands. A successful marketer has to incorporate all of this into their market segmentation strategy. As Roberts [87] states, market segmentation could be described as a strategy with the philosophy of "something for everybody".

However, it is exactly this "something for everybody" philosophy that is hard to achieve in real applicable market segmentation. Finding out how to define market segments is an aspect of market segmentation with the greatest applicable consequences [5]. The many variables to choose from make market segmentation no easy task [41]. The purpose of market segmentation is to target the segments that are found individually with an altered or specialized marketing and advertising strategy. Therefore, the way we segment is going to decide what promotional material and communication the consumers are going to receive from the company or even the next line of products a company may develop and manufacture or the kinds of services it will add to its offerings [60]. An extensive work on Market Segmentation can be found in Wedel and Kamakura [104] however, for an understanding of the market segmentation bases, we will cover in summary the most well-known bases of market segmentation.

2.2.2 Different Ways of Cutting the Consumer Pie

We cannot market the same thing in the same way and at the same time to all consumers. This is why the consumer market needs to be split up (segmented) in order for marketers to target each segment. To describe the different bases of segmentation here, we have taken the broad categories outlined previously by Schiffman et al. [89]. In turn, these segmentation approaches are highly similar to earlier works such as Beane and Ennis' market segmentation review in the late 1980s [7] and many other market segmentation works. Both of these works include the following bases for segmentation: geographic, demographic, psychological, psychographic, sociocultural, user-related, user-situation and benefit segmentation as well as possible hybrid segmentation methods.

2.2.2.1 Geographic Segmentation

The key idea behind geographic segmentation speaks for itself. It has been happening for as long as people have exchanged goods and services. When starting any sort of transaction, for many years, the only possibility has been to exchange locally or within reasonable travel time. Before the advent of the internet and e-commerce, businesses had no choice but to geographically segment their market. Although this has changed in today's online society, geographic segmentation still exists and is still practised by businesses in conjunction with other segmentation criteria. When segmenting geographically, factors such as housing density, climate (weather) and relating factors still need to be taken into account [89].

2.2.2.2 Demographic Segmentation

As the name suggests, demographic segmentation is a partition of the market based on a consumer's specific demographic information, such as their age, sex, marital status, income, education level and occupation [89]. This is also one way of segmenting that is age-old and has happened even before scholars coined the term "market segmentation". Any product that is specifically for males or females or for children or adults has used the concept of demographic market segmentation. Furthermore, a lot of research is done on the behavioural and spending patterns of consumers with different levels of incomes.

2.2.2.3 Psychological Segmentation

From the surface, psychological segmentation is difficult to define. It refers to the inner or psychological characteristics of consumers, their intrinsic qualities [89]. Endless amounts of qualities could be associated to any given person which makes it extremely difficult to drill down to several qualities that are able to segment the complete market. Some of these qualities include consumers' motivations, personality, perceptions, learning, level of involvement or their attitudes. As market segmentation theory continued to evolve, marketers wanted to "dig deeper" into the minds of their consumers. Along this line, psychological segmentation surfaced as a way to define, describe and segment the market in a way that geographic or demographic variables could not.

This "new" and exciting alternative approach to segment the market was promised to "shake up" market segmentation was traditionally conducted. However, although psychological constructs are "rich" in information, they are often unreliable when investigating large groups of people [85]. In order to still use psychologically-related variables, marketers have come with "life-style analysis" which has proven to be a useful marketing tool. Lifestyle analysis is part of psychographic segmentation [89].

2.2.2.4 Psychographic Segmentation

Psychographic segmentation is also referred to as *lifestyle segmentation*. The concept of consumer lifestyle patterns was first introduced to marketing by Lazer in 1963 and in the later 1960s and the early 1970s became more popularly used as a basis for market segmentation [85]. As Plummer [85] states, some of the most commonly used lifestyle segmentation approaches back then were activities, interests and opinions. Another argument for psychological segmentation is that it may be more generalizable than demographic or geographic segmentation as the segments based on psychological characteristics could exist in multiple markets [70]. Although it seems like a logical way to segment the market, it has been difficult to apply these variables to large groups of people. However, nowadays with the widespread use of the internet by almost all customers and business, gathering information about these aspects has become more feasible. More on this later.

2.2.2.5 Sociocultural Segmentation

Sociocultural segmentation, as its name suggests, takes a cultural view of segmenting the market. Examples of such characteristics include stage of family life cycle, social class, core cultural values, subcultural memberships and cross-cultural affiliations [89]. As the family life cycle progresses, it is natural that the household will require different products and services (think single person to married couple with kids, from babies to toddlers to teenagers and so forth).

The other part of sociocultural segmentation is the cultural characterization of a consumer. Consumers' values and attitudes have been found to have an effect on their purchase behaviour and other consumer behaviours towards brands. As Vinson et al. [101] explained as early as in 1977, a person's attitudes and values are affected by their sociocultural surroundings. It is for this reason that cultural characteristics make an interesting segmentation base that provides rich information for segmentation purposes. However, again, its downfall is that it is difficult to apply to large groups of people due to its lacking generalizability and obvious concerns for possible discrimination occurring when segmenting based on social status or cultural backgrounds.

2.2.2.6 User-Related Segmentation

Rate-of-usage segmentation takes into account how much a consumer purchases or uses a product or service and a segmentation is made accordingly [89]. It has also been referred to as usage-rate segmentation [7]. Beane and Ennis [7] explain that usage rate segmentation may divide consumers into light-, medium- and heavy user groups. Marketers then generate separate marketing and targeting campaigns for each. One common example of this is insurance companies who give discounts for "less usage". This gives customers the feeling that the services are more personalized to them and that they have more control over the products they choose. This type of segmentation, where applicable, has now become extremely common and is intertwined with many other segmentation and targeting strategies.

2.2.2.7 Benefit Segmentation

Benefit segmentation can also be called "needs-based segmentation" as it refers to the actual need the product or service satisfies. It is also part of "behaviouristic segmentation" as covered by Beane and Ennis [7] and earlier by Kotler [67]. Companies have been using it since 1961; however, research about benefit segmentation lagged behind. Haley [50] describes it as an alternative, and more optimal base for segmenting the market. As he states, the belief that underlies this strategy is that the reason *why* a consumer chooses a product (or the benefits that they seek) are the true underlying reasons for the existence of market segments.

With benefit segmentation, the idea is to look at the segments that surface after consumers are divided into groups of similar benefits and needs followed by an examination of their demographic information, volume of purchase and other aspects. In doing so, a richer image can be created of the consumers. This strategy is still relevant today. For any goods or service provider it is important to understand the benefits their consumers are seeking from their products and the needs they want to have satisfied. When these questions are answered, part of the question of *why* that consumer has chosen to purchase that particular good or service is known.

2.2.2.8 Hybrid Segmentation

The last segmentation basis that Schiffman et al. [89] discuss is that of hybrid segmentation. In Beane and Ennis [7] several mixtures of segmentation approaches are also discussed. For example, "componential segmentation", which combines situational variables with consumer/respondent characteristics. Hybrid segmentation in general can be a mixture of some or all of the above-mentioned segmentation bases. This type of strategy is closest to what companies have done in real-life by combining several bases for segmentation to segment the market more accurately and target more efficiently.

2.2.3 Targeted Marketing and Advertising Campaigns

As stated, the main driver for segmenting the market is then to target each segment with a varying strategy. With the continued technological developments available, businesses have been able to implement narrower targeting strategies to target various consumers segments more precisely. Generally, even after the market is segmented, a lot of communication material or promotional content will reach people that it is not intended for, or who are not interested in it. For example, television or radio advertisements where one ad is shown to a large group of segmented people (for example, a cooking appliance ad during a cooking show), could reach a large section of that group may not be interested in purchasing a cooking appliance any time soon. With the increased online information available about consumers, targeted marketing and advertising campaigns can be made more narrow and precise resulting in more successful marketing strategies.

Furthermore, scholars and marketers now understand that we cannot treat all consumers in even the same segment the same. Consumers that have been segmented based on certain geographic, demographic or even psychographic criteria may have vastly different customer behaviours towards brands. High levels of heterogeneity even exist when examining consumers' online behaviours towards brands within customers of the same brand [23]. This shows to brands that, although they have a certain brand identity that they try to target consumers with, who themselves identify with that "brand personality", they cannot treat all these consumers in the same way. This means that we need to look at consumer behaviours (See Sect. 2.3 on Consumer Behaviour) when segmenting and targeting the consumer market and go beyond the "traditional" segmentation bases and even beyond known targeting strategies. This is where modern-day personalization comes into play.

2.2.4 "Back Again": Direct Marketing and Personalization

Personalization to and direct marketing at the individual level are now a reality. Although market segmentation and targeting strategies are not completely obsolete, they are now complemented by highly effective and detailed personalization analytics and artificial intelligence methods.

2.2.4.1 Direct Marketing Made Possible Through Technology

In some sense the direction for personalization of offers has made "direct marketing" more viable. However, this is not a new concept. Direct marketing is a form of advertising that has a strong focus on the customer and employs data and testing strategies. Its defining characteristics are that: (a) the messages are directly addressed to customers and/or prospective customers, (b) there is a "call for action" (it used to be a call free phone number or mail order, now it could be a click on a website) and; (c) there is an emphasis in quantifiable/measurable responses and in monitoring response rates (and objectively tries to minimize future advertising expenses while maximizing the response rates). It is a strong data-driven procedure in which pre- and post-campaign data analytic procedures are used.

This "learning-from-data" approach is certainly employed by all types of businesses nowadays. Companies like Amway, Avon, Herbalife, Markay, Vorwerk and Natura are leaders in total sales (in US dollars) using this approach. The total revenue of the top 100 companies is approximately 81 Billion US dollars in 2015.[2] Furthermore, direct marketing has also been central to the activities of many charities and non-for-profit organizations as a strategy to address people directly with compassionate messages [16].

It is reasonable to assume that the data analytics methods that were normally employed in direct marketing were heavily guided by "simple" statistics. Interventions in the marketing campaign were designed and the outcome evaluated *after* the fact. With the advancement of new technologies and computer science methods, in particular machine learning, we are now observing some changes. Quantitative

[2]https://www.businessforhome.org/2015/04/top-100-global-direct-selling-companies-2015/.

direct marketing models are now statistical and/or machine learning based [9]. Heuristic and metaheuristic techniques are being employed due to the large size of the datasets [82]. Marketers and advertisers now aim to take a more predictive approach in their strategies. This is possible due to the large sizes of datasets marketers now have access to.

Besides large datasets, what brings real value to marketers, researchers and business decision makers is the high variety and velocity of datasets and the combination of these sources of available information. Consumers' purchase and personal data has become extremely valuable and opinions of its use are highly contrasting. The next section covers the personal data use trend and aspects.

2.2.4.2 Personalization and Monetization in the Personal Data Business

By now, most consumers in the developed world will have used some form of product or service that is the result of today's great technological advancements making high levels of personalization possible. Think for example, Netflix, Spotify, an online booking system and so on. All of these services run on algorithms that include recommender systems, clustering, prediction and more in order to personalize the service to your experience. Furthermore, even if you have not subscribed to any of these services, but may have simply searched the internet, used a social media networking site or bought something online as a once-off customer, you have likely still been a consumer of a personalization strategy. Furthermore, as stated above, it is almost certain that the personal information that you have provided in these online activities and transactions has been recorded by one or multiple companies and stored for future marketing or advertising purposes.

2.2.4.3 Personal Data: Online Advertising

In recent years, data and (consumer) information has popularly been referred to as the "oil of the twenty-first century" (and some say this analogy has become a bit of a cliché). Having and being able to store, analyse and make use of extremely large amounts of business and personal data is now seen as extremely valuable. Not only for the company from which the data came from, but for many third-party companies or advertising companies.

One of the main and first reasons for this current personal data use and monetization "wave" is online advertising. Whole business models are now based on the notion of making profits from advertisements. Online advertisements are at the forefront of aiming to predict a consumer's next choice in order to make financial gains through their pay-per-click strategies, for example. Naturally, the more information available, the better the advertisements can be predicted and the more successful they are likely to be. One of the world's most popular social and digital companies, *Facebook*, has as its main "business" advertising. In fact, 2014, 2013 and 2012 advertising accounted for 92%, 89% and 84%, respectively, of Facebook's revenues [3]. Furthermore, Google in 2014 reported $66b in total earnings, of which $59.62b came from advertising alone.[3] Since 2014 these figures are likely to only have grown and they provide a good indication of the size and value of the online advertising industry.

2.2.4.4 Personal Data: Data Management

Advertising is not the only reason why companies collect and store our online (and offline) movements. Other ways in which personal data is used and monetized are for marketing and product development activities. For example, knowing how customers are actually using your online store, service or application can provide managers with useful insights that can feed into the product design stages and general marketing strategies of the company. If we take a popular social networking company like Instagram. Instagram actually started out as a company for consumers to "check-in" locations, share plans, information as well as photos. That is, photo sharing was only one feature of the first app. However, after learning how consumers were actually using their app, it became apparent that the photo-sharing feature was the most popular even though location-based applications were "all the rage" at this time.[4] Subsequently, they re-branded, changed their actual service and became what Instagram is today. Without the consumer insights coming from their usage and behavioural data, this would not have been possible.

More recently, extremely large amounts of data have become valuable for data management purposes. With the large deluge of data that companies and organizations find themselves with, having the capability and know-how of how to store and manage these amounts of data safely also becomes a valuable asset. Cloud computing companies' business model is to generate profit from providing storage space and computing power to businesses on an on-demand basis. In recent years, these types of services have become much more prevalent as the demand for them is ever-increasing. For many companies it simply is not feasible to store and manage the large amounts of data themselves as they do not have the resources available. This is where the value lies for storing and managing big data.

2.2.4.5 Personal Data: Ethics and Privacy

As expected, huge debates regarding the ethical implications of companies using, storing, analysing and selling consumers' personal data are happening everywhere. Consumers are increasingly more concerned with the safe storage and different uses of their personal data [20]. In the personal data debate we have two sides. On one

[3]https://investor.google.com/financial/tables.html.

[4]http://www.investopedia.com/articles/investing/102615/story-instagram-rise-1-photo0sharing-app.asp.

side are consumers who want to protect their privacy while being able to make use of the latest and greatest technologies, while on the other side we have companies who want to be able to make a large as possible profit from providing these technologies and doing so requires the use of their consumers' personal data.

Further, as the Accenture report explains [20], privacy is an inherently personal concept and a "one-size-fits-all" approach is not going to work in the prospect of protecting privacy in today's changing technological environment.

The Ethics of Personal Data: A Hot Debate
A somewhat recent example that was the topic of a "hot debate" about the use of personal data comes from the messaging "app" *Whatsapp*. After being bought by Facebook, Facebook's promise to consumers was that conversations within the Whatsapp "app" would remain private and would not be used for Facebook's personalized advertising campaigns. However, in August 2016, Whatsapp[5] announced a change in terms and conditions when users downloaded a new update which in fact would share personal information with Facebook unless consumers opted out before a specific month. Many users were angry and frustrated as Facebook Executives had previously proclaimed that "nothing will change" and that Whatsapp will remain independent when Facebook bought out the popular messaging service in 2014.[6] Many online bloggers and users of the app expressed their frustrations online meaning Whatsapp's PR team probably had some very hard times. This example shows how companies must take making promises to its users about the use of personal data very serious. We are yet to see the end of the story as the legal ramifications of these decisions continue.[7]

From a legal point of view, it is also important to see some progress in terms of how online communications, transactions and transfers of information and data are viewed. As Hasty [51] explains, digital interactions could (should) be framed as commercial exchanges of value in the eyes of the law. Many ethical and legal changes are going to be necessary with the ever-changing trends in the online personalization, advertising and information sharing environment. One main trend that we are seeing now is that consumers are increasingly seeking a data dividend as they are realizing the potential monetary value of their own data. Some call this the "second wave".

[5]https://www.whatsapp.com/.

[6]https://techcrunch.com/2014/02/19/facebook-buying-whatsapp-for-16b-in-cash-and-stock-plus-3b-in-rsus/.

[7]https://www.reuters.com/article/us-eu-dataprotection-whatsapp/facebooks-eu-regulator-says-whatsapp-yet-to-resolve-data-sharing-issue-idUSKCN1GC2H8.

2.2.4.6 Personal Data: "The Second Wave"

We are now on the cusp of the monetization trend changing to its second "wave". Consumers and businesses are each coming up with new alternatives to solve the "privacy debate" changing the course of personal data monetization. Besides the practical and economic motivations, the notion of giving consumers back the power over their own data is receiving increasing attention from various communities due to the many ethical and privacy-related issues with using personal data. As Peter Sondergaard, the Senior Vice President at Gartner Research explains, in this second wave, consumers will be enabled and empowered to own and thereby monetize their own data. Meaning that they can take back the control and drive up value for themselves.[8] Sondergaard predicted this trend in 2013, however now, this is actually happening.

In a recent report, Accenture found that 60% of their survey respondents reported to having engaged in activities to monetize their own data [20] showing that this is a growing trend. In return for this growing demand, companies are starting up providing exactly this service to consumers. For instance, *People.io* is a platform that allows consumers to sign up and benefit from "giving away" their personal data. The founders of *People.io* explain that one of the main motivations for this platform, and one of the main reasons for its (potential) success is the extremely high increase in use of ad blockers.[9] It is important to highlight the main difference between a consumer signing up through a platform like this, and simply "giving away" data to the companies they are already purchasing from or receiving discounts in return for information. The difference is that this platform will *license* the consumers' data, time and time again. A once-off provision of your personal information to a web-based company may only prove to be worth 50 cents (maybe even less depending how valuable of an eCommerce customer they deem you to be). However, the value lies in this information being used time after time aggregated with information from more sources and combined with information of other consumers. This is how online personalization advertising companies currently generate a profit. It is for this reason that letting consumers "licence" their own data out in a similar fashion, it would actually prove beneficial for those consumers.

The dialogue (and the online power play) will now continue to find an interaction between companies and consumers that will aim at creating a mutually beneficial outcome for all. One where consumers' do not feel like their privacy is intruded, but one where each individual determines their level of data sharing or "licensing" and where personalization does not equal privacy "invasion". Daniel Newman states the following in a Forbes Magazine article which, perhaps, states this next stage (possibly the "third wave"?) of online personalization:

[8]http://blogs.gartner.com/peter-sondergaard/monetizing-personal-big-data-second-stage-of-digitalization/.

[9]http://econsultancy.com/blog/67441-start-me-up-people-io-allows-people-to-monetize-their-personal-data/.

> Personalized, data-driven marketing will become more refined. There is a difference between data-driven marketing and intrusive marketing. While the former is based on relationship-building, the latter is nothing but old-school push marketing wrapped in a new cover. The difference between these two formats will become even more prominent in future. Marketers who focus on relationship building will be rewarded, while intruders will be shut out.[10]

2.3 Consumer Behaviour and Interaction

Consumers' varied and heterogeneous behaviour is one of the main drivers for market segmentation, targeting campaigns, direct marketing and personalization strategies [25]. Consumer behaviour is defined as the behaviour that consumers display (towards a brand or business) in searching for, purchasing, using, evaluating and even disposing of the products and services that they expect will satisfy their needs [89]. The key idea to take from this definition is that consumer behaviour does not necessarily mean purchase behaviour. It can be any display towards a brand in the form of an online comment or "like", sharing something to friends, blogging about a certain product or service, rating and reviewing services online and so forth. The current online nature of interactions between consumers and businesses has meant that it is possible to track and analyse a lot more non-purchase behaviour of consumers. At the same time, it has also amplified the effects that non-purchase behaviour can have on a company's success (both positive and negative). When we talk about consumer behaviour prediction, online recommendations and personalized advertisements, it all centres on being able to understand consumer behaviours. This is why we bring back some marketing fundamentals and present the broad topic of consumer behaviour and interaction.

2.3.1 Changing the Way We Communicate with Consumers

Customers can *engage* with brands online and display many different behaviours towards their brands of choice without ever having to purchase anything [24, 100]. Historically, businesses and organizations only communicated to consumers and consumers were always the receivers of communication messages, advertisements or promotional material. With the continued development of the marketing discipline among scholars and the continued acceptance by businesses of marketing as a "true" business unit, a trend was heralded by prominent scholars such as Kotler who continued some of the earlier views that Drucker held about the role of marketing in businesses and economies [31]. Marketing started moving away from

[10]http://www.forbes.com/sites/danielnewman/2015/04/14/10-top-trends-driving-the-future-of-marketing/2/.

a heavy emphasis on price and distribution and move to a greater focus on meeting customers' needs and on the benefits received from a product or service. Kotler [67] was one of the first (in 1965) to recognize that marketers needed to "dig deeper" to generate a better understanding of their consumers. All of the more advanced segmentation, targeting and personalization methods that we have discussed would not be possible without a sound understanding of consumer behaviour. This trend continues to evolve as the society and economy we live and operate in continues to be "disrupted" by newer, faster, better and more all-encompassing technologies.

While Drucker and Kotler were looking at purchase behaviour of consumers at physical stores and service outlets, we now deal with consumers who can be interconnected 24/7 and are able to make purchases at any time that suits them. There are many more possibilities of what constitutes as *consumer behaviour* towards brands than in Drucker and Kotler's time. Whereas previously, the only real behaviour marketers could analyse was purchase behaviour. Now, there could be years of information on and interaction with a consumer, without that consumer in fact having purchased anything from that brand. A popular joke these days on the Internet goes like this:

> A million guys walk into a Silicon Valley bar. No one buys anything. Bar declared massive success.

Another example of non-purchase consumer behaviour is a motor enthusiast who may follow everything a particular auto brand does and interacts with them online but cannot afford to purchase an old-timer classic model themselves. They can follow, like, comment, interact with and share any information of and with this brand, be an advocate for this brand and have a significant impact on the brand's online communication success without being a customer themselves. This "online playing field" brings a whole new area of both challenges and opportunities to marketers. Behavioural data has become a gold mine of information for those users that know how to extract it. We now need to see consumers as partners in a lasting relationship, and serve them accordingly. This book will feature many different aspects of analysing the purchase cycle of consumers or the business product development cycle and its related behaviours. Further, we now do not only look at communications and behaviours between consumers and companies, but also among consumers. Customer-to-customer (C2C) interactions have become a popular area of research for marketing and business practitioners and researchers.

2.3.2 Customer-to-Customer Interactions

It is obvious that with the widespread adoption of new technologies such as the internet, smartphones, tablets, wearable internet-connected devices and others, communication has changed. Typical business-to-consumer (B2C) communication still happens very frequently, as well as business-to business (B2B) communication; however, consumer-to-business (C2B) and consumer-to-consumer (C2C) communication have become just as common. Whereas in the past, a dissatisfied (or an extremely delighted) customer had to go through the effort of returning to the store, service outlet or organization location, or send a letter via the post to express how they feel, nowadays, leaving a review, a recommendation, complaint, compliment or a "thumbs up" or "thumbs down" are only a few clicks (or swipes) of effort away and can easily be seen by hundreds or thousands of other potential customers. Specifically, the C2C form of communication has potential to provide businesses with actionable insights if used effectively. Further, C2C interactions can also be represented by networks when these consumers are connected in some form, for instance, a Twitter follower network, or a Facebook friendship network. As you will find in this book, a network can be investigated, explored and analysed in a multitude of ways to provide meaningful insights to business decision makers and marketing managers. Specifically, Part III focusses on network analytics applications. As part of looking at C2C interactions, user-generated content and word-of-mouth communications are highly important.

2.3.2.1 User-Generated Content and Word-of-Mouth

C2C interactions and communication are extremely important to the success of a business as they can affect growth and profitability of a business as well as its reputation [72]. This also means that marketers and researchers now have access to more information about consumers than ever before. Online review platforms, discussion and help forums and online blogging platforms are rich sources of information where consumers often "speak their mind". Consumers in fact generate a lot of content whether it is through reviewing service providers such as restaurants or by simply uploading photos of their experience. These types of behaviours all become part of the C2C interactions that could affect the success of a business depending on the nature of the content being uploaded or the consumers' experience which they share with fellow internet users.

Consumers are more likely to trust communication about a brand coming from a fellow consumer than from the brand or business itself. Although WoM research is nothing new, the speed, levels of interactivity and times at which C2C interactions now happen are. Most review or booking sites (for example, Tripadvisor) now provide mobile application versions of their service (website) making it even easier for consumers to leave a quick review. Consumers are now also more likely than ever before to actually generate content in the form of photos, videos or blog posts about the service provider they recently used. This provides the business with both genuine content that they can use in their own communication strategies and a real insight into consumers' minds and their opinion of that business or brand. With some clever social media strategies, service providers can "ride" of the back of user-generated content and WoM without having to employ many blog writers themselves or hiring a team of photographers.

When a social media strategy that encourages consumers to take action themselves becomes extremely popular and successful, it has been termed the somewhat cliché “viral marketing” strategy.

Using “Viral Marketing” for Social Good

Online “viral” campaigns can also be used for social good. A great social movement like the “Ice Bucket Challenge” in 2014 would not have been able to occur without the widespread use of social media among so many different demographic groups. Although “The Ice Bucket Challenge” received some negative press about creating a “hype” rather than *actually* helping people with ALS, the movement proved hugely beneficial for ALS charities and their fundraising. According to the (US) national chapter of the ALS Association (ALSA), the challenge brought in a staggering $115 million. Participants also donated an additional $13 million to the association’s regional branches. Normally, these kinds of numbers were unheard of for the ALSA. The charity’s official form filings for 2013 show they brought in $23.5 million that year, meaning that in 2014, there was a national donation increase of nearly 490%! Since then, ALS researchers have actually been able to make a scientific breakthrough as a direct result of research that was made possible due to the large amount of money raised.[11] If that does not prove that successful viral marketing campaigns can make a “real” difference, no campaign will.

As campaigns such as the Ice Bucket Challenge have shown us, consumers can now engage with brands online through any number of technological portals, at anytime and from anywhere [24] at a never before-seen ease of access. In all of these online communications, “likes”, “follows”, “retweets”, “favourites” and so forth show how many options consumers have to interact with each other and with organizations.

The greater use of, and dependence on, social media in communication also means that consumers themselves have become more demanding. When a consumer poses a question or complaint to a brand’s Facebook page, they expect a swift response as this is the nature of online social communications. Gone are the days where you had to wait several days for an email in return or even a slow response on a review website. If you are unhappy with a company’s service, their product, or how long you have been waiting in their telephone queue, you simply find them on social media, “slander” them publicly and they will automatically feel compelled to

[11] https://www.theguardian.com/science/2016/jul/27/how-the-ice-bucket-challenge-led-to-an-als-research-breakthrough.

respond in a timely manner in order to save face. However, due to this quick and easy online nature of C2B and C2C communications, brands can also pleasantly “surprise” their consumers by taking quick action.

Pleasing, or “delighting” customers is all about “managing expectations”. Any first-year (services) marketing textbook will tell you that if a company fails to deliver the customer’s basic expectations, that customer will be dissatisfied (see, for instance, [74]). Oppositely, if a company or service manages to exceed the customer’s expectations, he/she will be delighted. This same theory applies to social media communication. Social media platforms themselves can be seen as a service, or at least a service platform [24]. By this we mean that the basic theories of service marketing also apply when communicating with consumers online. Therefore, if a customer is particularly unhappy and expresses so in an online post or comment, the company now has an opportunity to revert this situation into a positive one. It is likely the customer is not “expecting miracles” from their online communication which means that if the company is able to resolve the issue, the customer’s expectations were exceeded, making them a delighted customer.

It is extremely important for a brand or company to understand their customers (and potential customers), so that they can “delight” them in the time that these consumers spend on the online platforms of the brand. Through today’s advanced analytics capabilities businesses have become more successful in understanding their customers, or target market, and personalize their offerings accordingly.

2.4 Empirical Research in Marketing

As the whole theme of the first two chapters of this book is to *bridge the gap* between Marketing and Computer Science, we cannot leave out a section on the typical existing empirical research conducted by marketing and business researchers. In our search of ever greater “marketing understanding” as Brown has stated in the quote at the start of this chapter, marketers and researchers are moving closer and closer to the exact sciences and use ever increasingly complex mathematics to continually improve market understanding and knowledge. It is due to this that the Marketing discipline so heavily adopted Statistics in its basic workings. It is here that we will find how much Marketing has been influenced by the discipline of Psychology which in turn has had a long-lasting “marriage” with Statistics. It is thanks to statistics that researchers have been able to understand human and consumer behaviours in more depth through methodological analysis of data. Marketing has been influenced and borrowed methods from other disciplines, more recently psychometrics and mathematics [15] and computer science coming in the later years to make a difference.

If we look at the last few decades of economic trends and human developments, we have seen an immense improvement of our understanding of the world around us. Not just the physical world, but also the more nuanced world that is human behaviour and human personalities. In these trends, we have become obsessed with knowing, understanding and even predicting the world around us. From weather forecasts to financial forecasts to predicting whether Amy browsing on her mobile device is going to make the purchase in her shopping basket. As Marketing is a fairly young discipline (compared to its STEM relatives and even Psychology), it has only been around since the human drive for more and greater knowledge has grown to unprecedented levels. This means that marketing and business researchers have aimed at applying the latest and greatest research methodologies and the best statistics to generate a "greater marketing understanding".

As the internet has only been used by a widespread amount of people for the last few decades, prior to this, researchers had to collect data in different, more traditional, manners. Whereas now, companies have gigabytes worth of data on their customers, previously this was not the case. So, if a Telecom company, for instance, wanted to understand its customers intentions and customer satisfaction better, the best way to find out was through survey research. The researchers would put together a carefully selected questionnaire that would comprise of *constructs* that would aim to get a good grasp of the theoretical concepts such as customer satisfaction. Each construct may be made up of a number of variables that do the best job of truly collecting information that reflects the theoretical concept. Surveys would be collected from a large enough number of customers and this data can then be used for statistical analysis to create useful business insights. Hence, most marketing research that you will see conducted in recent decades will be survey-based research.

2.4.1 Using Surveys in Marketing Research

Survey research is most definitely not just a thing of the past. Even though we have a deluge of data available to us these days, sometimes, the specific questions we want to ask cannot be answered with existing available data. This is where survey research still holds an important place in today's market research space. Direct, targeted and specific information can be obtained from consumers (with their given consent) easily, quickly and at a low cost. They also still provide a good starting point for companies to find out initial insights about customers or, on the other hand, in order to dig a little deeper into an issue that a company might be facing (e.g. why are customers leaving the company?). Due to the still many reasons why survey research is still relevant today, we will cover the basics that go into empirically setting up survey research projects. A good study for the interested reader is that by Malhotra and Grover [75] on survey research. They explain that a survey typically has three characteristics: *asking people* for information in an organized format, a *quantitative method* and a *sample* (i.e. only a fraction of the population of interested is investigated) [75]. These characteristics bring with them important considerations. Asking people information in a structured approach can be done in many different ways (offline, face-to-face, email, etc.), the quantitative method used needs to be the appropriate one for the study and for the specific research questions that need to be answered, and finally, it needs to be possible for the findings based upon information from the sample to be generalized for the whole population. In explaining these characteristics and the important considerations for survey research, what we are really saying is that we want to avoid GIGO; as Churchill [19] stated, marketers need to avoid spending too much effort on what computer specialists sometimes call "Garbage-In-Garbage-Out". Hence, following a sound statistical framework when conducting market research is of paramount importance. When a business issue for investigation is decided upon, setting up a survey starts with researchers deciding which constructs are going to represent their business concepts they want to find more about (for instance, customer satisfaction).

2.4.1.1 What Are Constructs?

The key ingredient in creating a survey is a *construct*. In short, constructs are quantifiable measurements that reflect a theoretical concept. The reason we use these is the often lack of quantifiability of business concepts that marketing researchers and professionals want to know more about. In our domain of business and marketing, a common construct we can take as an example is "customer satisfaction". It is a theoretical concept that we cannot easily quantify into measurable variables. It is not possible to count customer satisfaction in particular units of measure or even assign arbitrary scores to it. However, if we generate a set of variables (which could be questions, personal attributes etc.), and if we combine these in such a way that they form a *construct*, then we have something to work with for our statistical analysis approach. Constructs are sometimes referred as *latent variables*, a denomination of variables that are inferred, via a mathematical model, from the data but they are not directly observed or measured. For the case of those constructs that are hypothesized to exist, perhaps the denomination of *"hypothetical variables"* better matches the description of a theoretical construct. This would distinguish from *"hidden variables"*, which may contain other that could be measured but, for some practical, feasibility or cost-benefit reason has not been measured so it is not part of the dataset of the study.

This means that in the analysis of some marketing datasets multiple individual variables (or measurements) are required as combined they can form a construct. It is well-documented in research covering these topics that multi-measurement items (i.e. constructs/latent variables) are better than individual measurement items as they average out the uniqueness of such individual items, make fine distinctions between people and have higher reliability [19, 75]. However, it is important when developing these measures that the domain of the construct is well specified and the items making up the multi-item construct are generated based on that domain.

Churchill [19] has created a flow diagram (Fig. 1, p. 66) with advice for creating better measurements in his 1979 publication which is still largely followed in survey research today (as shown by, for instance, Malhotra and Grover [75]). These steps are specify the domain of construct, generate sample of items, collect data, purify measure, collect data, assess reliability, assess validity and develop norms. Further, several of these steps are feedback loops to prior steps where it may be necessary to advise changes or new inputs. He also states that the accuracy level of whether the construct has captured what intended to measure depends on the rigour with which the rules have been followed [19]. Much debate on the quality of research measures and constructs has taken place in the literature and these debates show how important it is to understand the statistical rigour required for accurate and reliable survey research.

2.4.1.2 The "Cons" of Constructs

Using constructs in Psychology and Marketing research has received a lot of criticism. As stated, they represent a theoretical concept and in doing so, turn non-tangible ideas into quantifiable variables. Naturally some hiccups may be expected in this process. In the article we have discussed by Churchill [19], he refers to a publication by Jacoby in 1978 where Jacoby [58] blames much of the poor quality of marketing literature on the measures (i.e. constructs or latent variables) that researchers used to represent their theoretical items of interest. This opinion is still present today by members of the Marketing and Psychology research communities. For instance, Michell [79, p. 1] even states that "the "construct" concept is unworkable and laden with confused philosophical baggage accrued under the hegemony of logical empiricism, and its real function in psychology is obscured". Michell concludes his highly critical and in-depth historical analysis of the construct concept by stating that as long as there is no evidence of "quantitative structure" in the theoretical attributes we call "constructs" then, there is no good scientific reason to retain the "construct" concept.

Even though the above being said, Jacoby [58], in his highly critical paper on consumer research findings states that it is of paramount importance to have *construct validity*[12] when conducting research using constructs. Investigating and ensuring validity in quantitative methods, specifically construct validity in survey research is not a step that can be skipped. Therefore, following Jacoby and Churchill's arguments, there may yet be hope for survey research using constructs, as long as we follow a high standard of statistical rigour!

The main thing to remember when it comes to quantitative research is that any mistakes made during the set-up, design and execution of a research study will have confounding impacts on the reliability and validity of its findings in general as these errors at the start of the process may introduce significant flaws in the process. Measurement error is a significant problem in social sciences [4, 36]. Bagozzi et al. [4] explain that if construct validity is not assessed, the researcher cannot estimate and correct for the confounding influences of random error and method variance. This may result in ambiguous research results and possible wrongful rejection or acceptance of a hypothesis based on excessive error in measurement.

[12]https://en.wikipedia.org/wiki/Construct_validity.

This brings us to the statistics of creating and using constructs. Many things have been said about constructs on both sides of the argument and perhaps a lot of work still needs to be done; however, one thing is sure, survey research continues to be an integral part of Marketing, Psychology and other social sciences and ensuring its accuracy and reliability should never be ignored. Hence, we will briefly cover the basics of the vital statistics relating to constructs.

2.4.1.3 The Basic Statistics of Creating Constructs

As Cronbach and Meehl [21] state, the best construct is the one around which we can build the greatest number of inferences, in the most direct fashion. Being able to do this relies on the reliability and validity of a construct. A measure (an individual variable) is *reliable* to the extent that independent but comparable measures of the same trait or construct of a given concept "agree" [19]. Basically, we can say that reliability measures the overall consistency of a measure. It is said that reliability is a necessary, but not sufficient condition, for validity. It is also true that if only single-measurement (one individual variable) is used for analysis, reliability is impossible to ascertain, hence showing that multi-measurement methods are more likely to be reliable [4].

A recommended measure for internal consistency (i.e. reliability) of a set of items (in a construct) is the coefficient alpha [19], also known as Cronbach's α or tau-equivalent reliability[13] [83]. Churchill states that this is the first measure a researcher should calculate as it is laden with information because the square root of coefficient alpha is the estimated correlation of the k-item test with errorless true scores. Therefore, a low alpha score indicates that the sample of items perform poorly in "capturing" the construct. Oppositely, a high alpha score means that the k-item test correlates well with true scores.

Cronbach and Meehl first coined the term "construct validity" in their 1955's psychometric study [21]. It is almost appropriate to say that *validity* has become a sort of *holy grail* of the statistics revolving around survey research. Jacoby even states "The most necessary type of validity is construct validity" [58, p. 92]. Construct validity examines the question: Does the measure behave like the theory says a measure of that construct should behave? [21]. As stated, reliability is a necessary but not *sufficient* aspect of construct validity.

Construct validation can be done with the multitrait-multimethod (MTMM) matrix. The MTMM matrix is a correlation matrix for different concepts when each of the concepts is measured by different methods [14]. The MTMM matrix is one of the more traditional approaches for assessing construct validity and there are other methods that have been brought forward since then. For instance, Westen and Rosenthal provide two further measures of construct validation based on two effect size estimates (correlation coefficient) in their 2003 publication [106].

[13]https://en.wikipedia.org/wiki/Cronbach's_alpha.

Further, construct validity can also be evaluated through different forms of factor analysis, structural equation modelling (see Sect. 2.4.2 for more information on these methods) or other statistical measures. It is important to note however though that construct validity cannot be proven in one single research study. Rather, it is a continuous process of evaluation, reevaluation, refinement and development as the flow diagram in Fig. 1 of Churchill's 1979 paper shows [19, p. 66].

Construct validity is made up of convergent and discriminant validity. Convergent validity refers to the extent that two measures of constructs (individual variables) correlate highly with other measures designed to measure the same construct. Discriminant validity on the other hand tests whether concepts or measurements (individual variables) that are supposed to be unrelated are in fact, unrelated [4, 19]. It can also be said that discriminant validity is the extent to which the measure is actually novel and not simply a reflection of some other variable [19], (hence *unrelated* to other measures from different domains). For the keen reader who would like to learn more about survey research and the statistics of construct validation we recommend the paper by Churchill [19] or several of the other studies we have cited in this section [4, 68, 83, 106].

2.4.2 More Marketing Research

As we have now covered the basics of using constructs in survey research, we can go through a brief overview of common quantitative research methods used by marketing researchers. In many of these, surveys are heavily used and they provide a wide range of examples from statistical tools to decision support models and even clustering approaches. Up until recently (i.e. before the adoption of more advanced data analytics approaches from computer science disciplines), Marketing and Business researchers predominantly only used statistics analysis in their empirical research studies. Specifically, the field of multivariate statistics[14] is what most empirical marketing research approaches fall into. For data scientists to be able to understand where quantitative marketing researchers come from, and for the novel marketing/business analyst, this section will provide a brief overview of those methods most commonly used in marketing and consumer behaviour applications.

Multivariate statistics methods focus on the simultaneous observation and analysis of more than one outcome variable. The practical application of multivariate statistics to a particular example may involve multiple types of univariate and multivariate analyses in order to understand the relationships between variables and their relevance to the problem being studied. It is an extremely large field and includes many different statistical methods, including but not limited to, Factor Analysis, Principal Component Analysis, clustering systems, path analysis,

[14]https://en.wikipedia.org/wiki/Multivariate_statistics.

structural equation models, etc. The book “Multivariate Data Analysis” by Hair et al. [49] provides a good starting point to learn more in-depth about these methods for those interested in doing so. We will cover the basic ideas of some of these methodologies, plus others, with a focus on those mainly used by marketing and consumer behaviour researchers and will provide directions for further reading.

2.4.2.1 Factor Analysis and Principal Component Analysis

Factor analysis and Principal Component Analysis (PCA) are both well-known multivariate data analysis methods. We would like to recommend any novice reader wishing to learn more in-depth about this topic to have a look at the textbooks “Multivariate Data Analysis: A Global Perspective” by Hair et al. [48, 49] and/or “Principal Component Analysis and Factor Analysis” by Joliffe [61]. Further, for readers who are quite new to quantitative research, we recommend a “quick and dirty” introduction to factor analysis and Principal Component Analysis (PCA), via their respective Wikipedia pages.[15,16] As many readers may be researchers or professionals in the marketing and business analytics space and may already be familiar with these topics, we will keep it brief. We will cover the basics of Factor Analysis and PCA with a focus on their applications in the marketing domain and link to resources for further reading.

Factor analysis is a multivariate statistical technique that is concerned with the identification of structure within a set of observed variables [97]. Factor analysis is used to describe variability between observed, correlated variables in terms of a potentially lower number of unobserved variables (which are in turn called “factors”). In this description we can see the commonality with that of creating constructs from several individual measure items. In fact, some researchers use factor analysis whenever starting their analysis to determine (and reduce) the number of dimensions in their data [19]. Factor analysis essentially helps researchers minimize the number of variables in their analysis while maximizing the amount of information carried by the factors.

Factor analysis in fact can take two forms: exploratory and confirmatory factor analysis. The approach described above refers to an exploratory factor analysis where data reduction (dimensionality reduction) is the objective. Further, an exploratory factor analysis approach can be used to uncover hidden structures in the data (thanks to the dimensionality reduction). Factor analysis as an exploratory tool does not have many statistical assumptions. The only assumption is the presence of “relatedness” between the variables as represented by the correlation coefficient. If it turns out there are no correlations, then there is no underlying structure (no latent variables can be found in the individual measures). On the other hand, factor analysis can also be used as a tool to test specific hypotheses, in which case it refers

[15]https://en.wikipedia.org/wiki/Factor_analysis.

[16]https://en.wikipedia.org/wiki/Principal_component_analysis.

to a confirmatory approach [49, 97]. For instance, the measurement of customer (or user) satisfaction can be analysed in an exploratory fashion where theory perhaps can be built upon its findings. Oppositely, if the suspicion of a relationship or a set factor structures between certain measured variables and customer satisfaction already exists, then a confirmatory approach can be used to test this hypothesis [29]. For further reading about confirmatory factory analysis we recommend the book *"Confirmatory Factor Analysis for Applied Research"* by Brown [12].

Stewart [97] provides us with an interesting study on the applications, as well as the misapplications, of factor analysis in marketing. For instance, Stewart calls the use of factor analysis as a clustering technique an "extreme perversion" of the method. As factor analysis can be used to essentially "group" together independent individual variables into groups of variables (latent variables), the same technique has been used on the persons in the sample of the dataset, grouping them together instead. Hence, creating clusters of consumers, customers or whomever the population of interest may be. However, Stewart [97, p. 52] continues to explain that, "factors are not clusters" and this confusion may have arisen because a cluster is "more concrete, immediately evident and easier to understand than a factor" (a statement attributed to Raymond Cattell[17] in [97]).

Stewart's 1981 paper contains an in-depth presentation of different factor analysis approaches, different methods and citations to other interesting works with marketing applications at the time. He concludes that much of the criticism that factor analysis has received has been based on misinterpretation and misunderstanding (and subsequently misapplication) of the many different methods available. The above being said, factor analysis has provided marketing and social science researchers with a useful tool to analyse many different application areas. Factor analysis has been applied in business and marketing in a wide variety of applications such as the analysis of customer-perceived service quality [98], the development and validation of a solid construct for researching business' "innovativeness" [102], analysing destination competitiveness in a tourism application [32], analysing consumer "purchasing style" [92], customer satisfaction and relationship marketing outcomes in a banking application [59], and the analysis of social responsibility in terms of environmental marketing planning among European businesses [63]. These are a very few examples of the wide variety of applications in which factor analysis has been used. Nonetheless, they show the sheer diversity in which these long lasting statistical techniques can be applied in and help researchers analyse their respective topics of interest.

We can see that more recently, factor analysis has become a more standardized step in larger applied research studies where it is used as a construct and data validation tool in a more advanced methodological process. An example of this can be seen in Kim and Ko's study [65] who analyse whether social media use by luxury brands enhances customer equity. A confirmatory factor analysis is used to prove the validity of each of the measurement items, before proceeding

[17] https://en.wikipedia.org/wiki/Raymond_Cattell.

to a Structural Equation Modelling (SEM) analysis. As we learned earlier in this chapter, it is important to ensure the statistical soundness of a models' measurement constructs before deriving conclusions about a population from the study using said measurement constructs. Hence, factor analysis continues to provide a useful tool for business and marketing researchers who have extended their analytical methods well beyond the first developed factor analysis approaches. Specifically as in the example of Kim and Ko [65], SEM techniques (which include confirmatory factor analysis methods) have been used at increasingly higher rates over the last several decades with many software packages developed to help researchers in their analytic quests. More details on SEM are in the following section (Sect. 2.4.2.2).

We cover factor analysis together with Principal Component Analysis (PCA), on purpose. Factor analysis and PCA are similar and have typically been confused and used interchangeably. However, they are in fact distinctly different methods [61]. Often, PCA and factor analysis produce similar results and as Jolliffe explains, the confusion is exacerbated by software packages using the two interchangeably [61], in some cases with PCA as the default extraction method in its factor analysis routines. Many online blogs and software help pages may also help clear up the confusion between the two and provide easy-to-understand explanations for new readers in this field. One easy way of putting it is that "PCA is a linear combination of variables; Factor Analysis is a measurement model of a latent variable".[18]

PCA helps you to identify a new coordinate system to explore the data and the principal components bring information about the underlying structure in the data. They are the directions where there is the most variance; the directions where the data is most spread out. It provides an alternative for looking at data. Like Factor Analysis, PCA can also be described and used as a method of data reduction (or dimensionality reduction). We have to remember when statistically analysing our data, each added variable, adds a dimension. If you have, for instance, 16 variables that may be correlated you may use PCA to reduce your 16 measures to several principal components. If then, three components are extracted in this example, you may want to know the component scores (which are in fact variables added to your dataset) or you may want to look at the dimensionality of the data. If these three components accounted for 78% of the total variance, then you could say that those three dimensions in the component space account for 78% of the variance. Again, for those readers new to the field who would like to learn more in-depth about PCA, we refer them to the book by Joliffe [61].

PCA is most often used as an exploratory analysis tool for making predictive models. PCA is not usually used to identify underlying latent variables (or constructs). However, as with its comparative method factor analysis, applications of PCA run far and wide, even within the discipline of marketing and business analytics. Some examples include an application of PCA in the analysis of organic

[18]https://www.theanalysisfactor.com/the-fundamental-difference-between-principal-component-analysis-and-factor-analysis/.

food consumption and customer preferences towards these food products [18], using PCA as a decision model for vendor selection in a logistics application [84], an analysis of luxury brands' impact of social media marketing on customer purchase intentions and relationship marketing [66], and an analysis of online banking performance using PCA combined with data envelopment analysis [54]. Similar to factor analysis, PCA is now often used in combination with, or as part of other methodological analysis processes. Specifically some SEM Partial Least Squares (PLS) methods include PCA.

2.4.2.2 Structural Equation Modelling

As stated in the previously, Structural Equation Modelling (SEM) includes methods such as confirmatory factor analysis, path analysis, partial least squares path modelling, and latent growth modelling. SEM is often used to assess latent variables and its relationships (a.k.a. "constructs" we have covered in Sect. 2.4.1.1). The SEM analysis invokes a measurement model that defines these latent variables using one or more observed variables (individual measurement items) and a structural model which imputates relationships between latent variables. Then, the links between the constructs of a structural equation model are estimated using, for instance, independent regression equations. For in-depth reading on SEM, we refer the reader to the book "Structural Equation Modelling: From Paths to Networks" by Westland [13]. Here we provide a brief history and a basic introduction of the method.

The basis for structural equation modelling was developed by the American geneticist Sewall Wright in the 1920s with his development of the statistical path analysis method [111]. Wright showed that linear relationships among observed variables can be represented in the form of so-called path diagrams and their associated path coefficients. Through tracing causal and associated paths on the diagram, the linear structural relationship between the variables was easily observed. Some 50 years later, in the 1970s, Joreskog et al. [62] developed LISREL, one of the first advanced computer programs that implement structural equation analysis which kickstarted the widespread adoption of SEM methods in the social sciences. LISREL (as its abbreviation of 'linear structural relations' indicates) focuses on *linear* relationships in structural equations as it is heavily based on the covariance based statistical approach initially developed by Wright. Its method is however, much more flexible and generally applicable to a wide range of models [62].

Since the development on LISREL, a plethora of software packages have been developed using a SEM method approach. For instance inside SPSS, SEM analysis is possible using AMOS,[19] another software that includes SEM modelling is

[19]http://www.spss.com.hk/amos/index.htm.

Mplus[20] and in more recent years, Hair et al. [47] have championed the use of Partial Least Squares SEM (PLS-SEM) and the development of SMART-PLS[21] (more on Partial Least Squares below).

SEM first appeared in Marketing literature in the early 1980s [37] and has since become a "quasi standard in Marketing and Management research" [47]. SEM has been so widely adopted, and has been able to make a difference to such a large part of the business and marketing fields due to its ability of imputing relationships between unobserved constructs from observable variables. SEM analysis methods have been used and published in most marketing and consumer-related journals [6].

There are many applications of SEM in marketing and business literature. In first applications, the method started out as being used more as a construct validation tool [95] (like CFA). However, nowadays the method is being built upon and extended in such a way that allows researchers to build ever-more complex research models and gain greater customer insights. Besides the methodological improvements being made to SEM analysis methods and software, the range of applications in which the method is used continuously spread as well. A very quick search to some applications of SEM in the past two to three decades shows us the wide range of applications SEM is used for, even in the marketing and consumer research domains alone. Some examples of applications include; the role of commitment and trust in relationship marketing with consumers [80], analysing commitment in business relationships and partnerships [107], the analysis of customer acceptance of products in the electronics market [71] analysing the extent of customers' willingness to repurchase based on multiple variables such as equity, value, satisfaction etc. [52], customer loyalty in retail banking [8], destination loyalty in tourism based on satisfaction and motivation [112], analysis of trust in a corporate brand [90], consumers' perceptions of corporate social responsibility [94], the effects of social media marketing activities on customer equity in the luxury brand sector [65], and there are many applications looking at online consumer behaviour on social media, for instance, [24]. In short, the thousands (maybe hundreds of thousands) of studies conducted and published in the business, marketing and management fields that include some form of SEM analysis range widely in their applications and domains and continue to bring innovative insights to expand business knowledge.

One main new contribution to SEM analysis that we need to highlight has been the development of Partial Least Squares SEM. PLS-SEM was developed by Wold [109, 110], but has been particularly championed in recent years by Hair et al. [47] and has been widely adopted by marketing, business and psychology researchers. The reason why PLS-SEM has been so popularly received by the marketing research field is due to the fact that covariance-based SEM (CB-SEM) methods rely on a set of assumptions to be fulfilled such as, the multivariate normality of data, minimum sample size, a linear relationship between latent

[20] http://www.statmodel.com/index.shtml.

[21] https://www.smartpls.com/.

variables etc. [27, 47]. Thus, we cannot analyse non-parametric data with CB-SEM. In the cases that the assumptions necessary for CB-SEM cannot be satisfied, Partial Least Squares (PLS) analysis provides the solution as it allows non-parametric data to be analysed. Further, it is important to note when comparing CB-SEM and PLS-SEM, it is not the case that one is better than the other, but rather, different instances call for different approaches.

Hair et al. [47] further explain that the philosophical distinction between CB-SEM and PLS-SEM is as straightforward as when the research objective is theory testing and confirmation, the appropriate method would be CB-SEM. Contrary, when the research objective is prediction and theory development, then the appropriate method is PLS-SEM. Overall, when the measurement or model properties do not allow for the use of CB-SEM, or when the emphasis is more on exploration than confirmation, PLS-SEM is an attractive alternative to CB-SEM and often more appropriate [47]. We would recommend the book also written by Hair et al. titled “A primer to Partial Least Squares Structural Equation Modelling (PLS-SEM)” to any reader who would like to start their own analysis using SMART-PLS [46] or their online resources available on the SMART-PLS website.[22]

2.4.2.3 Multiple Regression and Symbolic Regression Analysis

Conceptually and practically, PLS-SEM is similar to using multiple regression analysis. The primary objective is to maximize explained variance in the dependent constructs but additionally to evaluate the data quality on the basis of measurement model characteristics [46].

Probably all the readers are somehow familiar with simple linear regression analysis[23]; some of the first approaches based on least squares analysis developed in the early 1800s. Regression analysis benefits from already established statistical procedures which help to estimate the relationships among sets of variables. The focus is generally to determine the relationship between a variable (said to be “dependent” or “observed”) as a function of predictor variables (also known as independent variables). In nonlinear regression, the observed variable is modelled by a mathematical function which is a nonlinear combination of one or more predictor variables and of a set of parameters which depend on the model selected. A simple market research example is the estimation of the best fit for advertising by looking at how sales revenue (the dependent variable) changes in relation to expenditures on advertising, placement of ads, and timing of ads.

However, there are cases in which we would aim that the analysis suggest a particular model. *Symbolic regression* is a methodology in which we use computers to search a space of mathematical expressions and the aim to find the model that best fits a given set of observed data. Ideally, a mathematical model which includes many

[22]http://info.smartpls.com/index.php.

[23]https://en.wikipedia.org/wiki/Regression_analysis.

independent/predictor variables (and associated set of parameters) is, a priori, more likely to better approximate observed data. However, when two models are able to approximate the data with the same level of accuracy, we generally prefer the one that uses a reduced number of predictor variables and parameters. In Symbolic Regression, heuristic and metaheuristic methods of non-linear optimization have been regularly used as an approach to search for better models. In general, no particular model is provided as a starting point to the algorithm. Popular techniques are based in Genetic Programming (GP) were already discussed in the first chapter of this book and they are also applied in some other chapters as well.

These methods help to understand how the typical value of the dependent variable being investigated is affected by changes in any of the independent variables, when all other independent variables are held fixed. Far from being a "closed field", there is a strong research focus in the area nowadays, particularly with the availability of data. Consequently, hybridization of techniques expand "simple" "old" methods hybridizing methods that have been around for a while, (e.g. PCA and multiple regression analysis [66]), or with others adaptive search techniques that are coming from the fields of computational intelligence (like Genetic Programming), or even metaheuristics, like population-based metaheuristics.

2.4.2.4 Grouping and Clustering in Marketing

As Sect. 2.2.2 covered, it is important in Customer Analytics to segment the existing market into groups of consumers that are similar, in order to be able to address their needs in a better way. However, thanks to advancements in technology, we do not have to rely only on demographics information or theory-based information alone. Instead, we can use data-driven and statistically-valid methods to segment consumers into groups in which their members are actually very similar to each other. Due to the popularity of customer segmentation, a number of methods for clustering and grouping have been applied in marketing and business analysis studies for quite a few decades.

Section II of this book covers clustering more in-depth and focuses on novel data science methods of clustering. Here, we will very briefly cover some traditional segmentation approaches more frequently used by marketing researchers prior to the adoption of more advanced data-driven analytics methods of the past two decades. First we would like to recommend the review by Punj and Stewart from 1983 [86] which covers clustering methodologies in Marketing up until that year. They focus on applications of clustering methods and the specific procedures used in clustering algorithms. Another review from a few years later by Hruschka [55] extends a segmentation review by also looking at fuzzy clustering methodologies brought forward by the literature at the time. This book covers fuzzy clustering in more detail in Chap. 3.

Two of the most common clustering techniques used traditionally in marketing literature are *k*-means (which is also covered in Chap. 3) and Ward's Minimum Variance method [99]. Ward's Minimum Variance Method is a hierarchical cluster analysis based on Ward's 1963 proposal of an agglomerative hierarchical clustering procedure for grouping activities [103]. Since its initial publication in 1963, it has been widely used across the marketing clustering literature, built upon, further developed and implemented in much more complex methods [81]. The aim of Ward's criterion is to minimize the in-cluster variance. This means that it aims to maximize the homogeneity within a cluster. The way this is works is that at each step of the method, it aims to find the pair of clusters that lead to a minimum increase in the total within-cluster variance after merging. The criterion used for the decision of whether or not to merge is based on the optimal value of an objective function which could be any objective function that accurately reflects the researchers' purpose. Ward used the error sum of squares as an example of the objective function and specifically this method is known as Ward's minimum variance method.

Another segmentation method that Punj and Stewart [86] refer to in their clustering in marketing review is iterative partitioning. This is actually a broad area of methods that differ from hierarchical clustering approaches. Iterative partitioning methods start by dividing observations into a predetermined number of clusters. Observations (objects) are then reassigned to clusters until some decision rule terminates this process. For instance, k-means is an iterative partitioning approach. With the k-means method, the pre-determined value of k indicates the number of resulting clusters. This is because the number of k sets the amount of centroids the method starts with and at the first step, each data point is assigned to its nearest centroid, based on the squared Euclidean distance. More information on the k-means method can be found in Chap. 3. Other iterative partitioning method that Punj and Stewart refer to are hill-climbing methods. With hill-climbing methods, objects are not reassigned to the cluster with the nearest centroid (as with k-means) but rather, they are moved from one cluster to another if a particular statistical criterion is obtained. Reassignment of all the objects continues until optimization occurs.

As with clustering in general, clustering methodologies used in marketing and business research take on many different forms and names, sometimes even having many different names for the same method. This makes understanding clustering or segmentation techniques used specifically in marketing research a difficult task. In order to gain an overview of the many different methods that can, and have been used for market segmentation, the review by Beane and Ennis [7] provides a good start. They include a discussion on the different clustering approaches we have covered in Sect. 2.2.2 followed by a detailed description of the many statistical and analytical methods used by marketing researchers for segmentation. They include Automatic Detection Interactor (AID), canonical analysis, factor analysis, cluster analysis, regression analysis, discriminant analysis, multidimensional scaling and conjoint analysis. These many different approaches show us that market segmentation is a hugely diverse and wide field of research in constant expansion.

Looking at more recent contributions to the literature on clustering in marketing we recommended the book by Wedel and Kamakura [105], who provide a more recent and in-depth review on segmentation approaches in marketing and cover the whole topic of segmentation from a marketing perspective, including empirical methodologies. Further, for a look at a more topic-specific review, Dolnicar [30], focuses on data-driven segmentation methodologies used in tourism applications. Hiziroglu [53] provides us with a somewhat more technical review of clustering methodologies, focusing on soft computing approaches to customer segmentation.

Finally, in the last two decades, Finite Mixture Models (FMM) have grown in use as a clustering technique in marketing literature. Mixture models are a statistical technique that uses a probabilistic model for representing the presence of subpopulations within an overall population. Before becoming a popular method in the marketing literature, they have already been successfully applied in astronomy, biology, genetics, physics, medicine, economics and other fields [78]. FMM's are said to be "elegant procedures that incorporate mixtures of parametric distributions to define the true cluster structure" [96, 99, p. 63]. They can be used to classify observations, to adjust for clustering, and to model unobserved heterogeneity. Observed data are assumed to belong to unobserved subpopulations (clusters or classes), mixtures of probability densities or regression models are used to model the outcome of interest. When the model has been fitted, class membership probabilities can also be predicted for each observation (object). The stata.com website provides some good introductions to finite mixture modelling and practical guides to those interested readers.[24] Recently, the developers of the SMART-PLS software (mentioned in Sect. 2.4.2.2), also extended their PLS algorithm to include finite mixture models to help identify and treat unobserved heterogeneity in the PLS models [46, 77]. Finally, for further reading, and a more comprehensive introduction to FFMs, we again refer the reader to the seminal work of Wedel and Kamakura [105] or the more recent updated review by Tuma et al. [99].

Next, we take a brief look at conjoint analysis in general and for further reading on clustering methodologies and new clustering ideas and algorithms, data-driven grouping methods we refer to Part II of this book.

2.4.2.5 Conjoint Analysis

Conjoint analysis is a survey based statistical technique used in market research that helps determine how people value different attributes (feature, function, benefits) that make up an individual product or service. Conjoint analysis involves presenting people with choices in a survey and then analysing what the drivers for those choices are. Conjoint research approaches are very commonly used in business market research and many commercial software packages and market research services using conjoint analysis exist [43, 108]. For instance, commercial services such as Sawtooth software[25] provide businesses with market research services using conjoint analysis methods.

[24] https://www.stata.com/manuals/fmm.pdf.

[25] https://www.sawtoothsoftware.com/.

Conjoint analysis in social science applications was developed and championed by Green and Srinivasan and has received a lot of attention by both academic researchers and business practitioners analysing consumer behaviour since the 1970s [43, 44]. It provides a good tool for measuring a customers' trade-offs among multi-attributed products and services. In Marketing applications, conjoint analysis can be used, for example, to test customer acceptance of new product designs, to assess the appeal of advertisements and in service design.

An example of a conjoint analysis experiment would be a case where the consumer is presented with four different mutually exclusive alternatives of a product. In this case, let's take a mobile phone service plan as an example. The four options could be as follows;

- 100 min talking time, 100 texts, 5 GB data per month for $54,
- 50 min talking time, 100 texts, 10 GB per month for $64,
- 0 min talking time, 100 texts, unlimited GB per month for $59, or;
- 20 min talking time, 30 texts, 20 GB per month for $59.

The consumer would then have to (hypothetically) choose between these options which one they would take. They could also be asked to rank or rate each of the options. You may have yourself participated in such a study or taken a personality test that includes this type of design.

The aim of conjoint analysis is to be able to predict the preferences of consumers and subsequently, serve them with your offering of goods and services more effectively. From responses to the questions above, conjoint analysis uncovers the underlying value for each level, depending on how often a level was included in the product selected (each of our attributes, for instance, 100 min talking time per month). The relative value of the levels is what is relevant, in other words, how the value of one level compares to the value of another. The complete set of values that represent a consumer's trade-offs are referred to as "utilities" or "part-worths". When marketers see the part-worths, they can understand which trade-offs to make in a product or service, such that it will be most desirable to the market. This is where the true benefit of conjoint analysis and its predictive power lies.

A lot of online resources provide detailed information and practical examples for further understanding and learning of conjoint analysis' processes, advantages (and disadvantages).[26,27] Many different analytics and software packages easily available to academic researchers include applications for conducting statistical conjoint analysis, such as in Excel,[28] in SPSS[29] or in R programming.[30] For a slightly more recent academic review of conjoint analysis, we refer to Green et al.'s "30 years

[26]https://www.pragmaticmarketing.com/resources/articles/conjoint-analysis-101.

[27]https://www.sawtoothsoftware.com/download/techpap/undca15.pdf.

[28]https://help.xlstat.com/customer/en/portal/articles/2062346-conjoint-analysis-in-excel-tutorial?b_id=9283.

[29]https://www.ibm.com/us-en/marketplace/spss-conjoint.

[30]https://cran.r-project.org/web/packages/conjoint/conjoint.pdf.

of conjoint analysis" paper [42], or the more recent book "Conjoint measurement: Methods and applications" by Gustafsson et al. [45]. They provide a collection of essays on conjoint analysis creating a wide and interesting view of the method as a whole, its different applications and uses in the literature.

This section is far not a conclusive list of all methods used by marketing and business researchers. It is simply a snapshot view of common methods used that heavily rely on survey research and statistical foundations and provides the reader with recommendations for further reading in those areas they may be particularly interested in.

2.5 Conclusion

As we have made clear, the business and marketing landscape is ever-changing and just as we may understand one aspect of consumer behaviours in one social networking platform, a new trend comes up or a new social media platform emerges. Every year we see new technologies supposedly "disrupting" the way we do business, the way we shop, communicate, search the web and even the way we live. Fortunately, however, the mathematical basis required for data analysis has been relatively well established by Applied Mathematics and Statistics since the early 1800s. It is also unequivocally stated that there is a clear hybridization of techniques in the horizon. One of the reasons is the existence of new types of hardware that can do unprecedented computations for data analysis. But another one is that problems which were previously "intractable" (i.e. those NP-complete problems discussed in the previous chapter, like the k-FEATURE SET), which are central for model building, can now be efficiently addressed with sophisticated methods based on heuristics, metaheuristics or even exact methods for small instances. This means that the new optimization techniques developed from Computer Science and Discrete Applied Mathematics will likely to have an impact in the development of Business, Customer and Data Analytics in general. The rise of Memetic Algorithms (and the whole Memetic Computing paradigm), surveyed in one of the chapters and a section of the book, is a witness of this wave of change.

The omnipresence of the internet and social networking technologies in all communications, business offerings, consumer's product research and exchanges between business and consumers continues to increase. In this changing world, the best we can do is to continue to learn about consumers' ever-changing tastes and apply new technologies as they emerge for the benefit of all.

In some sense, other chapters in this book try to expand on subjects presented in the previous two. Techniques in clustering, the different methods for market segmentation, the use of fuzzy logic, the use of symbolic regression, the study of techniques for graph optimization, the use of "meta-analytic" techniques, etc., are somehow extensions of the directions projected from these two chapters. Data-driven decisions are likely to be coming from a mix of different techniques, with computer science methods providing an unprecedented scalability. However, established methods from statistics are likely to continue influencing and supporting experimental and survey design, thus we decided to present in this book some of the main "traditional" approaches as a way to create awareness to the earlier generations of computer scientists about them. We truly believe in learning from "the classics" before aiming to innovate.

It is also important for everyone coming from another discipline to get a comprehensive image of how the new technologies and capabilities available can serve business and consumers better, and how to achieve an optimal outcome for all. In our "never-ending quest for 'absolute marketing understanding"', marketers and business researchers need to develop more advanced analytical skills and knowledge, and the computer scientists need to learn the intricacies of marketing and business applications. The rest of this book aims to provide the readers with a wide variety of novel examples where data science, social science, marketing, consumer behaviour analysis and more are completely intertwined. We hope to tighten the "gap" between business and computer science researchers (and practitioners) and keep the conversation open and flowing!

Acknowledgements The authors would like to thank Dr. Jamie Carlson for his review and useful suggestions for this chapter. Pablo Moscato acknowledges previous support from the Australian Research Council Future Fellowship FT120100060 and Australian Research Council Discovery Projects DP120102576 and DP140104183.

References

1. Sudhanshu Aggarwal, Juan A. Garay, and Amir Herzberg. Adaptive video on demand. In James H. Anderson, David Peleg, and Elizabeth Borowsky, editors, *Proceedings of the Thirteenth Annual ACM Symposium on Principles of Distributed Computing, Los Angeles, California, USA, August 14–17, 1994*, page 402. ACM, 1994.
2. Susanne Albers and Achim Passen. New online algorithms for story scheduling in web advertising. In Fedor V. Fomin, Rusins Freivalds, Marta Z. Kwiatkowska, and David Peleg, editors, *Automata, Languages, and Programming - 40th International Colloquium, ICALP 2013, Riga, Latvia, July 8–12, 2013, Proceedings, Part II*, volume 7966 of *Lecture Notes in Computer Science*, pages 446–458. Springer, 2013.
3. P. L. Andrade, G. Recalde J. Hemerly, and P. Ryan. From big data to big social and economic opportunities: Which policies will lead to leveraging data-driven innovation's potential? *The Global Information Technology Report 2014, Geneva: World Economic Forum and INSEAD*, 2014.
4. Richard P. Bagozzi, Youjae Yi, and Lynn W. Phillips. Assessing construct validity in organizational research. *Administrative Science Quarterly*, 36(3):421–458, 1991.
5. Frank M. Bass, Douglas J. Tigert, and Ronald T. Lonsdale. Market segmentation: Group versus individual behavior. *Journal of Marketing Research*, 5(3):pp. 264–270, 1968.
6. Hans Baumgartner and Christian Homburg. Applications of structural equation modeling in marketing and consumer research: A review. *International Journal of Research in Marketing*, 13(2):139–161, 1996.
7. T. P. Beane and D. M. Ennis. Market segmentation: A review. *European Journal of Marketing*, 21(5):20–42, 1987.

8. Asunción Beerli, Josefa D. Martín, and Augustín Quintana. A model of customer loyalty in the retail banking market. *European Journal of Marketing*, 38(1/2):253–275, 2004.
9. Indranil Bose and Xi Chen. Quantitative models for direct marketing: A review from systems perspective. *European Journal of Operational Research*, 195(1):1–16, 2009.
10. Burton F. Bowman and Frederick E. McCormick. Market segmentation and marketing mixes. *Journal of Marketing*, 25(3):pp. 25–29, 1961.
11. Stephen Brown. Art or science?: Fifty years of marketing debate. *Journal of Marketing Management*, 12(4):243–267, 1996.
12. Timothy A Brown. *Confirmatory factor analysis for applied research*. Guilford Publications, 2nd edition, 2014.
13. Westland J. C. *Structural Equation Models; Studies in Systems, Decision and Control*, volume 22. Springer, Cham, 2015.
14. Donald T Campbell and Donald W Fiske. Convergent and discriminant validation by the multitrait-multimethod matrix. *Psychological bulletin*, 56(2):81, 1959.
15. J. Douglas Carroll and Paul E. Green. Guest editorial: Psychometric methods in marketing research: Part i, conjoint analysis. *Journal of Marketing Research*, 32(4):385–391, 1995.
16. Gary H. Chao, Maxwell K. Hsu, and Carol Scovotti. Predicting donations from a cohort group of donors to charities: A direct marketing case study. *IJORIS*, 2(3):20–38, 2011.
17. Shiyao Chen and Lang Tong. IEMS for large scale charging of electric vehicles: Architecture and optimal online scheduling. In *IEEE Third International Conference on Smart Grid Communications, SmartGridComm 2012, Tainan, Taiwan, November 5–8, 2012*, pages 629–634. IEEE, 2012.
18. Gaetano Chinnici, Mario D'Amico, and Biagio Pecorino. A multivariate statistical analysis on the consumers of organic products. *British Food Journal*, 104(3/4/5):187–199, 2002.
19. Gilbert A. Churchill. A paradigm for developing better measures of marketing constructs. *Journal of Marketing Research*, 16(1):64–73, 1979.
20. Tim Cooper, Ryan LaSalle, and Kuangyi Wei. Guarding and growing personal data value. Technical report, Accenture, 01 2016.
21. L. J. Cronbach and P. E. Meehl. Construct validity in psychological tests. *Psychological Bulletin*, 52(4):281–302, 1955.
22. Marek Cygan, Marcin Mucha, Piotr Sankowski, and Qiang Zhang. Online pricing with impatient bidders. In Robert Krauthgamer, editor, *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016, Arlington, VA, USA, January 10–12, 2016*, pages 190–201. SIAM, 2016.
23. Natalie J. de Vries, Ahmed Shamsul Arefin, and Pablo Moscato. Gauging heterogeneity in online consumer behaviour data: A proximity graph approach. In *IEEE Fourth International Conference on Social Computing and Big Data and Cloud Computing*. IEEE, 2014.
24. Natalie J. de Vries and Jamie Carlson. Customer engagement, brands and social media. *Journal of Brand Management*, 21(6):495–515, Aug 2014.
25. Natalie J. de Vries, Jamie Carlson, and Pablo Moscato. A data-driven approach to reverse engineering customer engagement models: Towards functional constructs. *PLOS ONE*, 9(7):e102768, 2014.
26. Joel Dean. Problems of product-line pricing. *Journal of Marketing*, 14(4):518–528, 1950.
27. Adamantios Diamantopoulos and Judy A. Siguaw. *Introducing LISREL*. Sage Publishing, 2000.
28. Peter R. Dickson and James L. Ginter. Market segmentation, product differentiation, and marketing strategy. *Journal of Marketing*, 51(2):pp. 1–10, 1987.
29. William J. Doll, Weidong Xia, and Gholamreza Torkzadeh. A confirmatory factor analysis of the end-user computing satisfaction instrument. *MIS Quarterly*, 18(4):453–461, 1994.
30. Sara Dolnicar. A review of data-driven market segmentation in tourism. *Journal of Travel & Tourism Marketing*, 12(1):1–22, 2002.
31. Peter F. Drucker. Marketing and economic development. *Journal of Marketing*, 22(3):pp. 252–259, 1958.

32. Larry Dwyer, Robert Mellor, Zelko Livaic, Deborah Edwards, and Chulwon Kim. Attributes of destination competitiveness: a factor analysis. *Tourism analysis*, 9(1-1):91–101, 2004.
33. A. S. C. Ehrenberg. The pattern of consumer purchases. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 8(1):26–41, 1959.
34. Sunil Erevelles, Nobuyuki Fukawa, and Linda Swayne. Big data consumer analytics and the transformation of marketing. *Journal of Business Research*, "Article in Press":8 pages, 2015.
35. Jon Feldman, Monika Henzinger, Nitish Korula, Vahab S. Mirrokni, and Clifford Stein. Online stochastic packing applied to display ad allocation. In Mark de Berg and Ulrich Meyer, editors, *Algorithms - ESA 2010, 18th Annual European Symposium, Liverpool, UK, September 6–8, 2010. Proceedings, Part I*, volume 6346 of *Lecture Notes in Computer Science*, pages 182–194. Springer, 2010.
36. D. W. Fiske. Convergent-discriminant validation in measurements and research strategies. *New Directions for Methodology of Social & Behavioral Science*, 12:77–92, 1982.
37. Claes Fornell and David F. Larcker. Structural equation models with unobservable variables and measurement error: Algebra and statistics. *Journal of Marketing Research*, 18(3):382–388, 1981.
38. Ronald E. Frank and William F. Massy. Market segmentation and the effectiveness of a brand's price and dealing policies. *The Journal of Business*, 38(2):pp. 186–200, 1965.
39. Negin Golrezaei, Hamid Nazerzadeh, and Paat Rusmevichientong. Real-time optimization of personalized assortments. *Management Science*, 60(6):1532–1551, 2014.
40. Sulekha Goyat. The basis of market segmentation: a critical review of literature. *European Journal of Business and Management*, 3(9):45–54, 2011.
41. Paul E. Green. A new approach to market segmentation. *Business Horizons*, 20(1):61–73, 1977.
42. Paul E. Green, Abba M. Krieger, and Yoram Wind. Thirty years of conjoint analysis: Reflections and prospects. *Interfaces*, 31(3_supplement):S56–S73, 2001.
43. Paul E. Green and V. Srinivasan. Conjoint analysis in marketing: New developments with implications for research and practice. *Journal of Marketing*, 54(4):3–19, 1990.
44. Paul E Green and Venkatachary Srinivasan. Conjoint analysis in consumer research: issues and outlook. *Journal of consumer research*, 5(2):103–123, 1978.
45. Anders Gustafsson, Andreas Herrmann, and Frank Huber. *Conjoint measurement: Methods and applications*. Springer Science & Business Media, 2013.
46. Joe F. Hair, G. Tomas M. Hult, Christian M. Ringle, and Marko Sarstedt. *A Primer on Partial Least Squares Structural Equation Modeling (PLS-SEM)*. Sage Publishing, 2 edition, 2016.
47. Joe F. Hair, Christian M. Ringle, and Marko Sarstedt. Pls-sem: Indeed a silver bullet. *Journal of Marketing Theory and Practice*, 19(2):139–152, 2011.
48. Joseph F Hair, William C Black, Barry J Babin, and Anderson. *Multivariate Data Analysis*. Prentice hall Upper Saddle River, NJ, 7 edition, 2009.
49. Joseph F Hair, William C Black, Barry J Babin, Rolph E Anderson, Ronald L Tatham, et al. *Multivariate data analysis*, volume 5. Prentice hall Upper Saddle River, NJ, 1998.
50. Russell I. Haley. Benefit segmentation: A decision-oriented research tool. *Journal of Marketing*, 32(3):pp. 30–35, 1968.
51. Andrew Hasty. Treating consumer data like oil: How re-framing digital interactions might bolster the federal trade commission's new privacy framework notes. *Federal Communications Law Journal*, 67:293, 2014–2015.
52. Phillip K. Hellier, Gus M. Geursen, Rodney A. Carr, and John A. Rickard. Customer repurchase intention: A general structural equation model. *European Journal of Marketing*, 37(11/12):1762–1800, 2003.
53. Abdulkadir Hiziroglu. Soft computing applications in customer segmentation: State-of-art review and critique. *Expert Systems with Applications*, 40(16):6491–6507, 2013.
54. Chien-Ta Bruce Ho and Desheng Dash Wu. Online banking performance evaluation using data envelopment analysis and principal component analysis. *Computers & Operations Research*, 36(6):1835–1842, 2009.

55. H. Hruschka. Market definition and segmentation using fuzzy clustering methods. *International Journal of Research in Marketing*, 3(2):117–134, 1986.
56. Xinhui Hu, Arne Ludwig, Andréa W. Richa, and Stefan Schmid. Competitive strategies for online cloud resource allocation with discounts: The 2-dimensional parking permit problem. In *35th IEEE International Conference on Distributed Computing Systems, ICDCS 2015, Columbus, OH, USA, June 29–July 2, 2015*, pages 93–102. IEEE, 2015.
57. Philip Hunter. Biology is the new Physics. *EMBO Report*, 11(5):350–352, 2010.
58. Jacob Jacoby. Consumer research: A state of the art review. *Journal of Marketing*, 42(2):87–96, 1978.
59. Vimi Jham and Kaleem Mohd Khan. Determinants of performance in retail banking: Perspectives of customer satisfaction and relationship marketing. *Singapore Management Review*, 30(2):35, 2008.
60. Richard M. Johnson. Market segmentation: A strategic management tool. *Journal of Marketing Research*, 8(1):pp. 13–18, 1971.
61. Ian T Jolliffe. Principal component analysis and factor analysis. In *Principal component analysis*, pages 115–128. Springer, 1986.
62. Karl G Jőreskog and Marielle Thiilo. Lisrel a general computer program for estimating a linear structural equation system involving multiple indicators of unmeasured variables. *ETS Research Report Series*, 1972(2), 1972.
63. Jari Kärnä, Eric Hansen, and Heikki Juslin. Social responsibility in environmental marketing planning. *European Journal of Marketing*, 37(5/6):848–871, 2003.
64. Richard M. Karp. On-line algorithms versus off-line algorithms: How much is it worth to know the future? In Jan van Leeuwen, editor, *Algorithms, Software, Architecture - Information Processing '92, Volume 1, Proceedings of the IFIP 12th World Computer Congress, Madrid, Spain, 7–11 September 1992*, volume A-12 of *IFIP Transactions*, pages 416–429. North-Holland, 1992.
65. Angella J. Kim and Eunju Ko. Do social media marketing activities enhance customer equity? An empirical study of luxury fashion brand. *Journal of Business Research*, 65(10):1480–1486, 2012. Fashion Marketing and Consumption of Luxury Brands.
66. Angella Jiyoung Kim and Eunju Ko. Impacts of luxury fashion brand's social media marketing on customer relationship and purchase intention. *Journal of Global Fashion Marketing*, 1(3):164–171, 2010.
67. Philip Kotler. Behavioral models for analyzing buyers. *Journal of Marketing*, 29(4):pp. 37–45, 1965.
68. Jon A. Krosnick. Survey research. *Annual Review of Psychology*, 50(1):537–567, 1999. PMID: 15012463.
69. V. Kumar. Evolution of marketing as a discipline: What has happened and what to look out for. *Journal of Marketing*, 79(1):1–9, 2015.
70. Jack A. Lesser and Marie Adele Hughes. The generalizability of psychographic market segments across geographic locations. *Journal of Marketing*, 50(1):pp. 18–27, 1986.
71. Ting-Peng Liang and Jin-Shiang Huang. An empirical study on consumer acceptance of products in electronic markets: a transaction cost model. *Decision Support Systems*, 24(1):29–43, 1998.
72. Barak Libai, Ruth Bolton, Marnix S. Bügel, Ko de Ruyter, Oliver Götz, Hans Risselada, and Andrew T. Stephen. Customer-to-customer interactions: Broadening the scope of word of mouth research. *Journal of Service Research*, 13(3):267–282, 2010.
73. Christopher Lovelock and Evert Gummesson. Whither services marketing?: In search of a new paradigm and fresh perspectives. *Journal of Service Research*, 7(1):20–41, 2004.
74. Christopher Lovelock, Paul Patterson, and Jochen Wirtz. *Services Marketing*. Pearson Australia Group Pty Ltd, 6 edition, 2015.
75. Manoj K Malhotra and Varun Grover. An assessment of survey research in POM: from constructs to theory. *Journal of Operations Management*, 16(4):407–425, 1998.
76. William F. Massy and Ronald E. Frank. Short term price and dealing effects in selected market segments. *Journal of Marketing Research*, 2(2):pp. 171–185, 1965.

77. Lucy M Matthews, Marko Sarstedt, Joseph F Hair, and Christian M Ringle. Identifying and treating unobserved heterogeneity with fimix-pls: Part ii - a case study. *European Business Review*, 28(2):208–224, 2016.
78. Geoffrey McLachlan and David Peel. *Finite mixture models*. John Wiley & Sons, 2004.
79. Joel Michell. Constructs, inferences, and mental measurement. *New Ideas in Psychology*, 31(1):13–21, 2013. On defining and interpreting constructs: Ontological and epistemological constraints.
80. Robert M. Morgan and Shelby D. Hunt. The commitment-trust theory of relationship marketing. *Journal of Marketing*, 58(3):20–38, 1994.
81. Fionn Murtagh and Pierre Legendre. Ward's hierarchical agglomerative clustering method: Which algorithms implement ward's criterion? *Journal of Classification*, 31(3):274–295, Oct 2014.
82. T. A. Oliveira, Vitor N. Coelho, Marcone J. F. Souza, D. L. T. Boava, F. Boava, Igor Machado Coelho, and Bruno N. Coelho. A hybrid variable neighborhood search algorithm for targeted offers in direct marketing. *Electronic Notes in Discrete Mathematics*, 47:205–212, 2015.
83. J Paul Peter. Reliability: A review of psychometric basics and recent marketing practices. *Journal of marketing research*, pages 6–17, 1979.
84. Alberto Petroni and Marcello Braglia. Vendor selection using principal component analysis. *Journal of Supply Chain Management*, 36(1):63–69, 2000.
85. Joseph T. Plummer. The concept and application of life style segmentation. *Journal of Marketing*, 38(1):pp. 33–37, 1974.
86. Girish Punj and David W. Stewart. Cluster analysis in marketing research: Review and suggestions for application. *Journal of Marketing Research*, 20(2):134–148, 1983.
87. Alan A Roberts. Applying the strategy of market segmentation. *Business Horizons*, 4(3):65–72, 1961.
88. Roland T. Rust and Ming-Hui Huang. The service revolution and the transformation of marketing science. *Marketing Science*, 33(2):206–221, 2014.
89. Leon Schiffman, Aron O'Cass, Angela Paladino, and Jamie Carlson. *Consumer Behaviour*. Pearson Australia, Frenchs Forest NSW, 6 edition, 2014.
90. Christina Sichtmann. An analysis of antecedents and consequences of trust in a corporate brand. *European Journal of Marketing*, 41(9/10):999–1015, 2007.
91. A. Prasad Sistla, Ouri Wolfson, Yelena Yesha, and Robert H. Sloan. Towards a theory of cost management for digital libraries and electronic commerce. *ACM Trans. Database Syst.*, 23(4):411–452, 1998.
92. Noel Y. M. Siu, Charlie C. L. Wang, Ludwig M. K. Chang, and Alice S. Y. Hui. Adapting consumer style inventory to chinese consumers. *Journal of International Consumer Marketing*, 13(2):29–47, 2001.
93. Wendell R. Smith. Product differentiation and market segmentation as alternative marketing strategies. *Journal of Marketing*, 21(1):3–8, 1956.
94. Andrea J. S. Stanaland, May O. Lwin, and Patrick E. Murphy. Consumer perceptions of the antecedents and consequences of corporate social responsibility. *Journal of Business Ethics*, 102(1):47–55, Aug 2011.
95. Jan-Benedict E.M. Steenkamp and Hans C.M. van Trijp. The use of lisrel in validating marketing constructs. *International Journal of Research in Marketing*, 8(4):283–299, 1991.
96. Douglas Steinley and Michael J Brusco. Evaluating mixture modeling for clustering: Recommendations and cautions. *Psychological Methods*, 16(1):63, 2011.
97. David W. Stewart. The application and misapplication of factor analysis in marketing research. *Journal of Marketing Research*, 18(1):51–62, 1981.
98. GS Sureshchandar, Chandrasekharan Rajendran, and RN Anantharaman. Determinants of customer-perceived service quality: a confirmatory factor analysis approach. *Journal of services Marketing*, 16(1):9–34, 2002.
99. Michael N Tuma, Reinhold Decker, and Sören W Scholz. A survey of the challenges and pitfalls of cluster analysis application in market segmentation. *International Journal of Market Research*, 53(3):391–414, 2011.

100. Jenny van Doorn, Katherine N. Lemon, Vikas Mittal, Stephan Nass, Doreén Pick, Peter Pirner, and Peter C. Verhoef. Customer engagement behavior: Theoretical foundations and research directions. *Journal of Service Research*, 13(3):253–266, 2010.
101. Donald E. Vinson, Jerome E. Scott, and Lawrence M. Lamont. The role of personal values in marketing and consumer behavior. *Journal of Marketing*, 41(2):pp. 44–50, 1977.
102. Catherine L. Wang and Pervaiz K. Ahmed. The development and validation of the organisational innovativeness construct using confirmatory factor analysis. *European Journal of Innovation Management*, 7(4):303–313, 2004.
103. Joe H Ward Jr. Hierarchical grouping to optimize an objective function. *Journal of the American statistical association*, 58(301):236–244, 1963.
104. Michael Wedel and Wagner A. Kamakura. *Market Segmentation: Conceptual and Methodological Foundations*, volume 8 of *International Series in Quantitative Marketing*. Springer, New York, USA, 6 edition, 2000.
105. Michel Wedel and Wagner A Kamakura. *Market segmentation: Conceptual and methodological foundations*, volume 8. Springer Science & Business Media, 2000.
106. Drew Westen and Robert Rosenthal. Quantifying construct validity: two simple measures. *Journal of personality and social psychology*, 84(3):608, 2003.
107. Martin Wetzels, Ko de Ruyter, and Marcel van Birgelen. Marketing service relationships: the role of commitment. *Journal of Business & Industrial Marketing*, 13(4/5):406–423, 1998.
108. Dick R Wittink and Philippe Cattin. Commercial use of conjoint analysis: An update. *The Journal of Marketing*, pages 91–96, 1989.
109. H. Wold. Systems analysis by partial least squares. In P. Nijkamp, L. Leitner, and N. Wrigley, editors, *Measuring the Unmeasurable*, pages 221–225. Marinus Nijhoff, 1985.
110. Herman Wold. Causal flows with latent variables: partings of the ways in the light of nipals modelling. *European Economic Review*, 5(1):67–86, 1974.
111. Sewall Wright. Correlation and causation. *Journal of agricultural research*, 20(7):557–585, 1921.
112. Yooshik Yoon and Muzaffer Uysal. An examination of the effects of motivation and satisfaction on destination loyalty: a structural model. *Tourism Management*, 26(1):45–56, 2005.

Part II
Segmentation, Clustering and Pattern Mining

Chapter 3
Introducing Clustering with a Focus in Marketing and Consumer Analysis

Natalie Jane de Vries, Łukasz P. Olech, and Pablo Moscato

Abstract Clustering has become an extremely popular methodology for consumer analysis with many business applications. Mainly, when a consumer market needs to be segmented, clustering methodologies are some of the most common ways of doing so nowadays. Clustering, however, is a hugely heterogeneous field in itself with advances and explanations coming from many different disciplines. Clustering has been discussed in debates almost as heated as those about politics or religions, yet still, researchers and professionals agree on the methodology's usefulness in data analytics. This chapter provides the novice data scientist with a brief introduction and review of the field with links to previous large surveys and reviews for recommended further reading. The clustering contributions in this book focus largely on partitional clustering; hence, this is the type of clustering that will feature more prominently in this chapter. Besides sparking the interest of business and marketing researchers and professionals into this ever evolving methodological field, we aim at inspiring dedicated computer scientists and data analysts to continue to explore the wide application domains coming from business and consumer analytics in which clustering and grouping are making great strides.

Keywords Cluster analysis · Internal measures · External measures · k-Means · KNN

N. J. de Vries (✉) · P. Moscato
School of Electrical Engineering and Computing, The University of Newcastle, Callaghan, NSW, Australia
e-mail: natalie.devries@newcastle.edu.au; Pablo.Moscato@newcastle.edu.au

Ł. P. Olech
Department of Computational Intelligence, Faculty of Computer Science and Management, Wrocław University of Science and Technology, Wrocław, Poland
e-mail: Lukasz.Olech@pwr.edu.pl

P. Moscato, N. J. de Vries (eds.), *Business and Consumer Analytics: New Ideas*,
https://doi.org/10.1007/978-3-030-06222-4_3

3.1 Introduction

A good colleague of us once opened a seminar to computer scientists with a quote that reverberates in our heads: *"Clustering is a Religion, so I prefer not to talk about it"*. However, his talk was about a method to group data according to some similarities. His approach was different to others in that he used methods based on graph theory and combinatorial optimization. The message, nevertheless, was clear. There are many different techniques to group data and researchers seem to be particularly drawn to some mathematical models and they possess deep beliefs about them. We thus keep that in mind, we try to convey a message to newcomers in the field. There are many different ways to order and group data and practitioners need to be aware of the large variety of techniques that exist before passionately embracing only a small subset of them.

Clustering is a large methodological field, with many different approaches, algorithms and applications. Many good reviews exist and it is very difficult to select "the best one" as the criteria would depend on the reader and the application. There is indeed an extremely large body of literature in clustering, including many reviews of the area. For instance, the review by Jain et al. [57] titled "Data Clustering: A Review" is one of the most cited and comprehensive introductory reviews to the area of clustering. Jain added to this extensive review in 2010 with a more recent review of clustering in "Data clustering: 50 years beyond k-means" [56]. Some more reviews and introductions to data clustering have been done by Kaufman and Rousseeuw [61], Xu and Wunsch II [116] and Gan et al. [42] among many others. A recent survey on clustering methods based on combinatorial perspective was published by Levin [70] (and just this subset of the literature exceeds 200 selected references). Here we will add to this rich body of research with an emphasis on clustering in more recent business and marketing applications.

Even though clustering is such a popular area and methodology, and it is the source of many hard computational problems (i.e. those that we discussed in the first chapter as being NP-complete) which is why it continues to be further investigated today and why it is a prominent topic in this volume. We mentioned before that when a problem is NP-complete it is unlikely that efficient algorithms can be found for them. However, the mathematical model may be very useful for a practitioner (see, for instance [78, 94]). As a consequence, heuristics and metaheuristics are applied to find feasible solutions for these problems when we face large datasets.

This chapter will provide the reader with a useful introduction to the area of clustering, an idea of how and why clustering is an important (almost crucial) area to business and consumer analytics and provide specific examples of existing clustering methodologies used in these areas. A solid understanding of clustering methodologies, their inputs and their outputs, will go a long way to providing the data analytics novice with a solid base for further data-science exploration. For data science experts, it is useful to reflect on those methods most commonly used in marketing and in business and consumer analytics applications. First, we provide the novice data scientist with an introduction into clustering, how it works, why we do it, the most common and popular methods, their pros and cons and a final focus on those methodologies particularly developed by business and consumer analytics researchers.

3.2 The Methodology of Clustering

In Clustering the objective is to assign labels to objects (or observations, or data points). A set of objects that have the same label (or labels) is said to be a "group" or a "cluster". The aim of clustering algorithms and heuristics is to achieve the best possible grouping. The outcome of the algorithm applied will thus depend on the choice of the similarity between the objects. It will also depend on the nature of the dataset. In this way, we can define a cluster as a collection of data instances (or objects) which are "similar" to each other and "dissimilar" to instances in other clusters [72]. To give a proper introduction to clustering we cover the questions of *"What is clustering?"* and *"Why do we cluster?"* We also provide a brief introduction to the main different types of clustering approaches including those most frequently used by business and marketing researchers and practitioners. Furthermore, as with any data analysis method, it is important to be able to evaluate or compare our results and we include a brief explanation of some approaches to do this.

3.2.1 What Is Clustering and Why Do We Do It?

There are many definitions for clustering, but in essence, it is a methodology with the purpose of organizing objects into groups (clusters) that are similar to each other (and dissimilar to other groups) based on a set of measurements/characteristics that are known about those objects. These objects could be consumers, patients, companies, products, images, genes and proteins in a biological network or any other dataset that could contain multi-attributed objects. Examples of the measurements or characteristics could be online consumer behaviour patterns, a set of answers to survey questions by clients, customer shopping patterns, financial investment patterns, gene expression patterns and so forth. Ideally, the objective is to group them, based only on the information provided in the dataset, without biasing the clustering method/algorithm how to group them, how many objects to group in each cluster, and ideally, without predicting the number of resulting groups and with the least amount of a priori parameters (more on this later in this chapter).

A "natural" clustering method would generally convey the meaning that the basic method is an unsupervised learning approach. Unsupervised learning, unlike with supervised learning techniques, has no a priori classes and no identification labels or partitions are given [72]. The objective of these techniques is to uncover the *natural groupings* of objects in a dataset.

It is thus very important to distinguish between supervised and unsupervised learning. Supervised *clustering* would not be clustering, it would in fact be *classification* [57]. Classification can be called supervised learning because, in a way, the method operates under *supervision* by being provided with the actual outcome of each of the training examples (it's class) [113] by the person operating it. Classification is used when there are existing classes to which the data objects belong to, such as classifying patients into disease subtypes in the area of bioinformatics or classifying products into their correct categories for an online retail webpage.

In many clustering applications in a business and marketing context, classes are usually not known a priori and analysts often aim to find and explore "the unknown" in their dataset without previous assumptions. Three main purposes of clustering in general are presented in Jain [56] (we quote):

- Underlying structure: to gain insight into data, generate hypotheses, detect anomalies, and identify salient features.
- Natural classification: to identify the degree of similarity among forms or organisms (phylogenetic relationship)
- Compression: as a method for organizing the data and summarizing it through cluster prototypes.

These purposes convey the drive to find out the "unknown" about a group of people, a set of topics or documents or any other dataset with underlying natural groupings.

Clustering has not been "championed" by one dominant discipline in particular, but rather it has received many contributions from many disciplines [92]. As a consequence, there are many different approaches, varying vocabularies and sometimes even multiple names for the same approach.

To make this area even more complex, there are endless amounts of applications for the ever-increasing number of clustering methodologies. A popular field in which clustering has been used extensively is in the medical, health and biological research domains. Considering clustering is an unsupervised learning technique, it is a useful tool for exploratory analysis of large datasets that we do not know a lot about. This is why it has been very useful for medical and health researchers in recent decades who may deal with millions of data points when analysing datasets of thousands of people with thousands of samples and variables. Business, finance, marketing, psychology and many other areas have rapidly caught up with the size of datasets that they produce. Consumer analytics is now a big contender for large data instances that can be generated in a very short period of time. This means that highly scalable and high performance clustering algorithms are now necessary for successful business and consumer analytics.

As Chap. 2 has already explained, in marketing, a common objective is to segment the market into similar segments of consumers. Market segmentation has therefore been the most common application for clustering methodologies in marketing and consumer analytics to date. However, with the ever-increasing size of data instances, data types, sources and applications, clustering exercises now have many more uses in business and consumer analytics. Other applications include exploratory research of a large dataset, inputs for other methods such as recommender systems, visualization of a set of information, product analysis, logistics research and operational applications, financial analysis and prediction among many others.

In this volume some of these applications will be presented. In this chapter, however, we will focus on those methods of which the purpose is to uncover a *natural grouping*. That is, *clustering* (or partitioning) is the main purpose of the experiment. We need to look at the different clustering methodologies that already exist, particularly the most standard and well-known clustering algorithms and look at those methods most commonly used by consumer behaviour and business analytics researchers.

3.2.2 Different Types of Clustering

There are *many* different types of clustering approaches and many different names for highly similar clustering methodologies. The most well-known types of clustering include: partitional, density-based and hierarchical methodologies [100]. Most clustering approaches can be said to fall in one of these three categories, especially any method you will come across in this book. Some approaches such as k-means or nearest-neighbour clustering are described in further detail in Sects. 3.5.1 and 3.5.2, respectively. However, other clustering approaches besides the three "main" categories do exist. For instance, distribution-based clustering, which is somewhat similar to density-based clustering but rather than separating clusters by "low density" areas, it investigates the *distribution*, or spread, of the data points around the initial centre of the clusters. In this section we will discuss partitional, density-based and hierarchical clustering approaches as well as briefly introduce various other methodologies.

3.2.2.1 Partitional Clustering

The most commonly used clustering methods (at least in analysing consumer behaviour and for market segmentation) are partitional clustering methods. Partitional clustering separates a dataset into a set of disjoint clusters. With partitioning clustering, the objective is to maximize some function that measures the similarity of objects within the clusters, while at the same time, minimizing the similarities between objects assigned to different clusters. Partitional clustering procedures generate a single partition (as opposed to a nested sequence) of the data in an attempt to recover the natural grouping present in the data. There are many different partitional clustering methods and approaches, and in this chapter we will cover the most common ones and those commonly used by business and marketing researchers.

Partitional clustering can again be split into two sub-categories: *hard* and *fuzzy* clustering. With hard clustering, each data point is assigned to one and only one of the clusters which means that there are well-defined separations between clusters. However, with real-world data, there are often no real and well-defined boundaries between groups of objects that are being investigated. Particularly when it comes to analysing humans (rather than physical properties) whose behaviours could not lead to characteristics which can be discretized. For this reason, fuzzy clustering has increased in use and popularity. With fuzzy clustering, each node (or object) has a variable degree of membership in each of the output clusters [57]. More details on fuzzy clustering approaches are provided in Sect. 3.6 of this chapter.

3.2.2.2 Hierarchical Clustering

Hierarchical clustering produces a dendrogram, a structure that is a nested sequence of clusters which look like a *tree* as depicted in Fig. 3.1. The y-coordinate of the horizontal line is the similarity of the two clusters that were merged, where the objects being clustered are viewed as singleton clusters. Similarity will be further discussed in Sect. 3.2.3. Hierarchical clustering uses either a top-down or a bottom-up algorithm which has implications on the way in which the data is separated by the algorithm [20]. They can also be referred to as either divisive (top-down) or agglomerative (bottom-up). Depending on the approach selected, a different outcome can be obtained because at the top of the dendrogram, there is one root cluster which covers all data points, whereas at the bottom of the hierarchy, there are singleton clusters (representing individual data points) [72].

With divisive hierarchical clustering, all the observations are assigned to a single cluster and the first step splits them up into two least similar clusters. These are then each split up again and so forth. This process is continued iteratively until there is one cluster for each of the observations at the bottom of the hierarchy. Oppositely, with agglomerative clustering, each observation is assigned its own cluster and then, based on similarity, they are combined into clusters. This is repeated until there is only one cluster left at the top. Agglomerative (bottom-up) clustering is more frequently used than top-down clustering.

A well-known and popular consumer-related example of hierarchical clustering that can be used to easily explain this method is that of an online retail store. Or even better, an online retail aggregation website which combines the products on offer from several other sites into one place. The problem at hand is to organize each product (for example, items of clothing), into subcategories according to a clothing category hierarchy. Each online retail store could have thousands of products in different colours and styles which could lead to tens of thousands of products combinations (data points) to be analysed for a web retailer aggregation website. The question is how should these products be organized into categories? One way is for a human team to manually organize each product into their correct category at the correct level in the hierarchy. However, the manual organization is extremely time-consuming and therefore very costly. A better way to do this would be through a hierarchical clustering algorithm that sorts through the products based on their features and meta-information. Or even visually analyses the image and uses

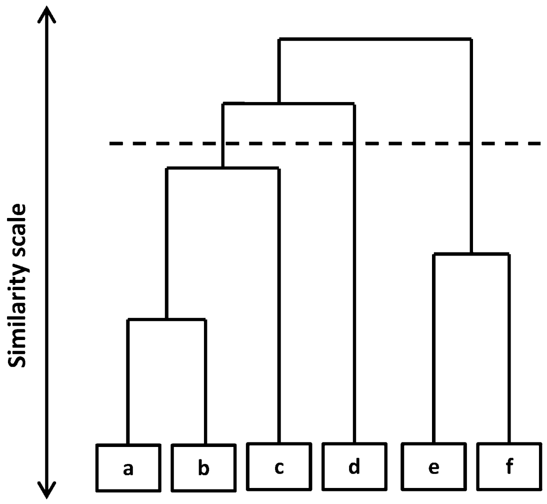


Fig. 3.1 A basic representation of a dendrogram showing hierarchical clustering. Each merge is represented by a horizontal line. The y-coordinate of the horizontal line is the similarity of the two clusters that were merged, where the objects being clustered are viewed as singleton clusters. The dashed line shows a section of the hierarchy that can be selected by the user for inspection. At this point we have three clusters: one cluster containing points *a*, *b* and *c*; one cluster containing only point *d* and one cluster containing points *e* and *f*

machine learning techniques to categorize the image (and give some extra help, for instance, by suggesting proper sets of tags according to the hierarchy level, etc.). This is one of the many examples of the use of hierarchical clustering. In essence, hierarchical clustering can be used for any set of information where there is some form of ranked order to be uncovered.

3.2.2.3 Density-Based Clustering

Density-based clustering uses a model to group objects according to specific density objective functions. Density is generally defined as the number of objects in a particular neighbourhood of a data objects. It is for this reason that density-based clustering is common in spatial applications of clustering. Density-based clusters are separated from each other by continuous regions of low density of objects. The Density-Based Spatial Clustering of Applications with Noise (DBSCAN) is an algorithm introduced by Ester et al. in 1998 which provides a density-based clustering suitable for managing spatial data [97]. This algorithm is probably the main density-based approach known to researchers. Figure 3.2 shows how density based clustering works. The two dense regions are clearly separated by less dense regions which is why they are in two separate clusters.

In the image, we can see data point alpha. With certain methods, such as k-means clustering (which we will explain in Sect. 3.5.1), the partitional clustering process starts by selecting centroids in the graph which will see its neighbouring nodes becoming part of its cluster. Therefore, if point alpha would be selected as an initial centroid, and the next nearest randomly selected centroid is point beta, then the node connected to alpha with a dashed line would likely be attributed to the smaller blue (lighter colour) cluster on the left rather than the red bigger cluster on the right. In instances like these, it is clear to see why and when density-based clustering would be the more appropriate method as in this case the density-based approach is more successful at uncovering the natural grouping (even through the peculiar shapes of the clusters). One limitation of density-based clustering methods is that they have trouble handling high-dimensional data [56]. This same aspect is the reason why other methods, such as k-means, have been more frequently and are so popular as they handle large datasets better.

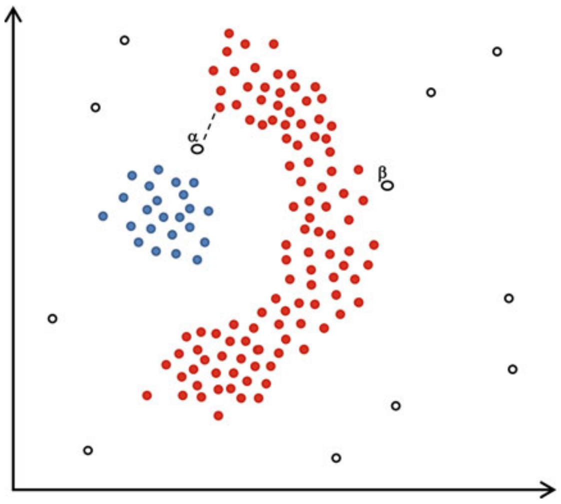


Fig. 3.2 This figure shows two clusters that are rather different in size and shape and are separated by an area of lower density. These two clusters would be identified much better using a density-based clustering technique than with, for instance, a k-means approach. If point alpha is selected as one of k-means initial centroids, it would likely attract the point connected via a dashed line to its cluster. This would in turn mean that the separation created by the lower density area would not be recognized by the method

3.2.2.4 Model-Based Clustering

Model-Based (MB) approaches can provide alternatives for heuristic approaches and have become more prevalent in the marketing literature since the early 2000s [105]. MB clustering approaches have the goal to optimize the fit between the given data and some mathematical model [48].

A mixture model corresponds to the mixture distribution that represents the probability distribution of observations in an overall population (dataset). Gaussian mixture models are some of the most commonly used model-based approaches. They investigate the number of Gaussian distributions evident in the data. A Gaussian distribution is simply a name for a “normal distribution” (or the “bell-shaped” distribution). The Gaussian distribution is a continuous function which approximates the exact binomial distribution of events. The Gaussian distribution is normalized so that the sum over all values of x gives a probability of 1. A Gaussian mixture model is a probabilistic model that assumes all the data points are generated from a mixture of a finite number of Gaussian distributions with unknown parameters. Variations of Gaussian mixture models can also be used for hierarchical clustering applications [88].

Finite Mixture Models (FFMs) are a type of MB approach and FMMs have become more prominently used in marketing literature as they are able to simultaneously derive segments and segment-specific weights that relate to dependent variables (e.g. ratings) to a set of independent variables (e.g. product quality) as well as derive a unique regression model for each segment [105]. Those who are great advocates of FMM say that it is/should be a preferred approach because it is a formal statistical model (i.e. does not have a priori parameters such as k-means). A methodologically detailed review of Finite Mixture Modelling for the keen readers can be found in Melnykov and Maitra [81].

3.2.2.5 Hybrid Approaches

Of course, there are several other methods that combine some of the approaches above. One example is the hierarchical clustering approach based on “Arithmetic-Harmonic Cuts” of Rizzi et al. [94]. In this approach the method works by finding a partition of a set of objects that are linked by weighted edges of a graph. At each stage, an NP-hard optimization problem needs to be solved. The solution of this problem is a partition of the graph vertices in a way that minimizes an objective function (the weight of the arithmetic-harmonic cut). Then the two sets of vertices of the partition are recursively partitioned. The input is now the set of edges that is not part of the previous cut, thus again giving rise to another NP-hard optimization problem of the same type. This methodology combines partitioning within a “top-down” hierarchical clustering approach and it has been tested in different scenarios in [78, 94].

Another hybrid approach, again using hierarchical approaches is that of Fernández and Gomez where they include multidendrograms into an agglomerative hierarchical clustering approach [35]. They propose a variable-group algorithm groups more than two clusters at the same time when ties occur to deal with the problem of non-uniqueness when two or more distances between different clusters coincide during the amalgamation process.

The types of clustering and grouping approaches presented in this section are not an exhaustive list of ways to segment consumers. Many more grouping, regression, clustering and statistical approaches exist and are being invented as this area of research continues to grow (including more combinations of hybrid approaches). However, the ones presented in this section provide a good basic understanding of the most commonly used clustering, segmentation and grouping methodologies.

3.2.3 Distances and Similarities in the Context of Clustering

When reading about clustering, grouping and classifying, you will hear and see the words “similarity” and “distance” come by many times. In many cases, the input for a clustering algorithm is not a similarity matrix. Instead, each object will have a set of features (also known as variables, attributes or characteristics) which may be extremely unique to that object, or very similar to other objects of interest. It is this information that is needed for many clustering algorithms. In the case of segmenting consumers, customers, users or followers, products, these are your objects and the variables relating to them (e.g. purchase patterns) are the features. In some cases, distance and similarity metrics will usually be (or will be normalized to) a value between [0,1]. With similarity metrics, a value closer to 1 means *more similar* and with distance metrics a value closer to 1 means *more distant* from each other.

The selection of a proper distance or similarity metric between objects then becomes an interesting issue in itself. There are many different metrics available for use and which one you choose, depends on your dataset, the context and the nature of your data among other aspects. In fact, a whole “Encyclopedia of Distances” has been published [28] and in no effect could we match this here. Instead, we provide a brief overview and introduction to the most commonly used distances and similarity metrics used for clustering. Further, we introduce some of the metrics that are used by subsequent chapters in this volume.

Distance matrices can be used to generate other graphs such as proximity graphs, relative neighbourhood graphs (RNGs) or any other distance-based graph such as those introduced in Chap. 4.

3.2.3.1 Distance and Similarity Metrics

Given a set of objects, a *metric* (or *distance function*) is a non-negative function that defines how far apart each pair of objects of a set are. Formally metric $d(a, b)$ is a function that for objects a and b:

1. returns 0 as the distance from point to itself, i.e. $d(a, b) = 0 \Leftrightarrow a = b$ (*identity of indiscernibles*),

2. distance between any two points is the same, regardless from which point we start to measure, i.e. $d(a, b) = d(b, a)$ (*symmetry*),
3. distance between any two points is lower or equal to the distance between the same points, but measuring through any third point c, i.e. $d(a, b) \leq d(a, c) + d(c, b)$ (*triangle inequality*).

For objects in an Euclidean space, a common and old metric to use is the *Euclidean distance*. The Euclidean distance is described as the "ordinary" (or straight-line, "like the crow flies") distance between two points in an Euclidean space [28]

$$d_{euc}(a, b) = \sqrt{\sum_{i=1}^{n}(a_i - b_i)^2}, \tag{3.1}$$

where $d_{euc}(a, b)$ is the Euclidean distance, $a = [a_1, a_2, \dots a_n]$ and $b = [b_1, b_2, \dots b_n]$ are the n-dimensional objects between which we want to calculate distance and n is the number of features that correspond to the points. How "good" the Euclidean distance is at finding the most appropriate partitions of a dataset depends on the circumstances. As our example already showed in Fig. 3.2 however, sometimes a straight-line distance may not always be the best approach for finding the most natural underlying groupings in a dataset. Therefore, many other metrics have been suggested since to deal with other datasets and instanced for which the Euclidean distance may not be the most appropriate. A variation is the "squared Euclidean" which is commonly used for k-means clustering. K-means is implicitly based on pairwise Euclidean distances between points, because the sum of squared deviations from each centroid is equal to the sum of pairwise squared Euclidean distances divided by the number of points. Hence, the k-means approach needs to use the squared Euclidean distance rather than the standard Euclidean distance.

Another distance is called the *Manhattan distance* d_{cbox}:

$$d_{cbox}(a, b) = \sum_{i=1}^{n} |a_i - b_i|. \tag{3.2}$$

Figure 3.3 shows how the Euclidean distance and the Manhattan distance work and how they differ from each other. The Manhattan distance gets its name from "going around the city block"; something that city-dwellers in Manhattan probably know all too well and accordingly also gets referred to as "city-block". The Manhattan distance function finds the distance that would be travelled to get from one data point to the other if a grid-like path is followed (i.e. on a vector space it is the sum of the absolute value of the differences on each coordinate dimension).

Both Euclidean (Eq. (3.1)) and Manhattan (Eq. (3.2)) can be generalized by the *Minkowski distance d_{min}*

$$d_{min}(a,b) = \left(\sum_{i=1}^{n} |a_i - b_i|^p \right)^{\frac{1}{p}}, \tag{3.3}$$

which is also called as L_p. In case $p = 1$ (L_1) it is a Manhattan distance, when $p = 2$ (L_2) it gives Euclidean.

Another distance is the *Chebyshev distance* (or Tchebychev distance). This is a metric defined on a vector space where the distance between two vectors is the greatest of their differences along any coordinate dimension. It is also called as a *Chessboard Distance*, because it shows the minimum number of king figure from one square to another. This distance is equivalent to Minkowski distance (L_∞) with $p \to \infty$ (Eq. (3.3)).

A *Similarity Measure* is a function assessing resemblance between objects. Contrary to a distance function, a similarity measure is a real-valued function that can give negative values and it is commonly assumed that similarity is an inverse of distance. More similar objects (high value of similarity) would have lower distance between them, and distant objects (higher metric value) have lower similarity. Even though similarity measure is symmetric it does not necessarily meet other distance properties (provided at the beginning of Sect. 3.2.3.1). Furthermore, in most of the cases, a similarity function is not additive. In such a case, adding or subtracting similarity values or computing an average is invalid, but similarities can be multiplied (scaled) for better comparison purposes, for instance.

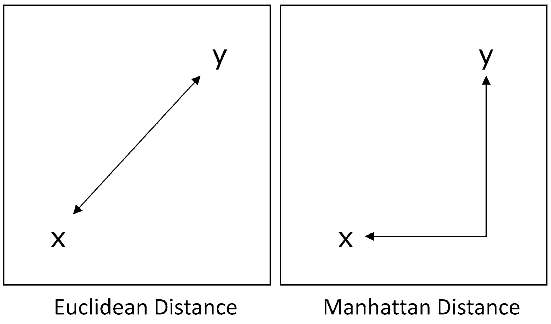


Fig. 3.3 A figure representation of the Euclidean and Manhattan distances which shows how the Euclidean distance is "how the crow flies" and the Manhattan distance "going around the city block" following a grid-like path between two data points

An example of a similarity measure is *Cosine Similarity* (s_{cos}) that uses vectors created from data points (a and b) in such a way that a vector begins in an arbitrary point (usually z = (0, 0)) and ends in point indicated by a or b. It is formulated as follows:

$$s_{cos}(a,b) = \cos(\Theta) = \frac{a \cdot b}{||a||\ ||b||}, \tag{3.4}$$

where Θ is the angle between vectors created by a and b, $||a||$ and $||b||$ are the lengths of vectors $d_{euc}(a, b)$ and $d_{euc}(a, b)$, respectively, $a \cdot b$ is a *dot product*

$$a \cdot b = \sum_{i=1}^{n} a_i b_i. \tag{3.5}$$

Notably, the cosine measure considers similarity as an angle (Θ) between the vectors. The magnitude of the vectors is not considered. Vectors with the same orientation are regarded as similar giving a maximum value $s_{cos}(a, b) = 1$, whereas perpendicular vectors have 0 similarity. On the other hand opposite vectors will result in lowest similarity $s_{cos}(a, b) = -1$.

Cosine similarity is often used in information retrieval and text mining [62, 63]. From business analytic perspective this could be valuable in product review analysis [11] or automatic product recommendation. Assuming that two text documents are described by vectors of word occurrences, this measure will give the similarity between those documents, regardless of their sizes. In other words, cosine similarity indicates how similar the subject is of the two given texts. Furthermore, this measure is efficient to calculate on *sparse vectors* (vectors having many zeros) since only non-zero values are important.

Since similarity and distance are opposite to each other, distance can be transformed into similarity, but it is important to note that a reverse operation is not always possible. As a consequence, Euclidean distance can be used to describe objects' similarity, but cosine similarity cannot be used as a distance function. Assuming that distance takes values in the closed interval [0, 1], a common way to transform distance function value (d) into similarity (s) is

$$d = 1 - s. \tag{3.6}$$

Another way to compute similarities is to use correlation metrics. These will in fact produce a number that relates to similarity (correlation) between points. Two of the most common correlation metrics are Spearman and Pearson correlation. The Pearson correlation coefficient should be used to cluster objects with similar behaviour patterns as those with opposite behaviours are assigned to different clusters. A variation of the Pearson correlation is the Absolute Pearson correlation where the absolute value of the Pearson correlation coefficient is used; hence, the corresponding distance lies between 0 and 1. Spearman correlation clusters together those objects who's profiles have similar shapes, that is, their trends are similar but the actual values may be quite different. The authors have previously used various distance metrics and correlation metrics in a clustering study and compared the effects on the outcome of using various similarity and distance metrics [75].

3.3 Measuring Clustering Quality

Some clustering methods are designed to take advantage of extra sources of information. For instance, if the number of clusters a user expects to find in the data is known (if that number is known a priori or can reasonably be guessed or predicted within reasonable bounds). In other cases, the number of clusters could be a user-defined request (due to reasons which do not belong to the data study in question). This information can be provided either explicitly, by the value of the *k* parameter as it is in the *k-means* algorithm (described in Sect. 3.5.1) or more implicitly, by providing a density of clusters that the method will be looking for (e.g. [34]). Moreover, considering Fig. 3.8 (in a section further in this chapter) it is visible that the *k* parameter has a huge influence on the method's result. After all, the vast majority of clustering algorithms have some parameters that should be (more or less) carefully tuned. This situation raises a question, *how to choose the appropriate k (and possibly other parameter's) value?*

One obvious technique is to execute the method with different parameter values and then *compare* their results in order to finally choose *"the best one"* (as the authors were able to do in [75], thanks to class labels that could be used for post hoc statistical analysis), but the process of a comparison might not be so straightforward. There could be a temptation to simply look at the results and subjectively decide which clustering result is better, but considering a real-life situation where the number of features is more than three, and the number of points is 10,000 or more, attempting to do this via a visualization approach becomes difficult. Furthermore, what happens when there are 1000 clustering results? One possible solution is to visualize the clustering in a grid of pairwise two-dimensional plots. In such a grid, every feature is plotted against each other giving a set of plots that are easier to analyse. The plot's size grows quadratically with the number of dimensions. For instance, having five-dimensional data, the grid would contain $5^2 = 25$ plots. How hard and time-consuming would it be to analyse such plots? Even though it is possible to reduce the dimensionality of the input data by aggregation or to use a dedicated method, such as *Principal Component Analysis (PCA)* [12], there is a risk of losing valuable information.

Considering all mentioned problems, it becomes clear that there is a general need for a tool that assesses which clustering is better than others, and it is desirable for the tool to be fully automated. Due to this, researchers devised the idea of *Quality Measures*. These are mathematical formulations aiming at expressing the quality of a clustering result as a number in a predefined interval. Quality measures will be discussed more in Sect. 3.3.1. Having two clustering results, we can compute their quality using a particular measure, and by comparing these measures, we can decide which one is of *higher quality*. This process is going to be discussed in Sect. 3.3.2. However, what does it mean to have *better quality*? One could define quality in a variety of ways, and because of this, there are many measures and measure paradigms aiming at assessing clustering. To choose the most suited measure, in Sect. 3.4 we elaborate on types of clustering measures and introduce the most common measures in a systematic way.

As will be further elaborated on in Sect. 3.6, apart from *hard* (*crisp*) clustering, there is a *fuzzy* approach, however in this section, we are going to focus only on the assessment of non-fuzzy non-hierarchical clustering results. Nevertheless, most of the presented measures can be easily applied to other types of clustering as well.

3.3.1 What Is a Quality Measure?

As was stated, a quality measure is a function that gives a quantitative rating to the outcome of a clustering algorithm. A clustering $\mathcal{C}$ is a set of clusters (i.e. $\mathcal{C} = \{C_1, \ldots, C_p\}$), where C_i is the ith cluster of the clustering. An example of such a measure could be the average size of clusters:

$$Q_{avgSize}(\mathcal{C}) = \frac{1}{|\mathcal{C}|} \sum_{\forall C_i \in \mathcal{C}} |C_i|, \tag{3.7}$$

where $Q_{avgSize}$ is a measure, $\mathcal{C}$ is a clusterization result, $|\mathcal{C}|$ is the number of clusters in the particular clustering under study and $|C_i|$ is the number of objects that belong to cluster C_i.

Assume that we have two different clustering outcomes $\mathcal{C}_1$ and $\mathcal{C}_2$ that are rated by $Q_{avgSize}$ in the following way:

$$Q_{avgSize}(\mathcal{C}_1) = 5, \qquad Q_{avgSize}(\mathcal{C}_2) = 17.5.$$

If we would be interested in configuring algorithm parameters in such a way to get (on average) smaller groups, then we would prefer clusterings to have lower $Q_{avgSize}$ values so that we would pick $\mathcal{C}_1$ as a final result. That interest could be present when creating personalized music recommendations for customers based on a music database. That if someone has listened to one of the songs, the remaining songs in the cluster will more likely suit their taste. As such, smaller clusters will reduce the chance of suggesting unrelated songs. Contrary, if someone else would prefer bigger groups, e.g. when trying to find main general music taste among all customers, then it would be better to pick $\mathcal{C}_2$. As a third option, if someone is not interested in the group sizes, but in a different aspect of clustering, they should rather look for another measure that would reflect his demand on the clustering.

This example shows that *better quality* can be treated differently by people and sometimes the clustering that is right for one application does not fit the needs of another. Due of this, there are many different quality measures, promoting various aspects of clustering and the meaning of *the best* clustering should be indicated by the task's specific requirements on a case-by-case basis (its application). Therefore, in Sect. 3.4, we present several measures and will explain the intuition behind them.

The usage of a quality measure should be application driven. Due to this, it is critical to understand both the clustering problem and the possible quality measures that could be used. It is vital to do this before starting to implement a solution. In order to do this, there are several important questions to answer before using any particular quality measure:

1. What clustering aspects are promoted by that measure and what aspects are ignored?
2. What are the possible values of that measure (i.e. what are the measure's boundaries)?
3. What values we are looking for and what we would like to avoid? (In other words, what values indicate better and worse clustering?)

The first question forces us to *understand* the measure. After doing this, it is much easier to address the remaining questions. As we have already noticed, the $Q_{avgSize}$ (Eq. (3.7)) measure is focused only on the average group sizes. So this measure will not tell us anything about the number of clusters or their distribution. At this point, we can decide whether this quality measure suits our needs or we should rather search for another one.

The second question can tell us what values we can expect to obtain. Sometimes the values vary between 0 and infinity (∞), in other words, the boundaries are, theoretically, $(0, \infty)$ (e.g. any positive real number). If it can't reach 0 or $[0; \infty)$ if it can. However, in the general case, the boundaries may be different, e.g. $(-\infty; \infty)$, $(-\infty; 0)$, $[1; \infty)$ or even between some arbitrary values (e.g. $[3; 50.5]$). By looking at our measure in Eq. (3.7), we can derive that it is impossible for the measure to be negative since clusters sizes cannot be below 0. This measure can give 0 only in a situation when all the clusters will be empty, so no points were grouped. For the sake of simplicity, we can assume that it is not a valid clusterization, and by this, the lowest possible value for our measure is not going to be equal or below 0. What about the highest value? It could be any positive number. One can imagine that if we have only one massive cluster, e.g. 10^{25} points, then the $Q_{avgSize}$ value would be very high (exactly 10^{25}). So the upper-bound is ∞ which gives the final boundaries of this measure as: $(0; \infty)$.

By now, we know what values to expect, but what numbers indicate a preferable result? This leads to the last question. As it was stated earlier in this section, sometimes we would like to *minimize* that quality measure value and sometimes *maximize* it with respect to the provided boundaries. In most of the cases, we are trying to reach the boundary value (either upper or lower one). However, with some other quality measures, we should reach a specific value within the boundaries, e.g. 0 or 4. Coming back to the $Q_{avgSize}$, being interested in smaller clusters, what we would be doing is to *minimize* the value of $Q_{avgSize}$.

3.3.2 How to Use a Quality Measure?

Now it is the time to use a quality measure in practice. Assume that we have the well-known *Iris dataset* [37] and we would like to tune k-means algorithm on this data. As it was stated in Sect. 3.5.1, the most important k-means parameter is actually the value of an integer $k > 0$.

The dataset that we are going to use contains measurements of three species of Iris Flowers: *Iris Setosa, Iris Virginica* and *Iris Versicolor*. For each class (species) we have 50 samples (varieties), and for every example we have four features measured, i.e. *Sepal Length, Sepal Width, Petal Length* and *Petal Width* (or five, if we consider the class being the target feature as discussed in the first chapter). For the sake of simplicity, we are going to use only two dimensions: *Sepal Length* and *Petal Length*. The dataset is visualized in Fig. 3.4. From the figure, one can see that *Iris setosa* (green triangles) samples are easily distinguishable from the rest, whereas when it comes to the *Iris Versicolor* (red circles) and *Iris Virginica* (blue squares), it is not that easy. In mathematical terms we could say that *Iris Setosa* is *linearly separable* from both the *Iris Versicolor* and *Iris Virginica* samples, which means (without going deeply into the details) that one could easily draw a straight line on Fig. 3.4 to have all the *Iris Setosa* samples on the one side of the line and all the others on the other side. On the other hand the *Iris Versicolor* and *Iris Virginica* species are ***not*** *linearly separable* because it is impossible to draw a straight line to separate those two classes using only these two features (see Sect. 1.6.3 if you are interested to know what to do in case that linear separability is not possible).

Our possible goal is uncover from the data the number of natural clusters present, which in turn in order to create a clustering outcome that would be the most informative. Such a clustering can then help us to *classify* unknown data samples as belonging to one of these classes. Since we know the target feature (what is their species), we expect to have some correlation between the result of our clustering and the target feature values. If there is total agreement, perhaps we would be to have only three clusters with 50 samples in each one.

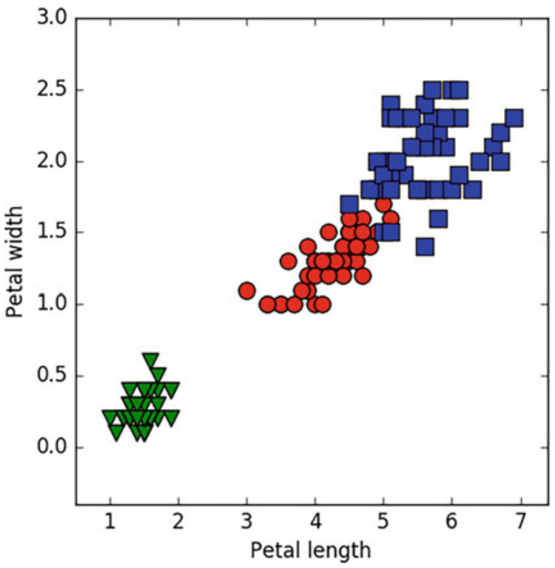


Fig. 3.4 Visualization of the Iris dataset. Each shape represents different species. Green triangle—*Iris Setosa*, red circle—*Iris Versicolor*, blue square—*Iris Virginica*

The quality measure we can use is *Within Cluster Sum of Squares* (*WCSS*) that is defined as

$$WCSS = \sum_{C \in \mathcal{C}} \sum_{x \in C} d\left(x, x\left(C\right)\right)^2, \tag{3.8}$$

where C is a particular cluster, $\mathcal{C}$ is a set of all clusters, x is a particular element attributed to cluster C, $x(C)$ is a centroid (central point) of cluster C and $d(x, x(C))$ is the distance measured between point x and $x(C)$. As a distance measure, for this problem we use the *Euclidean Distance* (Eq. (3.1)), but depending on the application others could be used, e.g. *Manhattan Distance* or *Cosine Similarity Measure*. Having defined the quality measure, the answers to three questions stated in Sect. 3.3.1 are straightforward.

1. This measure promotes clustering, where points in every cluster are close to the cluster centre (dense clusters) so, as a consequence, the clustering result with only one-object clusters is the most desired solution from the measure perspective. However, from the user perspective, in most of the cases, it is not desired because this does not give any knowledge about the data. Furthermore, this measure ignores the separation between clusters or how the clusters are distributed in the feature space. Using this measure, we do not care about the spatial relation between clusters which might sometimes be important (as stated in Sect. 3.4.1). Also, the number of clusters is not taken into consideration by this measure, so more clusters will, in general, give a better score since it gives a better fit into data, lowering the average distance of points to their centres.

2. The boundaries are $[0; \infty)$, where $WCSS = 0$ can be reached in a situation, where there are no clusters or every point is in its own cluster. Any other situation should end up with a positive value.
3. In general, we would like to *minimize* the value of the measure, because the lower the value is, the denser and more consistent the clusters are. However, on the other hand, we would like to avoid the situation when $WCSS = 0$.

After giving these answers, we can conduct a series of experiments (using k-means, for instance) with different parameter values (for k) that can vary from one up to four. The results, together with the measure value, are presented in Fig. 3.5. From this figure, we can see that for $k = 1$ there is only one cluster containing all the samples. Because of this, there are many objects that are far from the cluster centre, and because of that, the $WCSS$ value is the highest. On the other hand, when $k = 2$, the result correlates with our understanding that there is one group of samples that is highly different than the other. We point out, however, that one sample near the centroid of the blue group has been attributed to the other group. This shows a divergence between the result of k-means and our intuition, since we perceive that many "blue elements" are actually closer to that sample attributed to the other cluster.

Focusing only on the value of $WCSS$, the solution with $k = 2$ is preferable over the one with $k = 1$. Despite the fact that the result for $k = 2$ complies with human intuition, we know that experts agree in identifying at least three different iris species, not two. Following that fact, when $k = 3$ the result is similar to the *ground truth* (the expert opinion) shown in Fig. 3.4 except several samples. This is because k-means in its basic form is not capable of separating linearly non-separable groups (as Fig. 3.2 also shows).

Perhaps the most important observation is that, even though the number of created groups and the result of the k-means method is similar to Fig. 3.4, the quality measure $WCSS = 35.39$ is not the lowest of the four. The measure gives the minimum value when $k = 4$ and it is $WCSS = 19.55$, this is obvious when one looks at Eq. (3.8) and our answer to the first question given above. The more the clusters in a clustering result, the shorter the distances of points to their centres become. Since we should minimize the $WCSS$ value, if we follow this criteria we should choose the clustering with $k = 4$. Is it a solution that we are looking for? From Fig. 3.5, we can see that four-cluster solutions just fit additional cluster in the right-upper part of the plot. These three groups together do not look like well-separated groups.

From the above paragraph we can see that the procedure of choosing the best clustering is far from being trivial. The used measure, even though it focuses on the coherence of the clusters, ignores the separation between groups, thus simply promoting results with more clusters. Some researchers [45] say that, when operating with a quality measure with such a tendency, one should plot the obtained quality measures against the value of the parameter that we are tuning. Then one should look for a point giving the biggest change (increase or decrease) on the plot. This point is sometimes referred as a *knee* or *elbow point*, and the corresponding parameter value

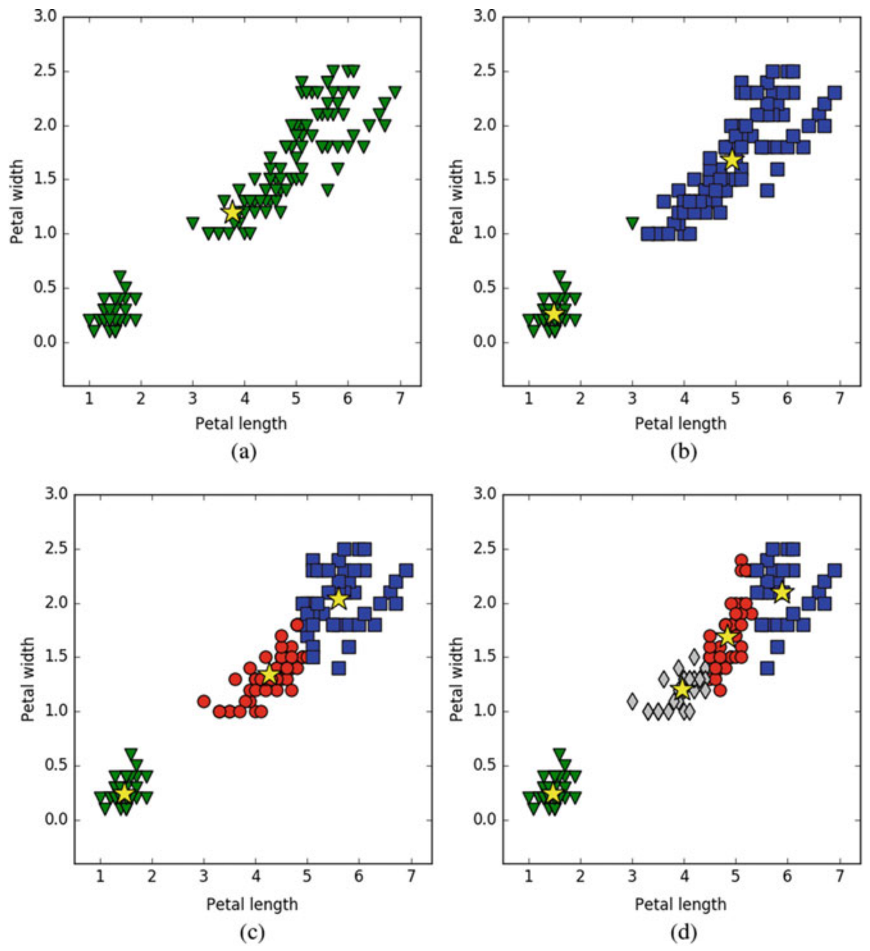


Fig. 3.5 Visualization of k-means results on the *Iris dataset*, where $k = 1, 2, 3, 4$. Different markers represent different clusters, and yellow star indicates cluster centres. The $WCSS$ values are as follows: (**a**) for $k = 1$ the $WCSS = 550.64$, (**b**) for $k = 2$ the $WCSS = 86.40$, (**c**) for $k = 3$ the $WCSS = 35.39$ and (**d**) for $k = 4$ the $WCSS = 19.55$

for that point should be chosen. From the plot for our example, shown in Fig. 3.6, one can see the *knee point* exists at $k = 2$. So, using this technique, this value should be our choice. However, what if there is no significant change in the plot? This could be an indication that there is no clear clustering structure in the data. Due to these problems, in a practical application, it is recommended to consider either using *several* quality measures that focus on different aspects of clustering or find a measure that combines several aspects in one equation [83]. This gives a more comprehensive view on the obtained results. Luckily there are many different quality measures from various groups that try to describe clustering from different perspectives.

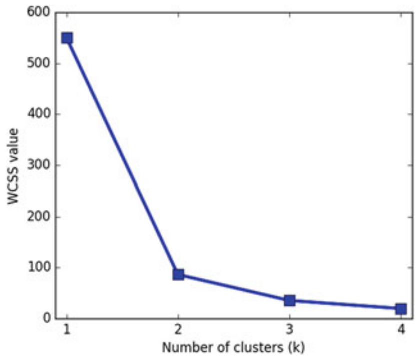


Fig. 3.6 Within cluster sum of squares (WCSS) of clusterization results for different values of k. The knee point is visible for $k = 2$

Moreover, based on the visual assessment of obtained results, and from the existence of the *knee point*, one would select a result with $k = 2$ which is not consistent with the real number of three groups. The answer to this problem is more complex. This situation could be caused by several factors:

- a weakness of the clustering method in discovering the proper shape of the groups,
- a weakness of the quality measure in selecting the clustering that is compliant with true classes assignment,
- the features used to describe samples make it impossible to distinguish between objects from different groups (i.e. the available features and data do not *tell the whole story*).

In most cases, the problem is a combination of all the three factors. That is why, when it is possible, one should choose the final clustering relying not only on the solution indicated by the used quality measure. An inspection of several (e.g. top five) clustering results is highly beneficial. Furthermore, sometimes it could happen that even if we manage to obtain a clustering with a "proper" number of clusters, the clustering does not perfectly match the ground truth. This situation is often caused by the clustering method, and a simple solution is to use several methods, produce a few results and then conduct a detailed analysis of what they would bring in terms of adding knowledge to the user. On top of that, the measures and dataset can simply make it impossible to obtain a clustering that fully complies with our expectations.

To sum up, the problem of choosing the best clustering result is complex. The best result indicated by the quality measure may not necessarily be the best for other people, and it may not even appear to be the best among all obtained results. This stems from the fact that clustering is unsupervised, and the assessment process is both subjective, and application dependent. The best clustering result for one application could be the worst for another.

3.4 Classification of Quality Measures

The central point in the assessment of clustering is the quality measure. Due to this, there are different measures and measure paradigms [45]. Since the number of measures is high, there are several ways to organize them into groups. The main division considers *Internal* and *External* measures. It was made based on whether we have any additional (*external*) knowledge about the data or we base only on the (*internal*) structure of clusters. In most of the cases, the external knowledge is *the ground truth* assignment with which the measures will try to compare the clustering assignment. Several external measures are provided in Sect. 3.4.2. On the other hand, internal measures are focused on the way that the points are clustered. Since we want to have clusters that are both compact and separated from each other, the internal measures often balance these two requirements. Example internal measures are provided in Sect. 3.4.1. The presented measures are a subjectively chosen representation, based on how frequently they appear in the literature.

It is also worth noting that there are other different ways to divide measures into groups. In addition to *internal* and *external* measures, one can also distinguish measures for *crisp* or *fuzzy* clustering or *hierarchical* clustering measures.

In the following sections, we will present the most commonly used quality measures in the field of cluster analysis. In order to make them easily readable all presented equations will obey the following variable definitions:

X	the set of all data points,
$\mathcal{C}$	the set of all clusters in a particular clustering,
C	a particular cluster in $\mathcal{C}$, it can have an additional lower index indicating which cluster it is, e.g. C_j is the jth cluster,
X_C	the set of objects in cluster C,
x	a particular object from a cluster, it can have an additional lower index indicating which point it is, e.g. x_j is the jth observation,
$d(a, b)$	distance between objects (or centroids) a and b,
K	set of all classes (i.e. from a *ground truth* assignment),
k	a specific element from K (a class label),
X_k	set of all objects of class k,
$\overline{X_C}$	the centroid of a cluster C,
$\lvert S\rvert$	the number of objects in a set S.

3.4.1 Internal Measures

One of the simplest internal measures is *Within Cluster Sum of Squares* (*WCSS*). It was used in Sect. 3.3.2, and its formula is in Eq. (3.8). However, as shown in the mentioned section, its main drawback is that it focuses only on the *compactness* of clusters, ignoring the *separation* that exists between pairs of clusters. Another simple measure is *Within-Between Index (WBI)*. It is formulated as follows:

$$WBI = \frac{\max\limits_{C \in \mathfrak{C}}\{\max\limits_{x_i, x_j \in X_C}\{d(x_i, x_j)\}\}}{\min\limits_{C_a, C_b \in \mathfrak{C}, a \neq b}\{d(\overline{X}_{C_a}, \overline{X}_{C_b})\}}. \tag{3.9}$$

It expresses the ratio between compactness (in the numerator) and separation (in the denominator). The compactness of the clustering is measured by the distance between the farthest pair of points belonging to one cluster. On the other hand, the separation part is the minimum distance from cluster centroids among all the possible pairs of clusters. Therefore, this measure should be minimized, and its values fall into $[0; \infty)$.

Another quality measure that utilizes the compactness–separation ratio is *Dunn Index (DI)* [32]. It has several different forms in the literature [6] providing different characteristics, e.g. robustness to noise. One exemplar form is:

$$DI = \frac{\min\limits_{C_a, C_b \in \mathfrak{C}, a \neq b}\{\min\limits_{x_a \in X_{C_a}, x_b \in X_{C_b}}\{d(x_a, x_b)\}\}}{\max\limits_{C \in \mathfrak{C}}\{\frac{1}{|X_C|(|X_C|-1)}\sum\limits_{x_i, x_j \in X_C, i \neq j} d(x_i, x_j)\}}, \tag{3.10}$$

that expresses the ratio between the closest clusters, expressed as the shortest distance between points from different clusters, in the numerator (separation) and the biggest pairwise distance in between points in a cluster in the denominator (compactness). Moreover, the value in the denominator is averaged guaranteeing that the size of the cluster is not taken into account. The value of DI varies from 0 to ∞, and the index should be as high as possible.

Both of the measures WBI and DI promote solutions with dense, well-separated clusters. It is also worth noting that they concentrate on the worst aspects of a clustering. They utilize the least separated clusters and the least compact one. It follows the rule: *a clustering is as good as its weakest part*. However, sometimes this may not be the case. In such situations, one could use *Davies–Bouldin Index (DBI)* [23, 27] or *Calinski–Harabasz Index (CHI)* [14, 27]. They take into account the sum or average of all the clusters but then it suffers from the fact that their result is biased when assessing clusterings with varying diameters. Other internal measures that are worth considering are *Silhouette Index (SI)* [95], or more recently *SD Index* [44], *S_Dbw* [46] or *Clustering Validation Index Based on Nearest Neighbours (CVNN)* [73]. Additionally, some internal measures have been updated in order to perform better, e.g. DBI [64].

3.4.2 External Measures

External measures commonly use a *ground truth* cluster assignment. In the previous example, the three species of Irises is a user-defined characteristic (target feature). External measures would then use this labelling to evaluate the quality of a clustering method.

3.4.2.1 Confusion Matrix

The external measures can be divided into several types. One group relies on the computation of a *confusion matrix (error matrix)* that serves as a data structure to quantify the performance of a classification method. This group is sometimes known as *pair counting* methods [93]. The confusion matrix consists of four numbers called the *true positive*, *true negative*, *false positive* and *false negative* rates. They can be calculated by considering all possible pairs of objects in the sets and their assignments in the clustering result. We will also use the ground truth information about those objects [27]. In order to calculate the rates above, one has to consider all $\frac{n(n-1)}{2}$ possible pairs, where n is the number of objects. Assuming that the clustering result we want to assess is $\mathcal{C}_a$ and the ground truth grouping is $\mathcal{C}_{gt}$ one could compute

- *True positive (TP)* as a number of pairs that belong to the same cluster in both $\mathcal{C}_a$ and $\mathcal{C}_{gt}$,
- *True negative (TN)* as a number of pairs that **do not** belong to the same cluster in both $\mathcal{C}_a$ and $\mathcal{C}_{gt}$,
- *False positive (FP)* as a number of pairs that belong to the same cluster in $\mathcal{C}_a$ but **do not** belong to the same cluster in $\mathcal{C}_{gt}$,
- *False negative (FN)* as a number of pairs that **do not** belong to the same cluster in $\mathcal{C}_a$ but belong to the same cluster in $\mathcal{C}_{gt}$.

The *TP* and *TN* are expressing the ability of our clustering method to properly group objects that should possibly be together and separate objects that are assumed to belong to different groups. We can call them *good decisions*. On the other hand, *FP* and *FN* are showing how many “mistakes” were made by our model. In Statistics, *FP* is often referred to as the number of *type I errors*, and *FN* as the number of *type II errors*.

Having computed all the values of the confusion matrix, one can calculate different external measures, e.g. *Specificity* (*True Negative Rate*), *Accuracy*, *Precision*, *Recall*, *F-Score* [107], *Jaccard Index*, *Rand Index*, *Fowlkes–Mallows Index* [40]. All of these indices can be found in [27]. Another external measure worth to be noted is *Matthews Correlation Coefficient* (*MCC*) [79] with its later generalization into multi-class version in [43, 58]. The advantage of using this index is its invariance to different class sizes. An example of using this measure in the marketing domain is presented in Chap. 20 of this volume. The most commonly used quality measure is the *F-Score* [101], also known as *F-Measure*. It is the harmonic mean of the *Precision* (*Prec.*) and *Recall* (*Rec.*) (also known as *Sensitivity* or *True Positive Rate*) and it is computed as follows:

$$Prec. = \frac{TP}{TP + FP} \tag{3.11}$$

$$Rec. = \frac{TP}{TP + FN} \tag{3.12}$$

$$F\text{-}Score = \frac{2 * P * R}{P + R} = \frac{2 * TP}{2 * TP + FP + FN}. \tag{3.13}$$

The bounds of the *F-Score* is the closed interval [0, 1] and the larger the *F-Score* gets, the more similar the model is to the ground truth. The intuition behind formula (3.13) is that when clustering, we would like to get high precision and recall, but since these two measures are in contradiction to each other, we will take a harmonic mean of them.

One notable fact about the *F-Score* is that it does not explicitly depend on *TN*. The consequence of this is that the assessment is positively influenced only by *TP*, so bigger clusters impact the measure more. Measures that have an explicit dependence on *TN* include the *Rand Index* [40] (a variant which is called the *Adjusted Rand Index* was employed in the network alignment study of Chap. 12 and in [83]).

3.4.2.2 Inter-Rater Reliability

Another group of external measures is connected with the statistical concept of *Inter-rater Reliability* (*Inter-rater Agreement*) [47]. The intuition behind that is connected with a situation when several raters (e.g. psychiatrists) assessed *subjects* (e.g. patients) into predefined *categories* (e.g. diseases). In such scenario a relevant question is about the agreement among raters. In other words how consistent the psychologists are in their diagnosis. Based on their assessments it is possible to build a *Contingency Table* [67] which is a generalization over a confusion matrix used in the previous subsection. In such a table the rows relate to cases (*subjects*), the columns to mental disorders (*categories*) and the elements are the number of raters that classify corresponding subject to the corresponding category. Based on such a structure one can compute the number of statistics measuring agreement among raters counting factors like agreement occurring by fortune. If there are only two raters, the *Scott's Pi* [98] or the *Cohen's Kappa* [21] could be computed. In a case of more than two raters the *Fleiss' Kappa* [38], which is based on Scott's Pi, is recommended.

Analogously, in clustering, given a clustering result and the ground truth the contingency table can be built in a way that the columns represent clusters from ground truth and the rows represent obtained clustering result groups. The elements of the matrix are the number of objects that are the same among particular groups. In such a situation the statistics from the previous paragraph can indicate how similar the obtained clustering is to ground truth. An example of how to use Inter-rater reliability indices is shown in [25].

3.4.2.3 Purity

The other group of external measures worth noting is based on the concept of *cluster purity*, that is, they focus in how homogeneous the created clusters are. A pure cluster is one that contains only points belonging to one class in the ground truth model. Cluster purity can be calculated as shown below

$$CP = \frac{\sum_{C \in \mathcal{C}} \left(\max_{k \in K} \{|X_k \cap X_C|\} \right)}{|X|}. \tag{3.14}$$

For every cluster from our result, this measure tries to find a class from the ground truth model which shares the largest number of points with that cluster. This number is then normalized by the size of input data to give the value of CP. The boundaries of this measure are [0; 1], and results closer to 1 are preferred. Its main drawback is that this measure is blind to the situation in which the clustering result has more clusters than the ground truth. It does not penalize cases when a class from the ground truth is fragmented into smaller groups in the clustering. This is because it does not take into account what fraction of particular class' points are within the considered cluster. Moreover, cluster purity concentrates only on the points that comply with the ground truth, omitting the points that do not, so one big cluster containing all the data would maximize the CP value.

The main advantage of this type of measure is that we can compare an obtained clustering result with the one that we have as a reference. If, for instance, we need a classification algorithm that relies on the results of a clustering method acting as a subroutine, it is important that the ground truth information guides the clustering algorithm, as in turn may indicate which are the best features to be used for the classification final objective. In those circumstances, external measures become really important.

There are circumstances, however, in which obtaining the ground truth is hard, expensive, or not possible for a percentage or even all the samples, e.g. when we know nothing about the ground truth clustering, and the task is to actually *discover* these groups (like in the marketing segmentation study of [26]). Additionally, there are also circumstances in which we should take extreme care when creating a ground truth model, since its form will influence the rest of the research.

3.4.3 *Other Measures*

There are other groups of external measures. One example are measures that originate from the *Theory of Information* [99] and are based on the concept of *Entropy*. Examples of such measures are *Normalized Mutual Information (NMI)* (used in the study of Chap. 9), *Information Gain (IG)*, or more recently *Variation of Information (VI)* [80] and *Confusion Entropy Confusion Entropy* [58].

Another group is the *Set Matching* indexes, where the purpose is to match groups from ground truth to the created groups in our model [93]. Example measures within this group are *Normalized Van Dongen (NVD)* [108] or *Pair Sets Index (PSI)* [93]. Additionally, the *Purity* and *F-Score* measures can also be viewed as *Set Matching* measures [88].

3.5 Some Partitional Clustering Methodologies in More Detail

Now that we have provided a general introduction to clustering, its key aspects and how to check quality measures we take a deeper look at some specific clustering methodologies. Specifically, most commonly known k-means algorithm is presented followed by k-Nearest Neighbour (k-NN) approaches and a variant of the k-NN. Throughout, we have also focussed on finding those business and marketing applications and domains in which these methods are heavily used and championed.

3.5.1 *The k-Means Approach*

It is safe to say that the k-means algorithm (first proposed in the 1960s [39, 76]) is one of the best known (and most used) partitioning clustering algorithms. The k-means method is still very much used today and one main reason for this is that it is also one of the simplest partitioning techniques available [56]. The algorithm finds a partition such that the squared error between the empirical mean of a cluster and the points in the cluster is minimized. With a k-means algorithm the user has to set a value for k a priori which will be the number of clusters the algorithm will divide the dataset in. K-means then selects k random points which are called cluster centroids (or *seeds*). Therefore, if k is set to 4 (as in the Iris dataset experiment described before), the number of centroids will be four. Then, the algorithm goes through each of the data points and it assigns each data point to the centroid that it is “closer” to (either in a graph or based on the distance matrix). Next, the algorithm calculates the cluster average for each cluster and moves the cluster centroids to the cluster average location. This action is repeated (iterated) until there are no further changes made to the clustering (or until an alternative stopping condition is met). Figure 3.7 shows the k-means algorithm graphically in a very simple example. Here we can easily visually see that the data has four natural groupings in Fig. 3.7a and how the centroids are moved closer to the cluster average in Fig. 3.7b.

The basic k-means algorithm has a few simple steps. They are shown in the algorithm below.

Algorithm 1: The basic k-means algorithm

```
    Input  : A set of points equipped with a distance metric.
    Output: A set of k clusters.
1   Select k points as the initial centroids.
2   repeat
3       Calculate the distance between each data point and cluster centroids.
4       Form k clusters by assigning all points to the cluster that corresponds to their closest
          centroid.
5       Recompute the centroids for each cluster.
6   until The generated clusters stop changing.
```

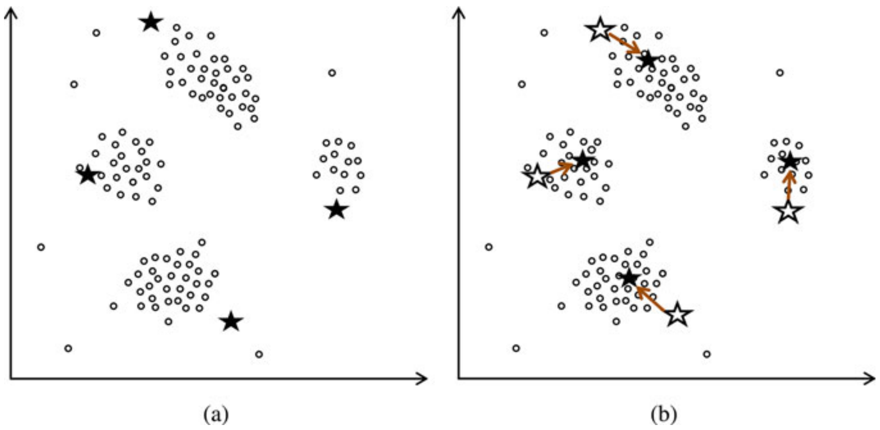


Fig. 3.7 Here we can see the k-means algorithm simplified to one simple step. In (**a**) the initial centroids/seeds (shown by stars) are randomly selected by the k-means algorithm and in (**b**) they are moved closer to the cluster average location. This step is repeated iteratively until all the squared errors between the empirical mean of a cluster and the data points in that cluster are minimized. Usually this process will be continued until no more changes are made or alternatively another stopping criteria such as the running time of the algorithm

Although *k*-means is a quick, efficient and simple clustering algorithm, this method is stochastic in nature and has known several disadvantages. Firstly, the a priori selection of *k* by the user constricts the quality of the algorithm's outcome to this user-determined parameter [56]. Further, *k*-means tends to have problems when the underlying clusters are of different sizes, densities or when they are non-globular shapes. The example in Fig. 3.2 already showed an instance in which *k*-means clustering would not be able to find the naturally occurring clusters in the two-dimensional graph space due to their non-globular shapes, density based separation and different sizes. Furthermore, *k*-means has problems when the data contains large outliers due to the fact that it is based on the arithmetic mean of data points [56]. Several large outliers could significantly skew the clustering outcome as they will alter the cluster average and this would have an adverse effect on the rest of the clustering outcome as *k*-means aims to minimize the squared error between the mean of a cluster and the points in that cluster. One more disadvantage of the *k*-means algorithm is that it can only be used for numerical datasets (i.e. not for categorical information).

Having recognized its disadvantages, it is still reasonable to consider *k*-means as one of the easiest to use clustering algorithms and the base for many other approaches of clustering. A variant of the *k*-means algorithm that was developed in order to deal with the disadvantage of being limited to numerical data is the "k-modes" approach [18]. As the name suggests, this algorithm takes the modes instead of averages. This means that it can be used for categorical data, or data of mixed types, and it is also a lot faster. With the increase of data mining throughout the years and the increased adoption of data analytics methods by the social sciences, categorical datasets and datasets of mixed types have become a lot more common. Variations to the *k*-modes have already previously been published such as Hartigan's method for *k*-modes [115]. Besides, *k*-modes, there are many other variations of the *k*-means algorithm including k-medoids (PAM) [91], CLARA [60, 61], CLARANS [86] among many others. It is likely to see many more variants to the *k*-means method to be developed and brought forward as research and data analysis capabilities continue to grow and expand.

3.5.1.1 *k*-Means in Marketing and Business Analytics

As stated, the *k*-means algorithm is by far one of the most widely used clustering algorithms and it is implemented in many different analysis software. Researchers from many different domains use *k*-means either as a sole clustering analysis, or as a comparison method to their own new algorithms. This also counts for the field of marketing and business analytics where many applications can be found using k-means. Most commonly, marketing researchers likely use the k-means algorithm to cluster customers, whether it is in tourism applications [4, 30, 59], banking applications [77, 84, 106, 109, 114], telecommunications [71] or customer behaviours relating to their weightloss and beauty preferences [55], the focus is to cluster consumers and more accurately target market for each of them.

The paper by Kau and Lim [59] provides an interesting example of k-means clustering used in a tourism application. They investigated the motivations of Chinese tourists who travel to and visit Singapore. Their study shows that a technique as common and as simple as the k-means algorithm can provide great insights into an industry application such as investigating tourism motivation for Singapore's third largest tourist generating country. Another marketing paper using the k-means method is that of Kleijnen et al. [66] who investigated consumers' adoption of wireless technologies in the earlier 2000s. In their study they found three different segments of consumers when it comes to the adoption of new wireless services and products showing that manufacturers and brand managers of these products can (and should) target these consumers differently.

Besides clustering and segmenting consumers, the k-means algorithm is also used in other business applications. For instance, Nanda et al. [85] conducted a clustering analysis comparing k-means with fuzzy c-means and other methods analysing Indian stock market data with the purpose of improving portfolio management. The idea of their study was to select the optimal combination of stocks to create a portfolio where portfolio risk is minimized and compared to the benchmark index. Their study is a good example of the k-means algorithm being used in a financial application domain and providing insights for stock market traders and investors.

3.5.1.2 Variations of k-Means Algorithm

New advances to the k-means algorithm are made using marketing and business applications. For instance, Kim and Anh introduce a Genetic Algorithm (GA) optimized approach for k-means clustering [55] when investigating demographic and behavioural information of Korean consumers related to health, beauty and weight characteristics about themselves. Other publications also saw researchers including and integrating Self-Organizing Feature Maps (SOMS/SOFMS) together with k-means clustering to improve market segmentation [68, 69]. SOMS are a type of artificial neural network that provide a discretized representation of the input space (which is called a map). Battiti and Brunato [5] provide a good introduction on SOMS in Chap. 14 of their book *The Lion Way: Machine Learning Plus Intelligent Optimization*[1] for those readers wishing to learn further about SOMS.

Other ways in which researchers have attempted to improve the k-means clustering method are by combining approaches, through, for instance, generating a hybrid approach. Wang et al. [110] propose the "K-means SVM (KMSVM) algorithm" in which Support Vector Machines (SVM) and the k-means algorithm are combined to generate better results for real-time business intelligence systems. Niknam and Amiri propose a hybrid approach combining k-means with FAPSO (fuzzy adaptive particle swarm optimization), ACO (ant colony optimization) called FAPSO-ACO-K [87].

Another example of making improvements to the clustering outcome of k-means comes from researchers combining variable selection (or weighting) techniques with the k-means algorithm. For instance, Carmone et al. also looked at the issues surrounding how to weight the variables or pre-select them for clustering [15]. They propose a new algorithm termed HINoV (the Heuristic Identification of Noisy Variables) to solve these issues and find that clustering when implementing HINoV improves the clustering outcome in terms of stability and robustness. More recently, Brusco and Cradit [13] also proposed a variable-selection heuristic for nonhierarchical (k-means) cluster analysis with the objective to include variables that truly define cluster structure and eliminate those that do not or even mask the structure. They applied their method on financial data and found that the method including a variable selection step to uncover variables that mask the structure provided an outcome with greater cluster stability than a simple clustering approach without variable selection. Steinley and Brusco [103] later built on this method proposing a variance-to-range ratio variable weighting procedure.

As this book highlights, marketing and business researchers and practitioners are increasingly adopting newer, better and more computationally complex approaches. Another example is the work by Liu et al. [74] who incorporate a multi-objective algorithm in their clustering approach combined with k-means. They propose MMSEA (a Multi-criterion Market Segmentation algorithm) and apply it on a cellphone network provider dataset (similar to the one analysed in Chap. 20 of this book) and a retail customer dataset. One of the main benefits of their method is that no multicriterion aggregation or trade-offs (of objectives) are required before the users see the full spectrum of the solution space allowing for greater flexibility and improved business decision making.

As can be seen, applications and research from the marketing and business fields have brought forward many k-means contributions. Here we have only "scratched the surface" of presenting studies that may use (some form of) the k-means algorithm; however, we have provided some ranging examples and a basic

[1] http://intelligent-optimization.org/LIONbook/.

understanding of this hugely popular approach. Further, Table 3.1 later in this chapter shows a small survey of clustering methodologies in consumer analytics and business applications for further reading. We focus on studies that are published between 2000 and 2017 in order to provide an up-to-date view of the field and so to avoid too much repetition with the survey and review works of Jain [56], Punj and Stewart [92], Steinley [102] and others.

3.5.2 Who Is Your k-Nearest Neighbour?

We have already highlighted the difference between supervised classification methods and unsupervised clustering methods. k-Nearest Neighbour (kNN) approaches can be said to be a "bridge" between these two methods. They combine parametric approaches that need a priori knowledge of the distributions underlying the data, and non-parametric approaches that assume the functional form of the discriminant surfaces partitioning the different pattern classes [22]. The kNN approach has the basic principle that an unknown entity (object in the data) is best to be assigned to the category (or cluster) to which it is closest to in a suitably defined information space (dataset) through an appropriate metric (i.e. using a distance matrix).

When k is not explicitly defined, k-NN techniques assume that $k = 1$ for the approach. That is, node a is connected to node b if b is one of node a's nearest neighbours (in the general case there may be more than one), or if node a is one of the nearest nodes of b. However, often, researchers use a k-NN approach in which the value of k will be set to suit the specific requirements. Consequently, the value of k has a significant effect on the density of the clustering result. Where a lower value of k is selected, nodes will only be connected to one or two other nodes. However, if the value of k is fixed, e.g. $k = 4$, then each data point will be connected to its $k = 4$ nearest neighbours. This is illustrated in Fig. 3.8, where on a small simulated dataset $k = 1$, $k = 3$ and $k = 6$ nearest neighbour graphs are computed and shown in the subfigures Fig. 3.8a–c, respectively.

Due to the significant effect of the value of k on the clustering outcome, there is a lot of debate about finding an "automatic" selection of k or finding the "optimum" number for the particular instance at hand. As we stated at the start of this chapter, the number of a priori parameters set by the user should ideally be kept to a minimum with unsupervised learning as this allows results to be completely data-driven.

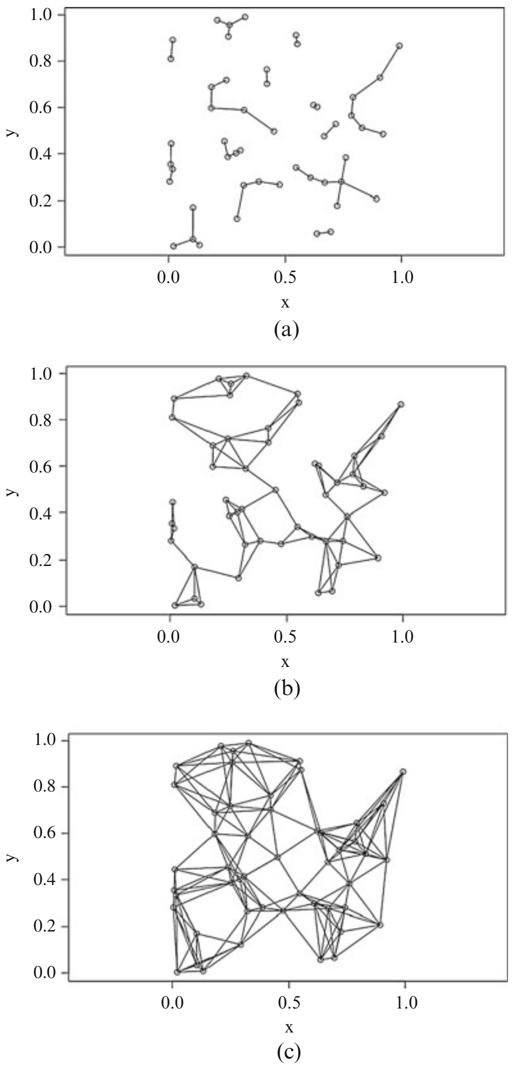


Fig. 3.8 In this figure, the effect of selecting a different value for k on the resulting nearest-neighbour graph is shown. The graph becomes a lot less or a lot more dense depending on whether nodes are connected only to their 1-nearest neighbour, their 3-nearest neighbours or to their 6-nearest neighbours. (**a**) Shows the results of running a 1-NN algorithm on a random simulated graph, in (**b**) the value of k is set to 3 and in (**c**) k is equal to 6. As can be seen in this image, the value of k has a large impact on the outcome of the kNN algorithm and thus the resulting clusters. In (**a**) we still have many partitioned small clusters, whereas when k is increased to 3, the graph is already one completely connected network and when $k = 6$, it becomes quite a densely interconnected graph

One common approach is through the use of a type of validation process (for example, cross-validation or leave-one-out). Generally, as the value of k increases, error would decrease until it stabilizes and then starts raising again as k is further increased. The rule-of-thumb is then to set k at the start of the "stable" zone in the error curve. Another rule-of-thumb approach to selecting a value for k, sometimes used in some machine learning scenarios, is to take the square root of the number of training patterns/samples (n) as this would lead to better results [31].

It is difficult to give a generic mathematically well-principled answer to which would be the best approach to select the value of k. It may be an ill-posed task because it depends not only on the problem, but also on the problem instance/input we are working with, the metric being used and other considerations. For instance, in [83] the authors calculate graphs with k ranging from 1 to 10 and then they apply their new methodology that makes use of the Normalized Mutual Information (NMI) and Adjusted Rand Index (ARI). They report results with all these graphs and then particularize the discussion on the one that according to their proposal maximizes the product of both indexes (which turned to be $k = 5$ in their study).

3.5.2.1 Introduction to the MST-*k*NN

Recently, one of the editors of this book and his colleagues presented a new clustering methodology that uses the generation of a k-NN graph combined with a sparsification of the *Minimum Spanning Tree* (MST). They named this approach the MST-kNN [53] method. Since we will be using this approach in subsequent chapters, we introduce it here. For further details of proximity graphs that are supersets of the nearest neighbourhood graph, we refer to Chap. 4. The MST-kNN method has been tested on comprehensive studies on large-scale biological weighted networks and it has been successfully applied in various areas, see, for instance, Arefin et al. [3].

The MST-kNN algorithm has led to results that seem to be superior to known classical clustering algorithms (e.g. k-means and SOMs) in terms of homogeneity and separation [53, 54] in spite of not using an explicitly defined objective function. Due to its characteristics, it performs well even if the dataset has clusters of different mixed types (i.e. MST-kNN is not biased to "prefer" convex clusters or when the data has clusters that are embedded in subspaces of different dimensionalities). Most importantly, the MST-kNN algorithm scales very well, allowing the possibility that the methods that are based on it can be extended to the analysis of very large datasets. This opens a door for new methods in marketing that involve the analysis of datasets with millions of samples, e.g. as those arising from online behaviours, products, web pages, etc. A Graphics Processing Unit-based implementation has been made available [2].

The MST-kNN approach is basically a constructive heuristic that is not biased for the choice of a particular objective function, yet it provides a strong guarantee of optimality of a property of the final solution [53]. We explain this property after we explain the algorithm. First, the algorithm's input can be either a distance matrix between all pairs of nodes or a weighted graph. As an example, we take a dissimilarity matrix that is computed from the Spearman rank correlation matrix as the input for the algorithm (as is done in Chap. 5). Formally, if $r(a, b)$ is the Spearman rank correlation between two objects (nodes) a and b over a set of features, then the corresponding distance matrix $D = [d(a, b)]$ with each coefficient is calculated as $d(a, b) = 1 - r(a, b)$. Given this input matrix D, the output of the MST-kNN algorithm is a forest. This means that the MST-kNN generates a partition of a set of nodes given as an input using the information of similarities/dissimilarities between each pair.

We mentioned that the algorithm returns a forest that satisfies a property. The set of nodes are the ones that are part of the input. In the forest given as output, any edge of the forest that connects two nodes does so if the edge is one of the edges of the minimum spanning tree ($MST(G)$) and, at the same time, it is also an edge present in the set of edges of the k-nearest neighbour graph ($kNN(G)$). The k-NN graph is the graph that has one node per object and that has an edge between each pair of nodes, for example, a and b, if either a is one of the k nearest neighbours of b or if b is one of the k nearest neighbours of a, or both. We note that edges of the minimum spanning tree are not bound to have this property regarding "k-neighbourness". The addition of this extra constraint has the effect of disconnecting the MST, thus creating a multi-tree forest and consequently leading to a natural partitioning of the set of nodes.

There are several variations of this scheme. In one of them, the value of k is set up to a relatively large value which is linked to the total number of nodes, and then, when the MST is fragmented in different components, a different value is selected for the different connected components using the same formula but now having for each of the connected components the number of nodes in each of them as input, thus leading to different values of k for each component. Another approach is when a value of k is fixed or when multiple values for k are trialled as done in de Vries et al. [24]. The MST-kNN will reappear in subsequent chapters in this section where outcomes of the method can be found and interpretations are explained.

An example of the MST-kNN method has been shown in Fig. 3.9. In this figure, the Wine Qualities dataset (also discussed and presented in Chaps. 16 and 26) has been clustered using the MST-kNN approach and visualized using red colours for "bad" quality wines and green colours for "good" quality wines. The quality measure is a rating given by a wine connoisseur and the features are physico-chemical properties of the wines (for instance, sugar level, acidity level, alcohol level, etc.). The figure shows the properties that we have explained of how a Minimum Spanning Tree is further subdivided leaving separate trees only connected if they are nearest neighbours. This figure shows that "bad" and "good" quality wines cannot easily be separated and wines may be similar or dissimilar based on other values. It shows that wine quality is rather heterogeneous in nature and perhaps also indicated the human subjectiveness in wine quality compared to physical properties of wine.

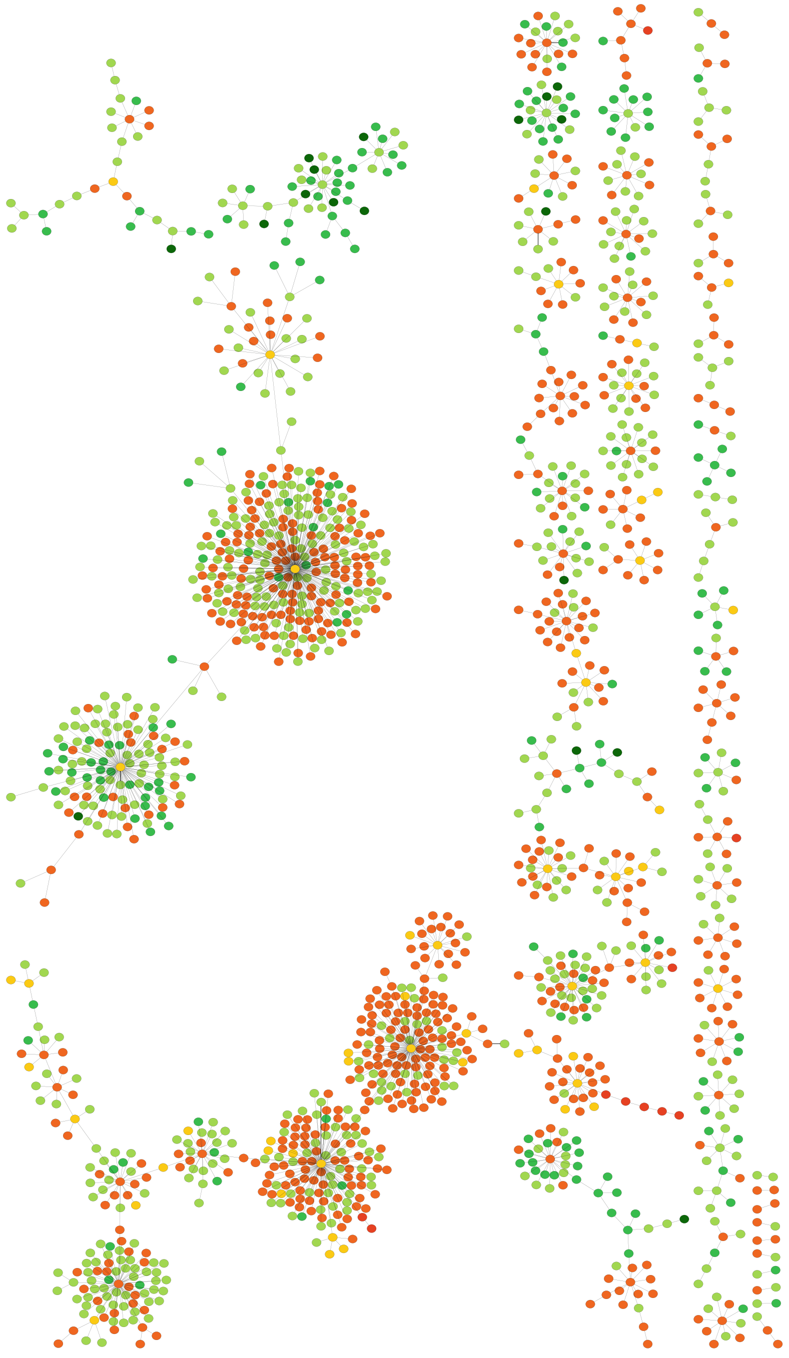


Fig. 3.9 Outcome of the MST-*k*NN algorithm on the Portuguese wine quality dataset (see Sect. 16.4 for details). Colours indicate the quality of the wine, as evaluated by specialists, showing the heterogeneity of responses between clustering outcomes and quality

3.6 All Things Fuzzy

Fuzzy clustering is also referred to as "soft clustering" as each object has a level, or percentage, membership to more than one cluster rather than a "hard division" between clusters. Soft clustering produces a membership clustering outcome [65], is commonly known as *fuzzy clustering* and is based on *fuzzy logic* [104]. With roots in control theory and artificial intelligence, fuzzy clustering is a relatively new approach in the marketing literature. Clustering methodologies used by marketers and consumer behaviour researchers encompass more "typical" clustering approaches such as k-means clustering. New studies, however, are seeing an improvement in market segmentation using "fuzzification" methods [16]. In this volume Chap. 22 provides an example of the use of a fuzzy clustering approach in a tourism application. In addition, another contribution presented in Chap. 24 uses fuzzy logic to analyse data from a tourism sustainability application.

Fuzzy logic and fuzzy sets, including fuzzy clustering, is a relatively well-known field among applied mathematicians and computer scientists with a journal dedicated to the topic since 1978 (i.e. *Fuzzy Sets and Systems*). It has been recognized in Marketing as a tool for market segmentation since the 1980s as well (see, for instance [1, 49] or [111]), but with the catalytic growth in online applications, advanced analytical approaches such as fuzzy clustering are able to make significant multi-disciplinary contributions in areas like business and marketing.

3.6.1 Fuzzy Clustering Fundamentals

Fuzzy clustering was first introduced by Bezdek in 1973 [7] and the first work leading to the "Fuzzy C-Means" (FCM) algorithm was developed and brought forward by Bezdek et al. in 1981 in a two-part publication; [8] and [9]. Bezdek used membership function matrices associated with fuzzy c-partitions of X (a set of objects). An Euclidean norm is used and fuzzy clusters are obtained. The actual FCM algorithm (and its FORTRAN coding) was published by Bezdek et al. in 1984 [10]. The FCM algorithm was based on the already popular k-means clustering methodology. In this paper, Bezdek and colleagues use a geological application to illustrate their method. Some other early seminal papers on fuzzy clustering for the interested reader can be found in [32, 96] and [112].

With the fuzzy clustering paradigm, we assume that each object/data point belongs to a cluster with a certain "degree of membership" which is represented as a number in the closed interval [0,1]. Intuitively, data points on the edge of a cluster may have a lower degree of membership than other points of the cluster. A simple example, just for illustration only, of the difference between "hard" clusters and "soft" (fuzzy) clusters is shown in Fig. 3.10. In Fig. 3.10a the clusters are completely partitioned with no overlapping data points being part of more than one cluster. In Fig. 3.10b however, we can see that some nodes (vertices) are part of two clusters.

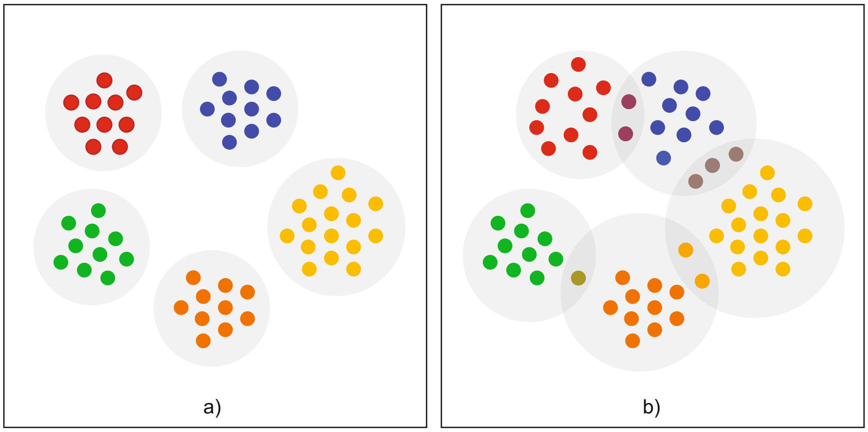


Fig. 3.10 This figure shows the difference between a hard clustering partition and a soft (fuzzy) clustering partition. On the left in figure (**a**) the clusters are clearly separated and defined by their own boundaries. On the right in figure (**b**) however, some nodes belong to two clusters at the same time. They may be 50/50 per cent split between the two clusters, or any other degree between [0,1] with a sum of 1. In this example the nodes that are part of the "fuzzy" outcome only belong to a total of two clusters; however in reality, some nodes may belong, to some degree, to as many clusters there are in the data

In this particular example there are nodes that only have a shared non-zero membership between a pair of clusters. In general, it is possible for some nodes to have a certain non-zero membership in all or in most of the clusters. Consequently, fuzzy methodologies are slightly more complex and often take longer to compute [41].

3.6.1.1 Fuzzy Clustering in Marketing and Business Analytics

As fuzzy clustering attracted more attention, it became apparent that it has many useful applications in different domains. Fuzzy clustering has successfully been used in marketing simply for market definition and segmentation. For instance, Hruschka et al. [49] found that fuzzy methods performed better in segmenting the market than "hard" clustering methodologies in terms of internal validity. They go on to state that in fact, fuzzy partitions provided more insights on segments and markets than their "hard" counterparts and the ease of interpreting outcomes from fuzzy or overlapping results was "satisfactory".

Fuzzy clusterwise regression (FCR) has been used as a benefit segmentation strategy in marketing [111]. As explained in Chap. 1, benefit segmentation separates consumers into groups who are similar to each other in terms of the benefits derived from, and the reasons for using a particular product or service. In this work, Wedel et al. [111] develop a method that estimates the models relating preference to product dimensions within each cluster while estimating the parameters indicating the degree of membership of individuals in these clusters at the same time. Wedel et al. found their fuzzy approach to be a powerful method that uncovers a large amount of useful information. As they state, non-overlapping or "hard" partitioning clustering methodologies ignore the presence of heterogeneity that may be present in a segment and among consumers. This is one of the reasons why fuzzy or overlapping clustering methodologies is increasingly proving to be useful in marketing and consumer analytics domains. A useful review and introduction to various extensions of the fuzzy k-means algorithm are introduced and explained in Ferraro and Giordani [36].

3.6.1.2 Variations of Fuzzy Clustering

One example of a variation to the method that has been introduced is termed the "fuzzy k-modes method". As with its non-fuzzy variant, this method actually tackles some of the problems faced when working with categorical data or mixed data types. Introduced by Huang [51] and Huang and Ng [52] in 1998, it provides us with an alternative of the "standard" fuzzy C-means [33] that can cluster datasets with categorical values as well as those of mixed numerical/categorical natures. As explained by Huang, the k-modes algorithm replaces the means of clusters with modes and uses a frequency-based method to update modes in the clustering process to minimize the clustering cost function. One other advantage outlined by Huang and Ng [52] is that (in their experiments) the fuzzy k-modes algorithm performed much quicker, with less CPU time than the fuzzy k-means algorithm. Similar to the non-fuzzy k-means algorithm, we will likely see many more variants (or completely new methods) of fuzzy approaches to grouping and clustering as the business and marketing fields become increasingly intertwined with computer science and data science approaches.

3.7 Examples of Clustering Techniques in Marketing and Consumer and Business Analytics

As we have already stated, many reviews, surveys and introductions already exist to the field of clustering, segmentation and grouping. However, considering this field is such a fragmented and complex mine field, we have provided the reader with a brief survey of clustering methodologies with applications related to business, marketing and consumer analytics, focussing on those published between 2000 and 2017 for a current view of the field. This small collection of articles is shown in Table 3.1 and provides further reading material for the interested reader.

Table 3.1 A selected sample of some clustering methodologies and applications in marketing from 2000 to 2014 (ordered by year) that use some of the techniques described in this chapter

Application area & year	Clustering technique	Key characteristics
Tourism segmentation of customers to a B&B [4] (2001)	k-Means clustering	The popular k-means algorithm is used to cluster visitors to a Bed and Breakfast using multistate categorical survey data
Segmentation of customers of online music services [90] (2001)	Fuzzy c-means clustering algorithm	Users of online music services are clustered according to the fuzzy c-means and the study provided interpretable results for practitioners
Tourism segmentation using the Austrian National Guest Survey [30] (2004)	Bagged clustering	Bagged clustering is introduced as a new clustering approach for post hoc marketing segmentation drawing benefits from both partitional and hierarchical clustering methods
Segmentation of consumers regarding their adoption of wireless technologies/services [66] (2004)	k-Means clustering	Three different segments of consumers when it comes to the adoption of new wireless technologies were found using k-means providing business insights to better target market to these consumers
Clustering of Chinese tourists based on their motivations for travel in Singapore [59] (2005)	k-Means clustering	The popular k-means clustering algorithm is used on survey data of tourists using many variables related to motivation
Clustering and model-building of customers using credit card consumption data [114] (2005)	Combination of marketing RFM analysis and k-means clustering	Different values of k are trialled for k-means clustering and $k = 6$ is selected as providing the clustering results with the highest level of difference between clusters. Different consumption patterns are found between clusters of consumers
e-Banking customers in Thailand and their usage/motivations patterns [109] (2006)	k-Means clustering, SOMS and marketing RFM analysis (recency, frequency, monetary)	The resistance of Thai customers to adopt internet and e-Banking is investigated using a variety of clustering and grouping methodologies
Clustering customers of a drink company [50] (2007)	Support vector clustering (SVC)	An SVC approach is shown to outperform k-means and self-organizing feature map (SOFM) methods in providing a solid customer segmentation approach
Clustering customers to improve a recommender system based on demographic and personal information, related to weightloss and beauty needs and their related behaviours [55] (2008)	Genetic Algorithm (GA) optimized k-means clustering approach	The input for the k-means algorithm is optimized using GA approaches and compared with simple k-means and SOMS. The findings show that the GA k-means improves the segmentation of customers
Mobile phone provider customer usage behaviour clustering for targeted marketing purposes [71] (2009)	k-Means clustering	600,000 mobile customers are analysed and a real company in China implemented the results and saw their customer base increase by 64% from 2006 to 2007

(continued)

Table 3.1 (continued)

Application area & year	Clustering technique	Key characteristics
Analysing "anti-consumption" (why consumers *do not* buy something) in the case of toy libraries [89] (2010)	Hierarchical clustering followed by k-means	Four groups of consumers were found in terms of their "anti-consumption" behaviour regarding toys and they displayed different motivations and behaviours regarding their use of toy sharing libraries
Clustering using customer relationship information from an Iranian bank [84] (2010)	Combination of marketing RFM analysis and k-means clustering	A new framework for segmenting banking customers is proposed using two stages of analysis: clustering using k-means and use of demographic values followed by the creation of a customer profile for target marketing purposes
Customer relationship systems of airlines for Taiwanese travellers [19] (2011)	Fuzzy k-means algorithm	A fuzzy decision rules approach using fuzzy k-means in its approach is applied in a tourism application. The fuzzy k-means algorithm is used as a step in creating fuzzy decision rules
Clustering customers' requirements for product design [17] (2012)	Genetic algorithm (GA) for fuzzy clustering	The method integrates a fuzzy compression technique for multi-dimension reduction and a fuzzy clustering technique. Subsequently the centre points of market segments are used as "ideal points" for new product development
Analysis of Russian credit institutions (banks) [106] (2014)	k-means clustering and Kohonen's network	Through the clustering analysis the study confirms their expected hypothesis of an institutional misalignment existing in the Russian banking system and possible development ways of the interconnection and interaction between banks and the other economic sectors are suggested, thanks to the clustering outcomes
Wine consumption trends of Italian consumers analysis [29] (2014)	Hierarchical clustering followed by k-means	Three different clusters of consumers were so identified in terms of their wine consumption behaviours and attitudes (occasional and choosy consumers, basic consumers and high quality demanding purchasers)
Analysis of a large study on attitudes towards and habits of food consumption in Germany [82] (2014)	Hierarchical clustering followed by k-means	Their study investigating market segment stability in the German market of food consumption (over the period of 2005–2008) using clustering showed that neither the internal nor the dynamic stability of market segments should be taken for granted. This means that marketers face the challenge of designing segment-specific marketing strategies that allow changes to be made to them to remain flexible and keep up with changing consumer trends and segments

3.8 About Other Chapters That Relate to Clustering and Some Final Conclusions

It is hard to find a chapter in this book that does not have some sort of connection with clustering, either as a pre-processing or a post-processing technique to reveal structure in data. For instance, in the introductory first chapter, when we discussed the case of the US Presidential elections, feature selection techniques allow to “cluster” samples in particular groups (see Table 1.4 in Chap. 1). Analogously, techniques like those for *Frequent Itemset Mining*, described by Cafaro and Pulimento in Chap. 6, allow to “compress” information by identifying samples that share common characteristics, enabling the post-processing of large databases and the identification of interesting clusters in the outputs of itemset mining algorithms. The chapter includes techniques for parallel processing allowing the possibility of using large computing cloud systems and high-performance supercomputers.

Mathieson and Moscato, in Chap. 4, generalize the discussion on k-nearest neighbour graphs, and the MST-kNN algorithm, by considering several other “proximity graphs”, which in turn can help to reveal clusters in the data. In “*Clustering Consumers and Cluster-Specific Behavioural Models*” (Chap. 5), the authors use the MST-kNN algorithm to cluster social media users. Based on the clusters found, consumer behaviours relating to the user engagement with the pages are investigated using symbolic regression analysis powered by the genetic programming techniques introduced in Chap. 1. The authors conclude that these models obtained better “*inform possible personalised marketing strategies after proper segmentation of the customers based on their online consumer behaviour, rather than simple demographic characteristics*”.

Clustering methods in which membership is shared by several sets also are gaining popularity in business and customer analytics. Chapter 22 presents an application of *fuzzy clustering* in tourism analytics, presenting two different types of algorithms for fuzzy data and two empirical case studies. Chapter 21 presents a result on a hierarchical clustering algorithm, powered by metaheuristic-based optimization, on a dataset of hotel ratings. Also on the theme of tourism analytics, Chap. 23 presents a study on the *bundle design problem* which occurs when a company wants to set up offers based on sets of services and they do so based on evidence collected from consumer redemption data. More specifically, they address a problem in which a company manages multiple service providers, each responsible for an attraction, in a leisure park in Asia.

Finally, we have seen with the example presented in Fig. 3.9 that the visual presentation of the result of clustering brings some interesting challenges to computer scientists. Usually displayed in two dimensions, the output of some software currently available does not take into consideration the inter-cluster similarity. In Chap. 16, this issue is addressed presenting different alternatives for the same dataset on wine quality of Fig. 3.9 (see Fig. 16.2). The authors also include a clustering of the set of characters in the Marvel Universe (Fig. 16.1) and of an interesting dataset on customer churn behaviour in telecoms, again showing, like in Chap. 5, the need of establishing cluster-specific models of customer behaviour for service personalization and market segmentation.

In conclusion, clustering "is more than a religion", it is actually a very necessary step for the analysis of data in business and customer analytics. Although our brief chapter cannot cover all possible techniques that exist, we hope it can motivate the reader to further study the methods presented here (as well as in other chapters). We also hope the readers will explore other methods and techniques. The diversity of points of view on "how to group things" is essential in the exploration of the structure present in large datasets.

Acknowledgements We would like to thank Ademir Cristiano Gabardo, Luke Mathieson and Shannon Fenn for their technical and proofreading help with this chapter. Łukasz P. Olech is supported by "THELXINOE: Erasmus Euro-Oceanian Smart City Network", Erasmus Mundus Action-2, Strand-2 (EMA2/S2), project funded by the European Union. Pablo Moscato acknowledges previous support from the Australian Research Council Future Fellowship FT120100060 and Australian Research Council Discovery Projects DP120102576 and DP140104183, and his academic partners in their funded project THELXINOE.

References

1. Phipps Arabie, J. Douglas Carroll, Wayne DeSarbo, and Jerry Wind. Overlapping clustering: A new method for product positioning. *Journal of Marketing Research*, 18(3):310–317, 1981.
2. Ahmed Shamsul Arefin, Carlos Riveros, Regina Berretta, and Pablo Moscato. GPU-FS-kNN: A software tool for fast and scalable kNN computation using GPUs. *PLOS ONE*, 7(8):1–13, 08 2012.
3. AhmedShamsul Arefin, Mario Inostroza-Ponta, Luke Mathieson, Regina Berretta, and Pablo Moscato. *Clustering Nodes in Large-Scale Biological Networks Using External Memory Algorithms*, volume 7017 of *Lecture Notes in Computer Science*, book section 36, pages 375–386. Springer Berlin Heidelberg, 2011.
4. George Arimond and Abdulaziz Elfessi. A clustering method for categorical data in tourism market segmentation research. *Journal of Travel Research*, 39(4):391–397, 2001.
5. Roberto Battiti and Mauro Brunato. *The Lion Way: Machine Learning Plus Intelligent Optimization*. LIONlab, University of Trento, Italy, 2014.
6. J. C. Bezdek and N. R. Pal. Cluster validation with generalized Dunn's indices. In *Proceedings 1995 Second New Zealand International Two-Stream Conference on Artificial Neural Networks and Expert Systems*, pages 190–193, Nov 1995.
7. James C. Bezdek. Cluster validity with fuzzy sets. *Journal of Cybernetics*, 3(3):58–73, 1973.
8. James C. Bezdek, Chris Coray, Robert Gunderson, and James Watson. Detection and characterization of cluster substructure I. linear structure: Fuzzy c-lines. *SIAM Journal on Applied Mathematics*, 40(2):339–357, 1981.
9. James C. Bezdek, Chris Coray, Robert Gunderson, and James Watson. Detection and characterization of cluster substructure II. Fuzzy c-varieties and convex combinations thereof. *SIAM Journal on Applied Mathematics*, 40(2):358-372, 1981.
10. James C. Bezdek, Robert Ehrlich, and William Full. FCM: The fuzzy c-means clustering algorithm. *Computers & Geosciences*, 10(2):191 – 203, 1984.
11. Saprativa Bhattacharjee, Anirban Das, Ujjwal Bhattacharya, Swapan K. Parui, and Sudipta Roy. Sentiment analysis using cosine similarity measure. In *2nd IEEE International Conference on Recent Trends in Information Systems, ReTIS 2015, Kolkata, India, July 9-11, 2015*, pages 27–32. IEEE, 2015.

12. CM Bishop. *Bishop Pattern Recognition and Machine Learning*. Springer, New York, 2001.
13. Michael J. Brusco and J. Dennis Cradit. A variable-selection heuristic for k-means clustering. *Psychometrika*, 66(2):249–270, 2001.
14. Tadeusz Caliński and Jerzy Harabasz. A dendrite method for cluster analysis. *Communications in Statistics-theory and Methods*, 3(1):1–27, 1974.
15. Frank J. Carmone, Ali Kara, and Sarah Maxwell. HINoV: A new model to improve market segment definition by identifying noisy variables. *Journal of Marketing Research*, 36(4):501–509, 1999.
16. Mònica Casabayó, Núria Agell, and Germán Sánchez-Hernández. Improved market segmentation by fuzzifying crisp clusters: A case study of the energy market in Spain. *Expert Systems with Applications*, 42(3):1637 – 1643, 2015.
17. Kit Yan Chan, C.K. Kwong, and B.Q. Hu. Market segmentation and ideal point identification for new product design using fuzzy data compression and fuzzy clustering methods. *Applied Soft Computing*, 12(4):1371 – 1378, 2012.
18. Anil Chaturvedi, E. Paul Green, and Douglas J. Caroll. K-modes clustering. *Journal of Classification*, 18(1):35–55, 2001.
19. Wen-Yu Chiang. Establishment and application of fuzzy decision rules: an empirical case of the air passenger market in Taiwan. *International Journal of Tourism Research*, 13(5):447–456, 2011.
20. Prabhakar Raghavan Christopher D. Manning and Hinrich Schütze. *Introduction to Information Retrieval*. Cambridge University Press, 2008.
21. Jacob Cohen. A coefficient of agreement for nominal scales. *Educational and psychological measurement*, 20(1):37–46, 1960.
22. Belur V. Dasarathy. *Handbook of Data Mining and Knowledge Discovery*, chapter Nearest-Neighbor Approaches, pages 288–298. Oxford University Press, 2002.
23. David L Davies and Donald W Bouldin. A cluster separation measure. *IEEE transactions on pattern analysis and machine intelligence*, (2):224–227, 1979.
24. Natalie J de Vries, Ahmed S Arefin, and Pablo Moscato. Gauging heterogeneity in online consumer behaviour data: A proximity graph approach. In *2014 IEEE Fourth International Conference on Big Data and Cloud Computing (BDCloud)*, pages 485–492. IEEE, 2014.
25. Natalie Jane de Vries, Jamie Carlson, and Pablo Moscato. A data-driven approach to reverse engineering customer engagement models: Towards functional constructs. *PloS one*, 9(7):e102768, 2014.
26. Natalie Jane de Vries, Rodrigo Reis, and Pablo Moscato. Clustering consumers based on trust, confidence and giving behaviour: data-driven model building for charitable involvement in the Australian not-for-profit sector. *PloS one*, 10(4):e0122133, 2015.
27. Bernard Desgraupes. Clustering indices. 2013.
28. Michel Marie Deza and Elena Deza. *Encyclopedia of Distances*. Data-Centric Systems and Applications. Springer-Verlag, 3rd edition, 2014.
29. Giuseppe Di Vita, Gaetano Chinnici, and Mario D'Amico. Clustering attitudes and behaviours of Italian wine consumers. *Calitatea*, 15:54–61, 03, 2014. Copyright - Copyright Romanian Society for Quality Assurance Mar 2014; Document feature - Tables; Equations; Graphs; Last updated - 2014-03-24.
30. Sara Dolnicar and Friedrich Leisch. Segmenting markets by bagged clustering. *Australasian Marketing Journal (AMJ)*, 12(1):51 – 65, 2004.
31. Margaret H. Dunham. *Data Mining Introductory and Advanced Topics*. Pearson Education, 2nd edition, 2003.
32. J. C. Dunn. A fuzzy relative of the ISODATA process and its use in detecting compact well-separated clusters. *Cybernetics and Systems*, 3(3):32–57, 1973.
33. Pierpaolo D'Urso and Paolo Giordani. A weighted fuzzy c-means clustering model for fuzzy data. *Computational Statistics & Data Analysis*, 50(6):1496 – 1523, 2006.
34. Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. pages 226–231. AAAI Press, 1996.

35. Alberto Fernández and Sergio Gómez. Solving non-uniqueness in agglomerative hierarchical clustering using multidendrograms. *Journal of Classification*, 25(1):43–65, 2008.
36. Maria Brigida Ferraro and Paolo Giordani. A toolbox for fuzzy clustering using the r programming language. *Fuzzy Sets and Systems*, 279:1 – 16, 2015. Theme: Data, Audio and Image Analysis.
37. R. A. Fisher. The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7(2):179–188, 1936.
38. Joseph L Fleiss. Measuring nominal scale agreement among many raters. *Psychological bulletin*, 76(5):378, 1971.
39. E.W. Forgy. Cluster analysis of multivariate data: Efficiency vs. interpretability of classifications. *Biometrics*, 21:768–769, 1965.
40. Edward B Fowlkes and Colin L Mallows. A method for comparing two hierarchical clusterings. *Journal of the American statistical association*, 78(383):553–569, 1983.
41. Hichem Frigui and Raghu Krishnapuram. Clustering by competitive agglomeration. *Pattern Recognition*, 30(7):1109 – 1119, 1997.
42. Guojun Gan, Chaoqun Ma, and Jianhong Wu. *Data clustering: theory, algorithms, and applications*, volume 20 of *ASA-SIAM Series on Statistics and Applied Probability*. Siam, Philadelphia, 2007.
43. Jan Gorodkin. Comparing two k-category assignments by a k-category correlation coefficient. *Computational biology and chemistry*, 28(5):367–374, 2004.
44. M. Halkidi, M. Vazirgiannis, and Y. Batistakis. *Quality Scheme Assessment in the Clustering Process*, pages 265–276. Springer, Berlin, Heidelberg, 2000.
45. Maria Halkidi, Yannis Batistakis, and Michalis Vazirgiannis. On clustering validation techniques. *Journal of Intelligent Information Systems*, 17(2):107–145, 2001.
46. Maria Halkidi and Michalis Vazirgiannis. Clustering validity assessment: Finding the optimal partitioning of a data set. In *Proceedings of the 2001 IEEE International Conference on Data Mining*, ICDM '01, pages 187–194, Washington, DC, USA, 2001. IEEE Computer Society.
47. Kevin A Hallgren. Computing inter-rater reliability for observational data: an overview and tutorial. *Tutorials in quantitative methods for psychology*, 8(1):23, 2012.
48. Jiawei Han and Micheline Kamber. *Data Mining: Concepts and Techniques*. The Morgan Kaufmann Series in Data Management Systems.
49. H. Hruschka. Market definition and segmentation using fuzzy clustering methods. *International Journal of Research in Marketing*, 3(2):117 – 134, 1986.
50. Jih-Jeng Huang, Gwo-Hshiung Tzeng, and Chorng-Shyong Ong. Marketing segmentation using support vector clustering. *Expert Systems with Applications*, 32(2):313 – 317, 2007.
51. Zhexue Huang. Extensions to the k-means algorithm for clustering large data sets with categorical values. *Data Mining and Knowledge Discovery*, 2(3):283–304, 1998.
52. Zhexue Huang and Michael K. Ng. A fuzzy k-modes algorithm for clustering categorical data. *IEEE Transactions on Fuzzy Systems*, 7(4):446–452, 1999.
53. Mario Inostroza-Ponta, Regina Berretta, Alexandre Mendes, and Pablo Moscato. *An automatic graph layout procedure to visualize correlated data*, pages 179–188. Springer, 2006.
54. Mario Inostroza-Ponta, Alexandre Mendes, Regina Berretta, and Pablo Moscato. *An integrated QAP-based approach to visualize patterns of gene expression similarity*, pages 156–167. Springer, 2007.
55. Kyoung jae Kim and Hyunchul Ahn. A recommender system using {GA} k-means clustering in an online shopping market. *Expert Systems with Applications*, 34(2):1200 – 1209, 2008.
56. Anil K. Jain. Data clustering: 50 years beyond k-means. *Pattern Recognition Letters*, 31(8):651 – 666, 2010. Award winning papers from the 19th International Conference on Pattern Recognition (ICPR)19th International Conference in Pattern Recognition (ICPR).
57. Anil K. Jain, M. Narasimha Murty, and Patrick J. Flynn. Data clustering: A review. *ACM Comput. Surv.*, 31(3):264–323, 1999.
58. Giuseppe Jurman, Samantha Riccadonna, and Cesare Furlanello. A comparison of MCC and CEN error measures in multi-class prediction. *PLOS ONE*, 7(8):1–8, 08 2012.

59. Ah Keng Kau and Pei Shan Lim. Clustering of Chinese tourists to Singapore: an analysis of their motivations, values and satisfaction. *International Journal of Tourism Research*, 7(4-5):231–248, 2005.
60. Leonard Kaufman and Peter J. Rousseeuw. *Finding Groups in Data: An Introduction to Cluster Analysis*. Wiley Series in Probability and Mathematical Statistics. John Wiley & Sons, Inc., Hoboken, New Jersey, 1990.
61. Leonard Kaufman and Peter J. Rousseeuw. *Finding Groups in Data: An Introduction to Cluster Analysis*. Wiley Series in Probability and Mathematical Statistics. John Wiley & Sons, Inc., Hoboken, New Jersey, 2005.
62. Navneet Kaur and Craig M. Gelowitz. A tweet grouping methodology utilizing inter and intra cosine similarity. In *IEEE 28th Canadian Conference on Electrical and Computer Engineering, CCECE 2015, Halifax, NS, Canada, May 3-6, 2015*, pages 756–759. IEEE, 2015.
63. D. Kavyasrujana and B. Chakradhara Rao. *Hierarchical Clustering for Sentence Extraction Using Cosine Similarity Measure*, pages 185–191. Springer International Publishing, Cham, 2015.
64. Minho Kim and R.S. Ramakrishna. New indices for cluster validity assessment. *Pattern Recognition Letters*, 26(15):2353 – 2363, 2005.
65. Frank Klawonn, Rudolf Kruse, and Roland Winkler. Fuzzy clustering: More than just fuzzification. *Fuzzy Sets and Systems*, 281:272 – 279, 2015. Special Issue Celebrating the 50th Anniversary of Fuzzy Sets.
66. Mirella Kleijnen, Ko de Ruyter, and Martin Wetzels. Consumer adoption of wireless services: Discovering the rules, while playing the game. *Journal of Interactive Marketing*, 18(2):51 – 61, 2004.
67. Solomon Kullback. *Information theory and statistics*. Courier Corporation, 1997.
68. R.J. Kuo, Y.L. An, H.S. Wang, and W.J. Chung. Integration of self-organizing feature maps neural network and genetic k-means algorithm for market segmentation. *Expert Systems with Applications*, 30(2):313 – 324, 2006.
69. R.J. Kuo, L.M. Ho, and C.M. Hu. Integration of self-organizing feature map and k-means algorithm for market segmentation. *Computers & Operations Research*, 29(11):1475 – 1493, 2002.
70. M. Sh. Levin. Combinatorial clustering: Literature review, methods, examples. *Journal of Communications Technology and Electronics*, 60(12):1403–1428, 2015.
71. Q. Lin and Y. Wan. Mobile customer clustering based on call detail records for marketing campaigns. In *2009 International Conference on Management and Service Science*, pages 1–4, Sept 2009.
72. Bing Liu. *Web Data Mining: Exploring Hyperlinks, Contents, and Usage Data*. Data-Centric Systems and Applications. Springer-Verlag, 2nd edition, 2008.
73. Ying Liu, Hong Li, Geng Peng, Benfu Lv, and Chong Zhang. Online purchaser segmentation and promotion strategy selection: evidence from Chinese e-commerce market. *Annals of Operations Research*, 233(1):263–279, 2013.
74. Ying Liu, Sudha Ram, Robert F. Lusch, and Michael Brusco. Multicriterion market segmentation: A new model, implementation, and evaluation. *Marketing Science*, 29(5):880–894, 2010.
75. Benjamin Lucas, Ahmed Shamsul Arefin, Natalie Jane de Vries, Regina Berretta, Jamie Carlson, and Pablo Moscato. Engagement in motion: Exploring short term dynamics in page-level social media metrics. In *2014 IEEE Fourth International Conference on Big Data and Cloud Computing, BDCloud 2014, Sydney, Australia, December 3-5, 2014*, pages 334–341, 2014.
76. James MacQueen et al. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, pages 281–297. Oakland, CA, USA., 1967.
77. Katariina Mäenpää. Clustering the consumers on the basis of their perceptions of the internet banking services. *Internet Research*, 16(3):304–322, 2006.

78. Pritha Mahata, Wagner Costa, Carlos Cotta, and Pablo Moscato. Hierarchical clustering, languages and cancer. In Franz Rothlauf, Jürgen Branke, Stefano Cagnoni, Ernesto Costa, Carlos Cotta, Rolf Drechsler, Evelyne Lutton, Penousal Machado, Jason H. Moore, Juan Romero, George D. Smith, Giovanni Squillero, and Hideyuki Takagi, editors, *Applications of Evolutionary Computing, EvoWorkshops 2006: EvoBIO, EvoCOMNET, EvoHOT, EvoIASP, EvoINTERACTION, EvoMUSART, and EvoSTOC, Budapest, Hungary, April 10-12, 2006, Proceedings*, volume 3907 of *Lecture Notes in Computer Science*, pages 67–78. Springer, 2006.
79. Brian W Matthews. Comparison of the predicted and observed secondary structure of t4 phage lysozyme. *Biochimica et Biophysica Acta (BBA)-Protein Structure*, 405(2):442–451, 1975.
80. Marina Meilă. *Comparing Clusterings by the Variation of Information*, pages 173–187. Springer, Berlin, Heidelberg, 2003.
81. Volodymyr Melnykov and Ranjan Maitra. Finite mixture models and model-based clustering. *Statist. Surv.*, 4:80–116, 2010.
82. Henriette Müller and Ulrich Hamm. Stability of market segmentation with cluster analysis – a methodological approach. *Food Quality and Preference*, 34:70 – 78, 2014.
83. Leila M Naeni, Hugh Craig, Regina Berretta, and Pablo Moscato. A novel clustering methodology based on modularity optimisation for detecting authorship affinities in Shakespearean era plays. *PLoS One*, 11(8):e0157988, 2016.
84. Morteza Namvar and Mohammad R. Gholamian. A two phase clustering method for intelligent customer segmentation. In *Proceedings of the International Conference on Intelligent Systems, Modelling and Simulation*, pages 61–68, Liverpool, UK, 2010. IEEE.
85. S.R. Nanda, B. Mahanty, and M.K. Tiwari. Clustering Indian stock market data for portfolio management. *Expert Systems with Applications*, 37(12):8793 – 8798, 2010.
86. R. T. Ng and Jiawei Han. CLARANS: a method for clustering objects for spatial data mining. *IEEE Transactions on Knowledge and Data Engineering*, 14(5):1003–1016, Sep 2002.
87. Taher Niknam and Babak Amiri. An efficient hybrid approach based on PSO, ACO and k-means for cluster analysis. *Applied Soft Computing*, 10(1):183 – 197, 2010.
88. Łukasz P. Olech and Mariusz Paradowski. *Hierarchical Gaussian Mixture Model with Objects Attached to Terminal and Non-terminal Dendrogram Nodes*, pages 191–201. Springer International Publishing, Cham, 2016.
89. Lucie K. Ozanne and Paul W. Ballantine. Sharing as a form of anti-consumption? An examination of toy library users. *Journal of Consumer Behaviour*, 9(6):485–498, 2010.
90. Muammer Ozer. User segmentation of online music services using fuzzy clustering. *Omega*, 29(2):193 – 206, 2001.
91. Hae-Sang Park and Chi-Hyuck Jun. A simple and fast algorithm for k-medoids clustering. *Expert Systems with Applications*, 36(2, Part 2):3336 – 3341, 2009.
92. Girish Punj and David W. Stewart. Cluster analysis in marketing research: Review and suggestions for application. *Journal of Marketing Research*, 20(2):pp. 134–148, 1983.
93. M. Rezaei and P. Fränti. Set matching measures for external cluster validity. *IEEE Transactions on Knowledge and Data Engineering*, 28(8):2173–2186, Aug 2016.
94. Romeo Rizzi, Pritha Mahata, Luke Mathieson, and Pablo Moscato. Hierarchical clustering using the arithmetic-harmonic cut: Complexity and experiments. *PLOS ONE*, 5(12):1–8, 12 2010.
95. Peter J Rousseeuw. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics*, 20:53–65, 1987.
96. Enrique H. Ruspini. A new approach to clustering. *Information and Control*, 15(1):22 – 32, 1969.
97. Jörg Sander, Martin Ester, Hans-Peter Kriegel, and Xiaowei Xu. Density-based clustering in spatial databases: The algorithm GDBSCAN and its applications. *Data Mining and Knowledge Discovery*, 2(2):169–194, 1998.
98. William A Scott. Reliability of content analysis: The case of nominal scale coding. *Public opinion quarterly*, pages 321–325, 1955.

99. Claude Elwood Shannon. A mathematical theory of communication. *ACM SIGMOBILE Mobile Computing and Communications Review*, 5(1):3–55, 2001.
100. Padhraic Smyth. *Handbook of Data Mining and Knowledge Discovery*, chapter 16.5 Clustering, pages 386–388. Oxford University Press, 2002.
101. Michał Spytkowski, Łukasz P. Olech, and Halina Kwaśnicka. *Hierarchy of Groups Evaluation Using Different F-Score Variants*, pages 654–664. Springer, Berlin, Heidelberg, 2016.
102. Douglas. Steinley. K-means clustering: A half-century synthesis. *British Journal of Mathematical and Statistical Psychology*, 59(1):1–34, 2006.
103. Douglas Steinley and Michael J. Brusco. A new variable weighting and selection procedure for k-means cluster analysis. *Multivariate Behavioral Research*, 43(1):77–108, 2008. PMID: 26788973.
104. Michio Sugeno and Takahiro Yasukawa. A fuzzy-logic-based approach to qualitative modeling. *IEEE Transactions on fuzzy systems*, 1(1):7–31, 1993.
105. Michael Nche Tuma, Reinhold Decker, and Sören Scholz. A survey of the challenges and pitfalls of cluster analysis application in market segmentation. *International Journal of Market Research*, 53(3):391–414, 2011.
106. VI Vagizova, KM Lurie, and Ihor Bogdanovych Ivasiv. Clustering of Russian banks: business models of interaction of the banking sector and the real economy. *Problems and perspectives in management*, 12(1):83–93, 2014.
107. C. J. van Rijsbergen. *Information Retrieval*. Butterworth, 1979.
108. Silke Wagner and Dorothea Wagner. *Comparing clusterings: an overview*. Universität Karlsruhe, Fakultät für Informatik Karlsruhe, 2007.
109. Anongnart Srivihok Waminee Niyagas and Sukumal Kitisin. Clustering e-banking customer using data mining and marketing segmentation. *Transactions on Computer and Information Technology (ECTI-CIT)*, 2(1), 2006.
110. Jiaqi Wang, Xindong Wu, and Chengqi Zhang. Support vector machines based on k-means clustering for real-time business intelligence systems. *International Journal of Business Intelligence and Data Mining*, 1(1):54–64, 2005.
111. Michel Wedel and Jan-Benedict E.M. Steenkamp. A fuzzy clusterwise regression approach to benefit segmentation. *International Journal of Research in Marketing*, 6(4):241 – 258, 1989.
112. William G. Wee and K. S. Fu. A formulation of fuzzy automata and its application as a model of learning systems. *IEEE Transactions on Systems Science and Cybernetics*, 5(3):215 – 223, 1969.
113. Ian H. Witten and Eibe Frank. *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann Publishers Elsevier, 2nd edition, 2005.
114. Jing Wu and Zheng Lin. Research on customer segmentation model by clustering. In *Proceedings of the 7th International Conference on Electronic Commerce*, ICEC ’05, pages 316–318, New York, NY, USA, 2005. ACM.
115. Zhengrong Xiang and Md Zahidul Islam. Hartigan’s method for k-modes clustering and its advantages. In *Proceedings of the Twelfth Australasian Data Mining Conference (AusDM 2014), Brisbane, Australia*, pages 25–30. Australian Computer Society Inc., 2014.
116. Rui Xu and Donald C. Wunsch II. *Clustering*. IEEE Press Series on Computational Intelligence. John Wiley & Sons, Inc., Hoboken, New Jersey, 2009.

Chapter 4
An Introduction to Proximity Graphs

Luke Mathieson and Pablo Moscato

Abstract Proximity graphs are one of the combinatorial data-miner's frontline tools. They allow expression of complex proximity relationships and are the basis of many other algorithms. Here we introduce the concept of proximity graphs, present basic definitions and discuss some of the most common types of proximity graphs.

Keywords Delaunay triangulation · Gabriel graph · Influence graph · Minimum spanning tree · Nearest neighbour graph · Relative neighbourhood graph · Steiner tree · Unit disk graph · Urquhart graph · Voronoi diagram

4.1 Introduction

When faced with a data set from which we wish to extract interesting relationships, one of the first natural, informal questions we ask is "how far are each of these data points from each other?" Formalizing and answering this question involves the development of notions of distance and space—we must first know how to measure distance before we can talk about "near" and "far". Once this is done and some suitable metric has been developed, we can begin to explore the relationships between point in the data set in terms of their *proximity*. One of the most natural, flexible and powerful mathematical tools for expressing an exploring relationships is the *graph*. Thus when we wish to express the totality of proximity relationships in a data set, *proximity graphs* are, unsurprisingly, one of the most expressive

L. Mathieson (✉)
School of Electrical Engineering and Computing, The University of Newcastle, Callaghan, NSW, Australia

School of Software, University of Technology Sydney, Ultimo, NSW, Australia
e-mail: luke.mathieson@uts.edu.au

P. Moscato
School of Electrical Engineering and Computing, The University of Newcastle, Callaghan, NSW, Australia
e-mail: Pablo.Moscato@newcastle.edu.au

and effective tools available. This has become even more true with the advent of computer hardware that can handle extremely large data sets. Not only are proximity graphs interesting as data mining tools in their own right, they form the basis of many clustering, and visualization algorithms. In this chapter we introduce the ideas central to proximity graphs, and explore some of the more common types.

4.1.1 Basic Definitions and Notations

In the following we assume we have a set of points (or *vertices*) $V = \{v_1, \ldots, v_n\}$ in a space over which is defined a distance measure $d : V \times V \to \mathbb{R}$. The choice of $\mathbb{R}$ as the range of the function d is somewhat arbitrary, any set equipped with a notion of ordering is typically sufficient for the definitions of most proximity graphs; however, we can typically find an isomorphism between the range and some subset of $\mathbb{R}$.

As we are considering an initial input of a set of points in a space, $d(u, v)$ is defined for every pair $u, v \in V$; however, we note that for some of the following definitions this is not a necessary condition, or more precisely, it is possible to relax d to allow a null value of some form. Numerically this can typically be represented by ∞, or even a sufficiently large number is often suitable. Similarly, it is also not strictly necessary for many of the following definitions that the points are in fact in a space, only that d is well defined; however, the geometric interpretations of the definitions quickly become unintuitive in these cases. Indeed, if we take the most general case where we are given a (not necessarily complete) graph with edge *lengths*, the distances may not even obey the triangle inequality, that is, the direct distance between two adjacent vertices may not be the shortest distance between those two vertices. In this case, it is helpful to distinguish the distance $d(u, v)$ between u and v and the length $l(u, v)$ of edge (u, v).

A graph $G = (V, E)$ consists of a set V of vertices and a set $E \subseteq V \times V$ of edges. In our context, the set of vertices will be the point set given as input. For the following, unless otherwise stated, we will consider as intermediates and outputs only simple, undirected, edge-weighted graphs. A simple graph is a graph with no edges of the form (v, v), i.e., no self-loops, and no multi-edges, i.e., there is at most one edge between any pair of vertices. An undirected graph is a graph where the edge (u, v) is identical to the edge (v, u), i.e., the edges do not have a direction. An edge-weighted graph is a graph where each edge has an associated weight (implicitly numeric). We may simulate unweighted graphs in this context by choosing all edge weights to be equal (say, all equal to 1, for example).

For convenience we define the complete, weighted graph on point set V to be the graph where V forms the set of vertices and for every pair $u, v \in V$ of distinct vertices, there is an edge $(u, v) \in E$ where the weight of (u, v) is $d(u, v)$. We will denote this by $K[V]$. Every proximity graph on V is a subgraph of $K[V]$.

For comparability, the definitions that follow will be accompanied by examples using the point set in Fig. 4.1. The coordinates for the points are given in the Appendix.

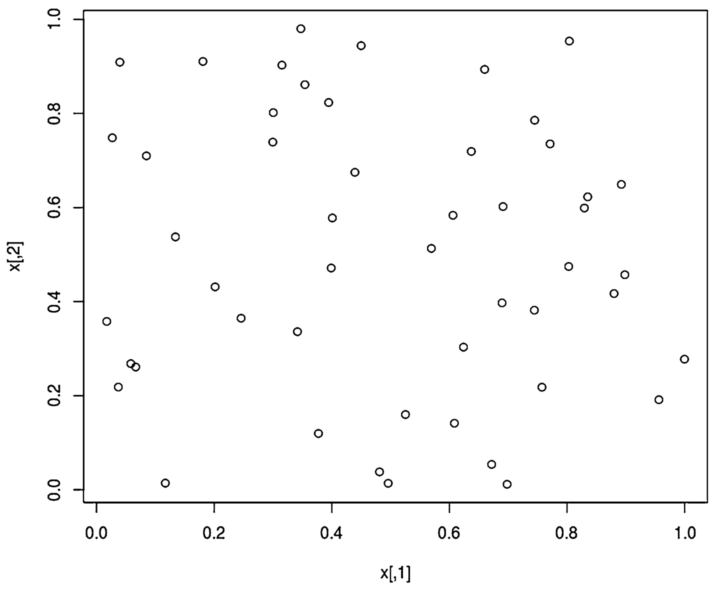


Fig. 4.1 A set of points in the 2D plane using the Euclidean distance metric. This point set will be used as a base for several of the examples below

4.2 Minimum Spanning Trees

A *path* between vertices u_1 and u_k in a graph is a set of edges $\{(u_1, u_2), (u_2, u_3), \ldots (u_{k-1}, u_k)\}$, where $u_i \neq u_j$ for each $i, j \in \{1, \ldots, k\}$.

A graph is *connected* if there is a path between every pair of vertices in the graph.

A *tree* is a connected graph with no cycles. Given a graph, a *spanning* tree is a tree that connects every vertex in the graph. A *minimum* spanning tree (MST) of a graph is a spanning tree where the sum of weights on the edges is the minimum over all possible spanning trees. MSTs are a fundamental notion in graph theory and are widely applicable in many practical areas.

To see how an MST is a prototypical proximity graph, we introduce Prim's algorithm for computing MSTs:

Thus the selection of which edges to add to the resultant MST is based on the next nearest vertex to anything currently in the tree (line 4).

Algorithm 1: Prim's algorithm

```
Input  : A simple, connected, undirected, edge-weighted graph $G = (V, E)$.
Output: A MST of $G$.
1 Arbitrarily select a vertex $v$ from $V$.
2 Initialise a tree $T = (W, F)$ with $W := \{v\}$ and $F := \emptyset$.
3 while $W \neq V$ do
4     Pick the smallest weight edge $(u, v)$ from $E$ with $u \in W$ and $v \notin W$ (break ties
        arbitrarily)
5     Add $v$ to $W$ and $(u, v)$ to $F$
6 end while
```

In general, MSTs are not unique. For example, if the weights of all edges in the input graph are identical, *any* spanning tree is an MST. However if the weight of each edge is unique, the graph has a single MST. In the case of the example point set, the MST, shown in Fig. 4.2, is unique as a result of the random distribution of points.

There are many algorithms for computing Minimum Spanning Trees: Borůvka's algorithm [5][1]; Prim's algorithm[2]; Kruskal's algorithm [29]; the reverse-delete algorithm [29]; Karger, Klein and Tarjan's randomized algorithm [27] and Chazelle's soft heap algorithm [8].

These are all polynomial time algorithms and apart from Karger, Klein and Tarjan's algorithm, these are deterministic. We note that the randomness in Karger, Klein and Tarjan's algorithm does not affect solution quality, but serves to reduce the expected runtime of the algorithm to linear time, with a worst case identical to Borůvka's algorithm. These algorithms also deal with the most general, non-geometric case, where the input is simply a set of vertices with arbitrary distances defined between them—that is, the vertices are not necessarily embedded in any kind of geometrically interpretable space.

In this particular setting we expect the input graph to be $K[V]$, a dense graph (indeed, the densest!) with $\frac{n(n-1)}{2}$ edges, where $n = |V|$. This results in a worst case performance (and for most a best case as well) of at least $O(n^2 \log n)$ for the above algorithms, unless particular care is taken with the data structures used. Using the Fibonacci heap data structure, Fredman and Tarjan developed a deterministic linear time algorithm [20], employing Prim's algorithm as a subroutine. This algorithm has a worst case runtime of $O(n^2)$ in our context, the best asymptotic running time possible for the general case at the cost of a complicated implementation.

If the space in which the points are embedded is planar Euclidean (i.e., a plane with an Euclidean distance metric), an MST can be found in $O(n \log n)$ time [38] by

[1]Reinvented three further times under different names [9, 17], most notably as Sollin's algorithm [39].

[2]Also invented repeatedly [14, 24, 35], first by Vojtěch Jarník [24]. The name Prim's algorithm is however almost universally used.

restricting the edges in the input graph to those of the Delaunay Triangulation (see Sect. 4.6). In higher dimensions, the Euclidean MST problem can still be solved (asymptotically) faster than the general, non-geometric case. For any number of dimensions greater than 2, there is a $o(n^2)$ algorithm [48], for three dimensions in particular this can be improved to $O((n \log n)^{\frac{3}{2}})$ [2].

4.3 Relative Neighbourhood Graphs

Originally defined for a set of points on the Euclidean Plane [43], the Relative Neighbourhood Graph (RNG) connects two points if there does not exist a third point that is closer to both than they are to each other. In terms of the distance function, the edge (u, v) is in the RNG if there is no third point r such that $d(r, u) < d(u, v)$ and $d(r, v) < d(u, v)$. Figure 4.3 gives the RNG for the example point set.

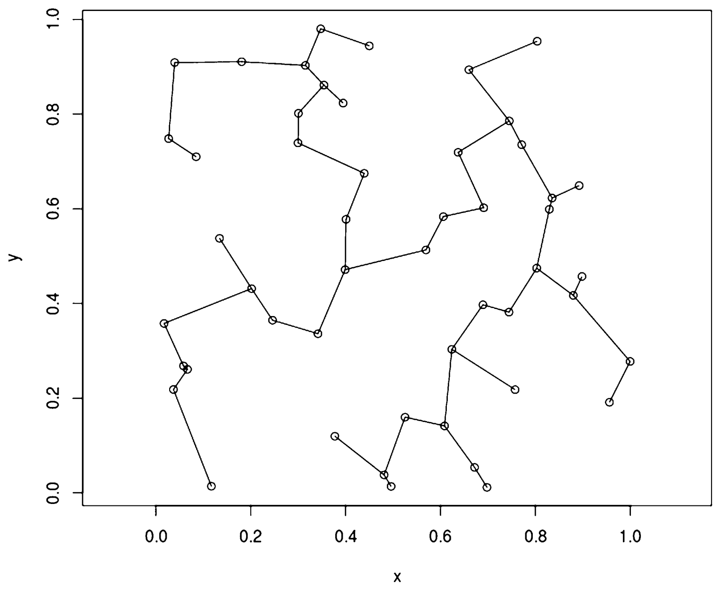


Fig. 4.2 An MST for the example point set

On the Euclidean Plane, the RNG can be computed in $O(n \log n)$ time [41]. Given a Delaunay Triangulation (Sect. 4.6), this can be reduced to $O(n)$ time [32].

As the definition depends only upon the distance between points, RNGs can easily be generalized to non-Euclidean spaces (or indeed non-metric). There exists a variety of algorithms for spaces of differing dimension and distance metric [25, 34], the time to compute the RNG in the most general setting however is $O(n^3)$.

When working with suitable spaces we may use an alternate definition which turns out to be a specific case of a more general proximity graph model. Arising from the work on geometric graphs, the RNG can be defined using lune-based β-Skeletons (see Sect. 4.5 and Fig. 4.6). In this model, an edge is added between points u and v if there are no points of V in the intersection of the two circles centred at u and v with radius $d(u, v)$ (more strictly, if the intersection of the two closed 2-balls so constructed is empty of points from V). This can easily be generalized to the intersection of closed n-balls in n-dimensional space.

4.4 Gabriel Graphs

The Gabriel Graph of a set of points in a space is defined similarly to the Relative Neighbourhood Graph; however, instead of using a lens arising from the intersection of the circles centred at two points, the potential edge is a diameter of a circle. If the circle contains no other points, then the edge is added. More precisely, given two points u and v, the edge (u, v) is in the Gabriel Graph if for all other points w, $d(u, v)^2 \leq d(u, w)^2 + d(v, w)^2$. Figure 4.4 gives the Gabriel Graph of the example point set using the Euclidean distance.

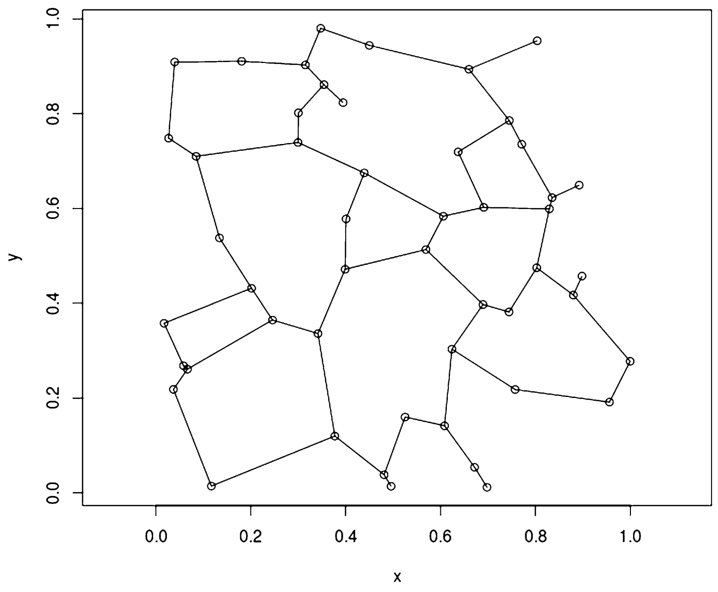


Fig. 4.3 The Relative Neighbourhood Graph for the example point set

Again, the Gabriel Graph was originally defined for the Euclidean Plane [21]; however, the idea can easily be generalized to other spaces where the distance between each pair of points is defined.

The Gabriel Graph, like the Relative Neighbourhood Graph is a type of β-skeleton (see Sect. 4.5 and Fig. 4.5b), and in fact contains the RNG (and hence the MST) as a subgraph. Also like the RNG, it is a subgraph of the Delaunay Triangulation; hence, in the Euclidean Plane, the Gabriel Graph can be found in $O(n \log n)$ time (or $O(n)$ if the Delaunay Triangulation is given), in much the same manner as the RNG, except with the modified condition.

4.5 β-Skeletons

The β-Skeleton of a set of points is a generalization of some of the geometric notions that lead to RNGs and Gabriel Graphs. Broadly speaking, an edge (u, v) is in the β-skeleton if no points lie in a region defined by circles related to the two points and the value of the parameter β. As special cases, we immediately obtain the RNG and Gabriel Graph.

There are two definitions of β-Skeletons, the *circle*-based definition and the *lune*-based definition.

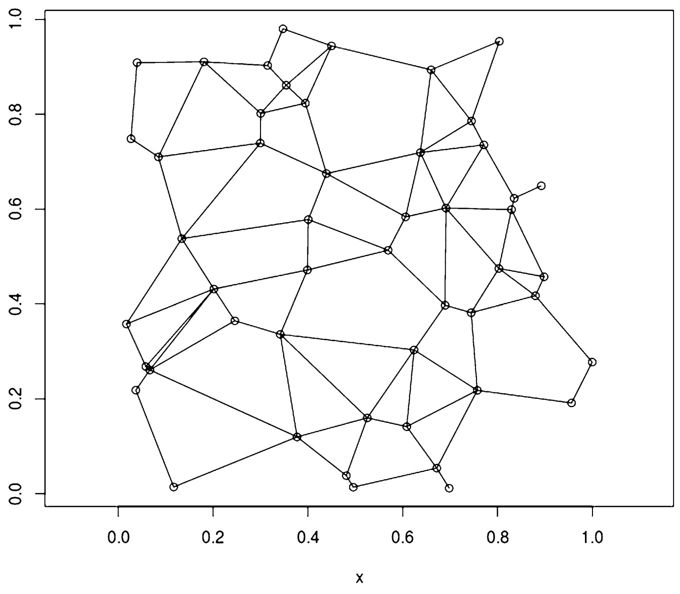


Fig. 4.4 The Gabriel Graph of the example point set

Circle-Based Definition The parameter β is a positive real number that gives the following angle θ:

$$\theta = \begin{cases} \sin^{-1}\frac{1}{\beta}, & \text{if } \beta \geq 1 \\ \pi - \sin^{-1}\beta, & \text{if } \beta \leq 1 \end{cases}$$

For any two points u and v, let P_{uv} be the set of points p such that $\angle upv \geq \theta$, the edge (u, v) is in the β-Skeleton if P_{uv} is empty. More intuitively what this defines is a pair of circles with diameter $\beta \cdot d(u, v)$ if $\beta \geq 1$ or $\frac{d(u,v)}{\beta}$ if $\beta < 1$, where u and v lie on the circumference (it is easy to verify that in the Euclidean Plane at least, there are only two such circles). If $\beta \leq 1$, then the edge is added if there are no points in the intersection of the circles, if $\beta \geq 1$, the edge is added if there are no points in the union of the circles. The case $\beta = 1$ is degenerate, and the two circles are identical. See Fig. 4.5 for an illustration of this definition.

Lune-Based Definition Alternatively, instead of the line segment uv forming a chord, for $\beta \geq 1$ the circles may be placed such that uv lies on a diameter of both circles, with u and v on the boundary of the intersection. That is, the lune is formed from the intersection of two circles of radius $\frac{\beta \cdot d(u,v)}{2}$ centred at $(1 - \frac{\beta}{2})u + \frac{\beta}{2}v$ and $\frac{\beta}{2}u + (1 - \frac{\beta}{2})v$. When $\beta \leq 1$, the definition is identical to the circle-based definition (Fig. 4.6).

The two definitions coincide when $\beta \leq 1$ and for larger values of β, the circle-based definition gives a subgraph of the lune-based definition. The Gabriel Graph is precisely the case where $\beta = 1$. The RNG is the case $\beta = 2$ in the lune-based definition. In general for two values β_1 and β_2 with $\beta_1 \leq \beta_2$, the β_1-Skeleton is a supergraph of the β_2-Skeleton, leading to the earlier observation that the RNG is a subgraph of the Gabriel Graph. We note however that the Delaunay Triangulation is *not* in general a β-Skeleton.

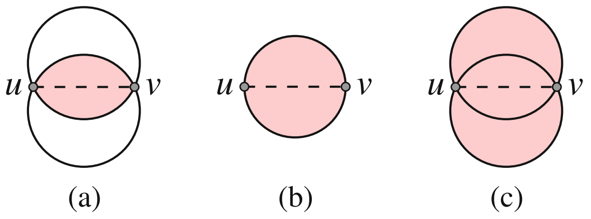


Fig. 4.5 The three basic cases for determining the edges of the β-skeleton in the circle-based definition. (**a**) When $\beta \leq 1$, the area of intersection of the two circles must be free of any other points for the edge (u, v) to exist. (**b**) $\beta = 1$ gives the degenerate cases where the intersection is precisely the union, i.e., the two circles are identical. This corresponds to the Gabriel Graph (Sect. 4.4). (**c**) When $\beta \geq 1$, the union of the two circles must be empty

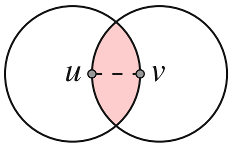


Fig. 4.6 The alternate, lune-based definition when $\beta \geq 1$. Again when $\beta = 1$, the circles coincide. Note that as β approaches infinity, the region of intersection approaches the infinite area demarked by the tangents to the circles at u and v. As β approaches zero, the lune approximates the line segment from between u and v. When $\beta = 2$, this definition gives the RNG (Sect. 4.3)

The naïve algorithm (testing all triples) gives an $O(n^3)$ time bound, and outside the Euclidean planar case, may be the best algorithm possible (indeed, moving to more general metrics requires some thought in simply defining what is meant by a circle).

For suitable cases where $\beta \geq 1$ we know that the β-Skeleton is a subgraph of the Gabriel Graph, and hence a subgraph of the Delaunay Triangulation, giving an $O(n \log n)$ algorithm for constructing the β-Skeleton (again $O(n)$ where the triangulation is given).

In the planar Euclidean case, we may also compute the $\beta < 1$ case in $O(n^2)$ time [23]. There is no better upper bound possible, as cases which produce the complete graph, and hence require $\Omega(n^2)$ comparisons, exist.

The β-Skeleton was proposed by Kirkpatrick and Radke [28] based on the idea of α-Shapes [16]. Generalizations using shapes other than circles also exist [7, 45].

4.6 Delaunay Triangulations

A Delaunay Triangulation[3] is built from a set of points in a similar manner to the Gabriel Graph; however, three points are used instead of two.

A triangulation of a set of points is any set of edges between the points such that for any set of edges that form a polygon, if there are more than 3 points, the polygon has a chord. Note that the definition is implicitly recursive, so the resulting graph is a collection of incident triangles (Fig. 4.7).

A Delaunay Triangulation of a set of points (again originally defined on the Euclidean Plane) is a triangulation where for every triangle, the circumcircle contains no other points. There are some sets of points for which the Delaunay Triangulation is not unique and when generalizing to higher dimensions or non-Euclidean metrics, a Delaunay Triangulation may not exist.

[3]Note also that the Voronoi Diagram of a set of points in general position is the dual graph of the Delaunay Triangulation.

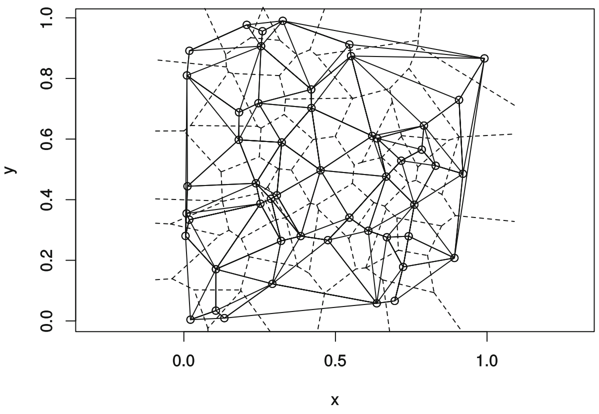


Fig. 4.7 The Delaunay Triangulation of the example point set. The dashed lines indicate the complementary Voronoi Diagram

The Delaunay Triangulation, as mentioned, contains the Gabriel Graph, the RNG and hence the MST as subgraphs when using suitable metrics, and in the planar Euclidean case in particular. In the Euclidean Plane each point has on average six surrounding triangles. Combining these facts gives the simple $O(n)$-time algorithms for finding RNGs and Gabriel Graphs when given a Delaunay Triangulation.

There are a number of algorithms for computing a Delaunay Triangulation, the simplest of which involves *edge flipping*. If two triangles that share an edge do not satisfy the Delaunay condition (i.e., all four points are in the circumcircle of at least one of the triangles), then the common edge can be "flipped" by swapping it for the other potential shared edge (i.e., if the triangles are on points w, x, y and x, y, z, respectively, then the common edge (x, y) can be flipped to the new common edge (w, z)). The resulting triangle pair will now satisfy the Delaunay condition. This prompts a simple algorithm: construct any triangulation, then proceed to flip edges until no triangle violates the Delaunay condition. This leads to an $O(n^2)$ algorithm, and can be generalized to higher dimensions [13]; however, the higher dimensional version may not converge—while the flip operation remains valid, it may disconnect the graph.

Alternatively, we may begin with a single triangle and incrementally add points to the triangulation, repairing the graph at each step via edge flipping. If done without care, this leads to an $O(n^2)$ algorithm; however, using a random insertion order and suitable data structures to quickly find triangles that need repairing in $O(\log n)$ time, the number of flips per insertion can be reduced to a constant on average. Taken over all vertices this gives a $O(n \log n)$ time algorithm [12]. This algorithm can be safely generalized to higher dimensions [15]; however, the running time may be exponential in the dimension.

The Bowyer–Watson algorithm uses a similar incremental approach, adding a point to the triangulation at each step; however, instead of flipping edges, it removes all conflicting triangles and retriangulates the polygonal hole generated.

The algorithm was discovered independently by Bowyer [6] and Watson [47].[4] The algorithm can triangulate a set of n points in $O(n \log n)$ time in any dimensionality; however, special cases exist which lead to $O(n^2)$ running time [36].

In two dimensions, the most efficient [40] algorithm employs a divide-and-conquer approach, where the set of points is recursively split in two by a line, the Delaunay Triangulation is computed for each partite set, and the two sets are merged along the separating line. The merging of the sets can be performed in linear time, giving a worst case running time of $O(n \log n)$. A divide and conquer approach was first proposed by Shamos and Hoey [38] and subsequently improved upon [22, 30, 31]. Divide and conquer approaches can also be employed in higher dimensions [10] with good empirical performance, but no theoretical reduction in complexity.

4.6.1 Urquhart Graphs

The Gabriel Graph of a set of points is obtained from the Delaunay Triangulation by removing the longest edge from each triangle in the graph. Originally suggested [44] as a possible way of quickly computing the RNG, it was later demonstrated [42] that the resultant graph was not always the RNG of the set of points, it however remains an efficient approximation although somewhat superseded by later results [41]. As might be expected by this point, the Urquhart Graph contains the Minimum Spanning Tree of the point set.

4.7 Sphere of Influence Graphs

The Sphere of Influence Graph retains aspects of the β-Skeleton approach, but essentially inverts the edge existence condition. Given a set of points P, for each point v we find the nearest neighbour n_v, and draw a circle centred at v with radius $d(v, n_v)$. If the circles for any two points u and v intersect, then the edge (u, v) is added. Equivalently, if the distance between u and v is less than $d(v, n_v) + d(u, n_u)$, the edge (u, v) is added.

Naturally, this graph includes the Nearest Neighbour Graph (Sect. 4.10) as a subgraph, but is not guaranteed to include the MST as a subgraph, and in fact, may not even be connected (Fig. 4.8).

[4]It is interesting to note that not only was it discovered concurrently, but both articles were published in the same issue of the Computer Journal.

4.8 Shortest Path Trees

While the Minimum Spanning Tree gives a minimally connected subgraph[5] of total minimum weight, if we are interested in distances in particular, say for modelling travel times or resource allocation, we may prefer instead to minimize path lengths. Given a nominated starting vertex, we can build a tree which captures the Shortest Path from that vertex to every other vertex.

We can efficiently construct a Shortest Path Tree in the obvious manner; given the distances from our nominated starting vertex v to every other vertex, which can be computed using any suitable algorithm,[6] then for every other vertex u select a parent p for u out of $N(u)$ such that $d(v, p) + l(p, u) = d(v, u)$ (a tie-break rule may be needed, such as selecting the parent with the smallest number of edges to v along its Shortest Path).

Assuming the initial graph is connected, the Shortest-path tree is guaranteed to exist, but may not be unique, much like MSTs.

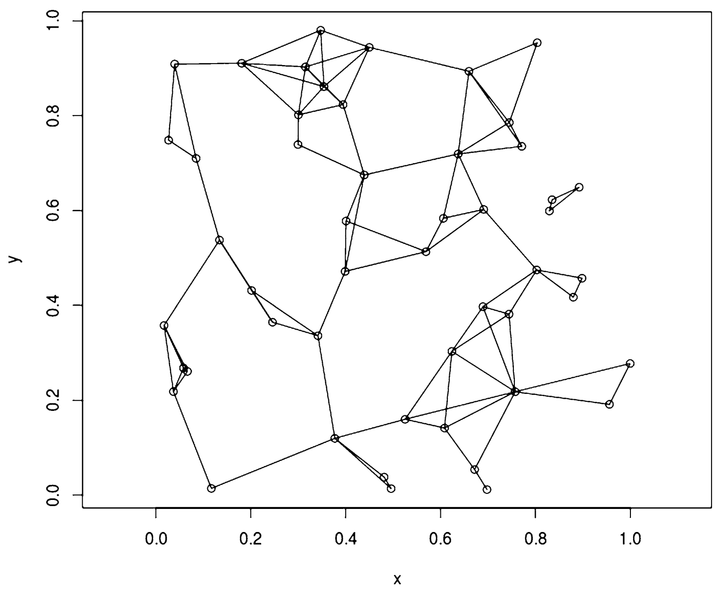


Fig. 4.8 Sphere of Influence Graph for the example point set

[5]In a case of a set of points, of the implicit complete graph.

[6]Some general algorithms are Dijkstra's Algorithm [14], the Bellman–Ford–Moore Algorithm [3, 19, 33], the Floyd–Roy–Warshall Algorithm [18, 37, 46] and Johnson's Algorithm [26].

4.9 Minimum Steiner Trees

A Steiner Tree of a set of points P is a Minimum Spanning Tree of an augmented set of points $P \cup S$ such that the points in S (the Steiner points) are added to reduce the total weight of the tree. In the most general setting, the problem may be viewed in purely graph theoretic terms as follows: given an edge-weighted graph G with a nominated set of required vertices P, a Steiner Tree is a tree in G spans P, the additional vertices of G that are used in the tree constitute S. In the Euclidean Plane (and other similar cases), our input graph is then “simply” the implicit infinite complete graph consisting in the set of point in the Euclidean plane.

In the Euclidean Plane, the Steiner points are Fermat points [1], which leads to the observation that at most $n - 2$ Steiner points are needed; however, for more general cases, no such bound necessarily holds.

Unlike the other proximity graphs mentioned so far, the *minimum weight* Steiner Tree is NP-hard to compute, even on the Euclidean Plane, requiring different algorithmic approaches to be used to solve the problem.

4.10 Nearest Neighbour Graphs

The Nearest Neighbour Graph (NNG) of a set of points is the smallest collection of edges such that each point is connected to all its nearest points.[7] Although the NNG is often presented as an undirected graph, the property of being a nearest neighbour is not symmetric, and thus the NNG is more strictly a directed graph.

The NNG is a subgraph of the Gabriel Graph and hence the Delaunay Triangulation. If the number of nearest neighbours chosen is restricted to one, or the vertex set is in general position,[8] the NNG is a subgraph of the MST. The full NNG may contain cycles however.

As would be expected, the NNG is not typically connected, but nonetheless finds many uses in a number of optimization fields.

The NNG naturally generalizes to the k-Nearest Neighbour Graph (k-NNG), where each vertex is connected to a set of nearest points which are at a distance less than or equal to the kth nearest distance. Thus the NNG of a set of points is precisely the 1-NNG. Figure 4.9 shows the 1-NNG, 3-NNG and 6-NNG for the example point set. For the 1-NN it is easy to verify that in this case it is a subset of the MST.

[7]Given a set of points P, a point q is a nearest point of point p if $d(p, q) = \min_{r \in P}\{d(p, r)\}$.

[8]That is, no hyperplane contains more than d points where d is the dimension of the space. In particular, in two dimensions this means that no three points are colinear.

4.11 Unit Disk Graphs

The Unit Disk Graph (UDG) [11] is a simple geometric intersection graph, similar to a simplified Sphere of Influence Graph. In the UDG, two points u and v are connected if $d(u, v) \leq c$, for some chosen constant c.[9] Equivalently this may be considered as the intersection graph of the set of radius c circles centred at the given points (Fig. 4.10).

This is one of the simplest proximity graphs, but still finds natural applications in computer networking, but more interestingly as models of random graphs associated with percolation phenomena. The UDG of a set of points can be computed in linear time [4], for any space of a fixed dimension, with suitable generalization to Spheres.

A Random Geometric Graph (RGG) is a randomly generated UDG—that is, the points are placed randomly in the space and the edges are added according to the UDG construction.

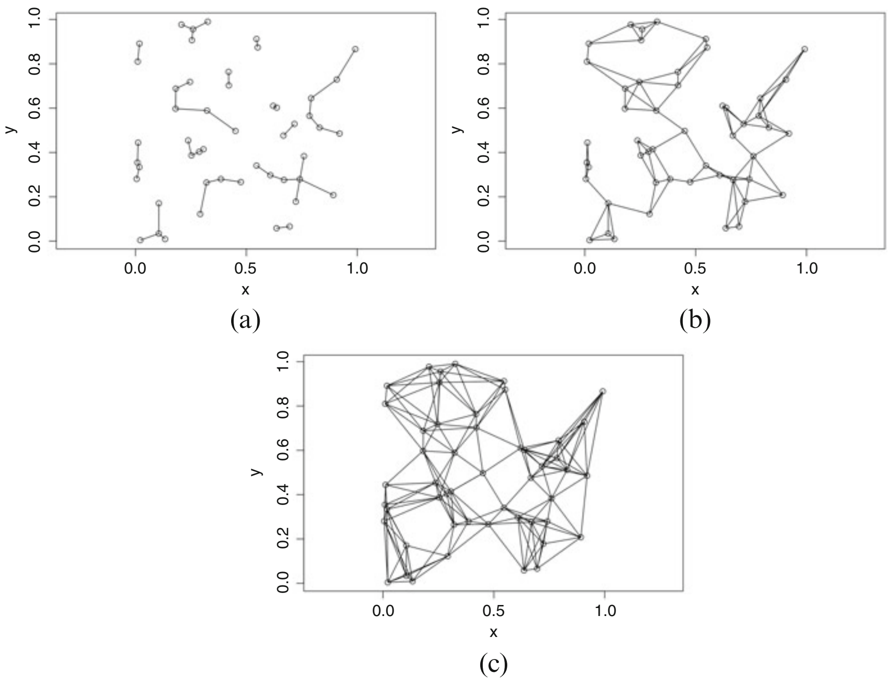


Fig. 4.9 The k-Nearest Neighbour Graphs for the example point set with $k \in \{1, 3, 6\}$. (**a**) $k = 1$. (**b**) $k = 3$. (**c**) $k = 6$

[9]c being the length of the "unit" in the graph's title.

4.12 An Example Application to Recommender Systems

Clearly the range of proximity graphs spans a wide array of concepts and provides a significant degree of flexibility for extracting interesting properties and information from a data set (*q.v.* Chap. 3 for an introduction to more advanced methods that also employ proximity graphs).

Here we briefly explore the utility of proximity graphs through the lens of recommender systems. In a recommender system, we have a set of objects with a variety of properties, and we wish to provide a method for recommending other similar objects from the set given a choice of one or several objects from the set. A typical example would be the recommendations seen on music or book websites; when a customer purchases an item, the system recommends other items that they may find interesting.

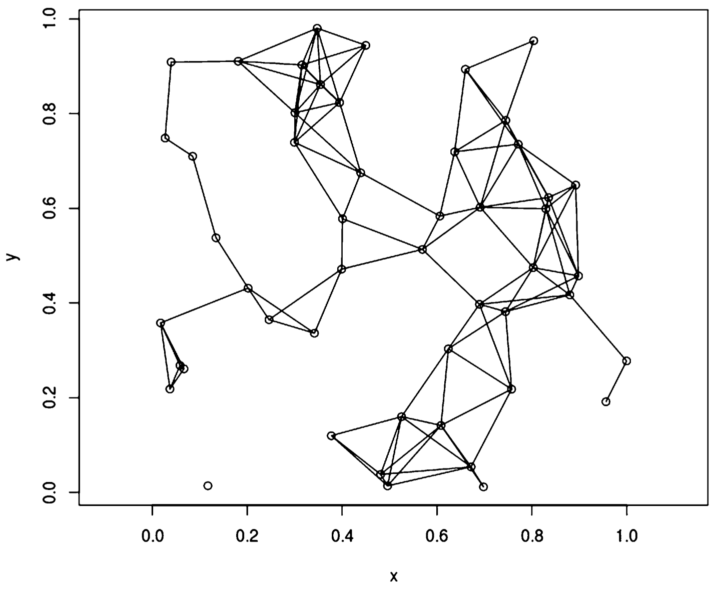


Fig. 4.10 Disk Graph for the example point set. For the example the unit length was taken as 0.2

Here we present a simple example using hotels in Venice, Italy, drawn from the travel website tripadvisor.com. For this example we reduced the total set of hotels to only 18 five star hotels, to provide a clearer example. Each hotel is rated by visitors and given from one to five stars.[10] We normalized the count of each rating category by the total number of ratings, and took each of these proportions as a dimension, giving a set of points in the five-dimensional unit hypercube. Using this

[10]Not to be confused with the amenity based star rating system!

point set (Fig. 4.11) we constructed a number of the proximity graphs mentioned in this chapter, presented in Fig. 4.12a–e.[11]

We can use the proximity graphs generated as a simple recommender system, or as part of a more complex system. Of course here we employ a simplified model; a fuller model would include other aspects such as location, price and availability.

Nonetheless, when choosing a hotel, it is reasonable that the customer would be interested in seeing similar hotels, particularly with regard to the quality of the hotel. The visitor ratings provide one measure of a hotel's quality and standing. Thus we can explore the idea that hotels that are "nearby" in ratings might be of interesting to the hypothetical customer. This naturally leads us to proximity graphs, which give a variety of ways of expressing "nearness" in multifaceted data sets.

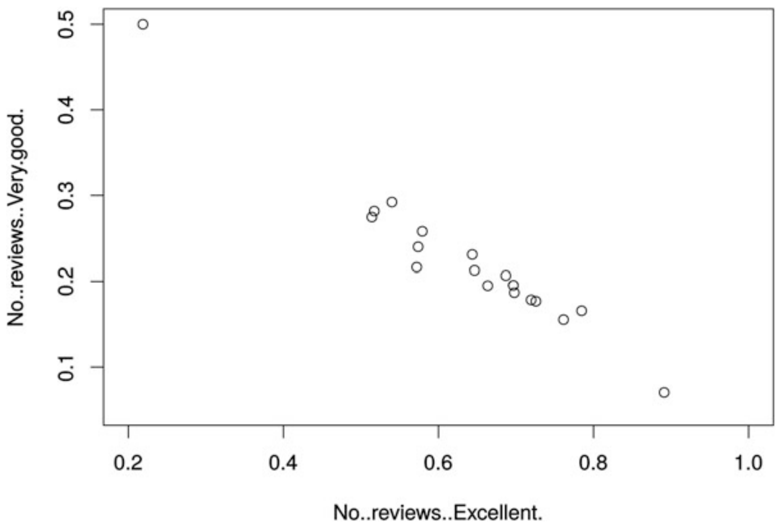


Fig. 4.11 The set of 5-star hotels in Venice, Italy. The arrangement is a 2-dimensional projection of the underlying 5-dimensional ratings data set

Quickly glancing at Fig. 4.12a–e, without particular consideration as to which proximity graphs they are, we can see that different methods provide noticeably different results.[12]

[11]Note that the figures are 2-dimensional projections of the 5-dimensional proximity graphs.

[12]Note that the choice of projective plane here is largely for convenience. The choice of two of the ratings categories to form the x and y dimensions in the image has no additional bearing on the graph itself, only its layout. Certainly other axes could be used; other selections of the ratings categories are of course possible, but we note that given the nature of ratings, they are all essentially pairwise correlated and have sharp distributions, leading to roughly the same point layout. We could also choose a projection unrelated to the underlying data, for example, we could generate the graph, then apply any of a suite of layout algorithms, divorcing the position of the vertices from any intrinsic meaning.

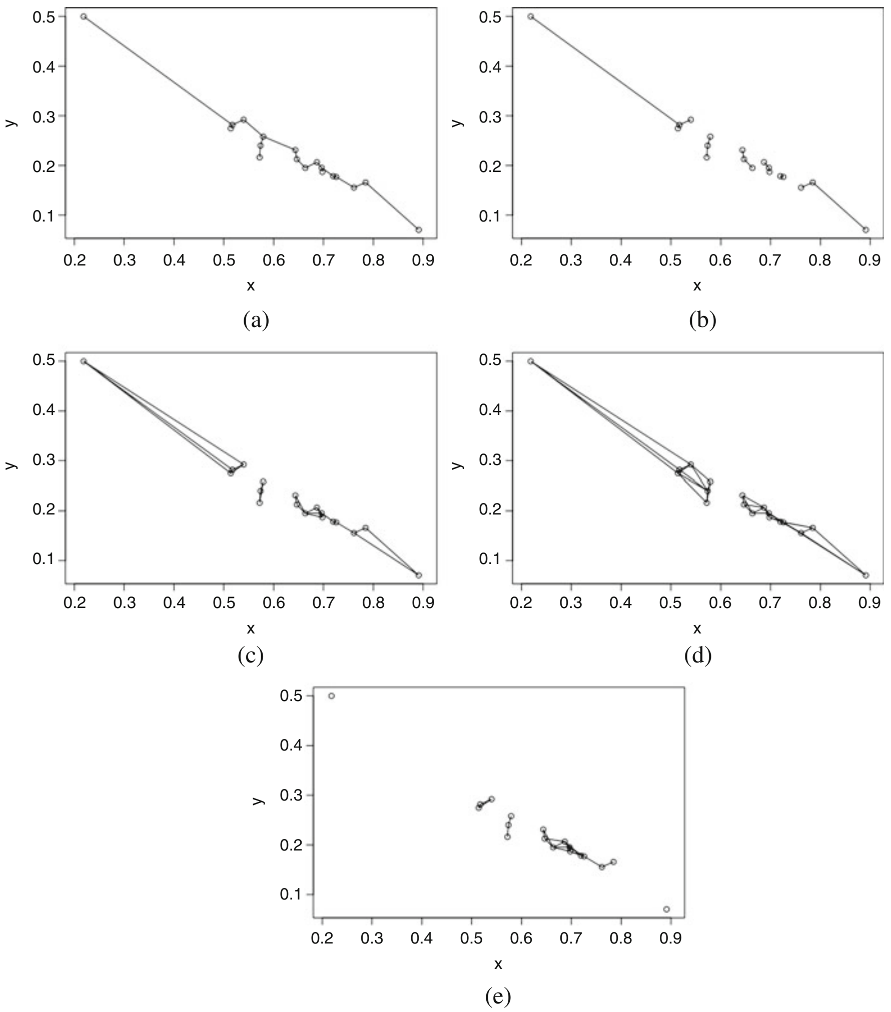


Fig. 4.12 (**a**) The Minimum Spanning Tree for the Venetian hotels TripAdvisor data set. The x axis is the proportion of "Excellent reviews", the y axis is the proportion of "Very Good" reviews. We note that for this particular data sets, the Relative Neighbourhood Graph, the Minimum Spanning Tree and the Shortest Path Graph happen to coincide. (**b**) The 1-Nearest Neighbour Graph for the Venetian hotels TripAdvisor data set. The x axis is the proportion of "Excellent reviews", the y axis is the proportion of "Very Good" reviews. (**c**) The Sphere of Influence Graph for the Venetian hotels TripAdvisor data set. The x axis is the proportion of "Excellent reviews", the y axis is the proportion of "Very Good" reviews. (**d**) The 3-Nearest Neighbour Graph for the Venetian hotels TripAdvisor data set. The x axis is the proportion of "Excellent reviews", the y axis is the proportion of "Very Good" reviews. (**e**) The Unit Disk Graph for the Venetian hotels TripAdvisor data set, with disk radius set to 0.05. The x axis is the proportion of "Excellent reviews", the y axis is the proportion of "Very Good" reviews

Unsurprisingly the MST (Fig. 4.12a) provides only limited utility for our purposes, but each vertex does at least have one other neighbour which is “close” and could be recommended as similar.

The 1-Nearest Neighbour Graph (Fig. 4.12b), although a subgraph of the MST, perhaps provides a better basis for recommendations as it separates the vertices into connected components that we can take as clusters of similar hotels.

The Sphere of Influence Graph (Fig. 4.12c) produces an output similar to the 1-NN, but with larger connected components, which could provide more flexibility and range in the recommendations possible.

The 3-NN (Fig. 4.12d) graph takes a further step in this direction, and indicates that there is a fairly clear distinction between two groups of hotels within this group. It is worth noting that these graphs guarantee that each vertex will have at least one neighbour, a possibly desirable property in a recommended system, as a recommendation can always be provided; however, this has the caveat that for data points that are truly unique, the recommendation will be extremely poor—note in particular the two extrema in the top left and bottom right corners, if these two dimensions constituted the full underlying data set, their inclusion with any other vertex in a cluster would be dubious.

The Unit Disk Graph (Fig. 4.12e) provides some remedy to this, although with drawbacks of its own. For this example the disk radius was manually selected as 0.05—in general it is not necessarily clear how to select the radius. However in this case this provides a series of clusters, but always excludes long edges, a desirable property for recommender systems.

In practice, and with more detailed data, this process could be taken further and the predictive power of each type of graph could be validated (for example, by using cross fold validation).

4.13 Conclusion

While we have briefly covered a number of proximity graphs, the topic includes much more than we can include here. Geometric graph theory in particular is a fertile ground for proximity graph definitions, and optimization provides a great number of applications, particular as components in larger algorithms. One unfortunate aspect however is the lack of a coherent presentation of detailed material on proximity graphs, with the literature scattered piecemeal across a number of fields.

Acknowledgements Pablo Moscato acknowledges previous support from the Australian Research Council Future Fellowship FT120100060 and Australian Research Council Discovery Projects DP120102576 and DP140104183.

Appendix: Example Point Set

Point	x-position	y-position	Point	x-position	y-position
1	0.723212968	0.178106849	26	0.907661273	0.729217676
2	0.920432236	0.485452615	27	0.545987988	0.340721248
3	0.551488565	0.873938297	28	0.181010885	0.597560442
4	0.306689395	0.414440461	29	0.254672002	0.906480972
5	0.252440631	0.386800466	30	0.105705	0.170610555
6	0.326211397	0.990150763	31	0.667526022	0.476145574
7	0.669619173	0.276130067	32	0.784634882	0.565077639
8	0.385344312	0.279983967	33	0.287186647	0.402820584
9	0.695385871	0.066082659	34	0.182059349	0.688221614
10	0.475169562	0.266587733	35	0.637400634	0.602165836
11	0.545891144	0.912037976	36	0.322844825	0.588968292
12	0.017635186	0.333899253	37	0.760253692	0.383415267
13	0.621307759	0.610259379	38	0.25928387	0.955812554
14	0.716927846	0.528621053	39	0.791903241	0.644445644
15	0.207482761	0.976842469	40	0.005358534	0.280421027
16	0.018216165	0.891276263	41	0.419955306	0.763752059
17	0.320578694	0.264128312	42	0.2918794	0.122258146
18	0.420885588	0.702595526	43	0.010059218	0.809975938
19	0.237547534	0.454149704	44	0.008974315	0.354484587
20	0.608278025	0.297481298	45	0.450824119	0.497067255
21	0.830447143	0.512086882	46	0.01188008	0.444212976
22	0.105544439	0.033963298	47	0.892440161	0.2077337
23	0.741506729	0.279505876	48	0.245942889	0.718327626
24	0.99144205	0.866106852	49	0.636074704	0.058158036
25	0.021779104	0.004131444	50	0.133528981	0.009221669

References

1. Fermat-Torricelli problem. In Michiel Hazewinkel, editor, *Encyclopedia of Mathematics*. Springer, 2001.
2. Pankaj K. Agarwal, Herbert Edelsbrunner, Otfried Schwarzkopf, and Emo Welzl. Euclidean minimum spanning trees and bichromatic closest pairs. *Discrete & Computational Geometry*, 6(3):407–422, 1991.
3. Richard Bellman. On a routing problem. *Quarterly of Applied Mathematics*, 16:87–90, 1958.
4. Jon L. Bentley, Donald F. Stanat, and E.Hollins Williams. The complexity of finding fixed-radius near neighbours. *Information Processing Letters*, 6(6):209–212, 1977.
5. Otakar Borůvka. O jistém problému minimálním. *Práce Moravské přírodovědecké. Společnosti*, 3(3):37–58, 1926.

6. A. Bowyer. Computing Dirichlet tessellations. *The Computer Journal*, 24(2):162–166, 1981.
7. Jean Cardinal, Sébastian Collette, and Stefan Langerman. Empty region graphs. *Computational Geometry Theory & Applications*, 42(3):183–195, 2009.
8. Bernard Chazelle. A minimum spanning tree algorithm with inverse-Ackermann type complexity. *Journal of the ACM*, 47(6):1028–1047, 2000.
9. Gustave Choquet. Étude de certains réseaux de routes. *Comptes-rendus de l'Académie des Sciences*, 206:310–313, 1938.
10. P. Cignoni, C. Montani, and R. Scopigno. DeWall: A fast divide and conquer Delaunay triangulation algorithm in e^d. *Computer-Aided Design*, 30(5):333–341, 1998.
11. Brent N. Clark, Charles J. Colbourn, and David S. Johnson. Unit disk graphs. *Discrete Mathematics*, 86(1):165–177, 1990.
12. Mark de Berg, Otfried Cheong, Marc van Kreveld, and Mark Overmars. *Computational Geometry: Algorithms and Applications*, chapter Delaunay Triangulations: Height Interpolation. Spring-Verlag, 2008.
13. Jesús De Loera, Jörg Rambau, and Francisco Santos. Triangulations, structures for algorithms and applications. In *Algorithms and Computation in Mathematics*, volume 25. Springer, 2010.
14. Edsger W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, pages 269–271, 1959.
15. H. Edelsbrunner and N. R. Shah. Incremental topological flipping works for regular triangulations. *Algorithmica*, 15(3):223–241, 1996.
16. Herbert Edelsbrunner, David G. Kirkpatrick, and Raimund Seidel. On the shape of a set of points in the plane. *IEEE Transactions on Information Theory*, 29(4):551–559, 1983.
17. Kazimierz Florek. Sur la liaison et la division des points d'un ensemble fini. *Colloquium Mathematicum 2*, pages 282–285, 1951.
18. Robert W. Floyd. Algorithm 97: Shortest path. *Communications of the ACM*, 5(6):345, 1962.
19. L. R. Ford. Network flow theory. Technical Report P923, RAND Corporation, Santa Monica, USA, 1956.
20. Michael L. Fredman and Robert Endre Tarjan. Fibonacci heaps and their uses in improved network optimization algorithms. *Journal of the ACM*, 34(3):596–615, 1987.
21. K. R. Gabriel and R. R. Sokal. A new statistical approach to geographic variation analysis. *Systematic Zoology*, 18:259–278, 1969.
22. Leonidas Guibas and Jorge Stolfi. Primitives for the manipulation of general subdivisions and the computation of Voronoi. *ACM Transactions on Graphics*, 4(2):74–123, 1985.
23. Ferran Hurtado, Giuseppe Liotta, and Hank Meijer. Optimal and suboptimal robust algorithms for proximity graphs. *Computational Geometry Theory & Applications*, 25(1–2):35–49, 2003.
24. Vojtěch Jarník. O jistém problému minimálním. *Práce Moravské přírodovědecké. Společnosti*, 6(4):57–63, 1930.
25. Jerzy W. Jaromczyk and Mirosław Kowaluk. Constructing the relative neighborhood graph in 3-dimensional Euclidean space. *Discrete Applied Mathematics*, 31(2):181–191, 1991.
26. Donald B. Johnson. Efficient algorithms for shortest paths in sparse networks. *Journal of the ACM*, 24(1):1–13, 1977.
27. David R. Karger, Philip N. Klein, and Robert E. Tarjan. A randomized linear-time algorithm to find minimum spanning trees. *Journal of the ACM*, 42(2):321–328, 1995.
28. David G. Kirkpatrick and John D. Radke. A framework for computational morphology. In *Computational Geometry, Machine Intelligence and Pattern Recognition*, volume 2, pages 217–248. North-Holland, Amsterdam, 1985.
29. Joseph B. Kruskal. On the shortest spanning subtree of a graph and the traveling salesman problem. *Proceedings of the American Mathematical Society*, 7:48–50, 1956.
30. Geoff Leach. Improving worst-case optimal Delaunay triangulation algorithms. In *Proceedings of the 4th Canadian Conference on Computational Geometry*, page 15, 1992.
31. D. T. Lee and B. Schachter. Two algorithms for constructing Delaunay triangulations. *International Journal of Computer and Information Sciences*, 9(3):219–242, 1980.
32. A. Lingas. A linear-time construction of the relative neighborhood graph from the Delaunay triangulation. *Computational Geometry*, 4(4):199–208, 1994.

33. Edward F. Moore. The shortest path through a maze. In *Proceedings of the International Symposium on Switching Theory 1957, Part II*, pages 285–292. Harvard Univ. Press, Cambridge, Mass., 1959.
34. Joseph O'Rourke. Computing the relative neighborhood graph in the L_1 and L_∞ metrics. *Pattern Recognition*, 15(3):189–192, 1982.
35. Robert C. Prim. Shortest connection networks and some generalizations. *Bell System Technical Journal*, 36(6):1389–1401, 1957.
36. S. Rebay. Efficient unstructured mesh generation by means of Delaunay triangulation and Bowyer-Watson algorithm. *Journal of Computational Physics*, 106(1):127, 1993.
37. Bernard Roy. Transitivité et connexité. *Comptes rendus de l'Académie des sciences*, (249):216–218.
38. M. Shamos and D. Hoey. Closest-point problems. In *Proceedings of the 16th Annual IEEE Symposium on Foundations of Computer Science*, pages 151–162, 1975.
39. M. Sollin. Le tracé de canalisation. *Programming, Games, and Transportation Networks*, 1965.
40. Peter Su and Robert L. Scot Drysdale. A comparison of sequential Delaunay triangulation algorithms. In *Proceedings of the Eleventh Annual Symposium on Computational Geometry*, pages 61–70, New York, NY, USA, 1995. ACM.
41. K. J. Supowit. The relative neighborhood graph, with an application to minimum spanning trees. *Journal of the ACM*, 30(3):428–448, 1983.
42. G. T. Toussaint. Comment: Algorithms for computing relative neighbourhood graph. *Electronics Letters*, 16(22):860, 1980.
43. G. T. Toussaint. The relative neighborhood graph of a finite planar set. *Pattern Recognition*, 12(4):261–268, 1980.
44. R. B. Urquhart. Algorithms for computation of relative neighbourhood graph. *Electronics Letters*, 16(14):556, 1980.
45. Remko C. Veltkamp. The γ-neighborhood graph. *Computational Geometry Theory & Applications*, 1(4):227–246, 1992.
46. Stephen Warshall. A theorem on Boolean matrices. *Journal of the ACM*, 9(1):11–12, 1962.
47. D. F. Watson. Computing the n-dimensional Delaunay tessellation with application to voronoi polytopes. *The Computer Journal*, 24(2):167–172, 1981.
48. A. Yao. On constructing minimum spanning trees in k-dimensional spaces and related problems. *SIAM Journal on Computing*, 11:721–736, 1982.

Chapter 5
Clustering Consumers and Cluster-Specific Behavioural Models

Natalie Jane de Vries, Jamie Carlson, and Pablo Moscato

Abstract Social media has almost become ubiquitous in everyday communications and interactions between customers and brands. A novel clustering algorithm, that has shown high scalability in previous applications, is applied to analyse and segment an online consumer behaviour dataset. It is based on the computation of a Minimum-Spanning-Tree and a k-Nearest Neighbour graph (MST-kNN). Cluster-specific consumer behaviours relating to customer engagement are predicted using symbolic regression analysis which, in a commercial setting, would provide the basis for personalized marketing strategies. Five major clusters were found in the dataset of 371 respondents who answered questions from theoretical marketing constructs related to online consumer behaviours. They are labelled as follows: 'Brand Rationalists', 'Passive Socializers', 'Immersers', 'Hedonic Sharers' and 'Active Participators'. For each of these clusters, a linear model of customer engagement was predicted using symbolic regression analysis. These models inform possible personalized marketing strategies after proper segmentation of the customers based on their online consumer behaviour, rather than simple demographic characteristics.

Keywords Brand · Customer engagement · Engagement · Loyalty behaviour · Online customer engagement · Customer engagement prediction · Segmentation methodologies · Symbolic regression analysis

N. J. de Vries (✉) · P. Moscato
School of Electrical Engineering and Computing, The University of Newcastle, Callaghan, NSW, Australia
e-mail: natalie.devries@newcastle.edu.au; Pablo.Moscato@newcastle.edu.au

J. Carlson
Newcastle University Business School, The University of Newcastle, Callaghan, NSW, Australia
e-mail: Jamie.Carlson@newcastle.edu.au

P. Moscato, N. J. de Vries (eds.), *Business and Consumer Analytics: New Ideas*,
https://doi.org/10.1007/978-3-030-06222-4_5

5.1 Introduction

Online consumer behaviours with a brand focus have increased in terms of volume and types of behaviours displayed, across many different online communication platforms. Due to the increasing capabilities of online platforms such as social networking sites, consumers are now able to communicate, interact and engage with brands, and each other in real-time and in many new and different ways [9, 20, 36]. These high levels of interactions and new approaches to communication create the conditions for online customer engagement which produce high levels of heterogeneity amongst customers of a particular brand as every customer may wish to interact with their favourite brand in a personalized manner. Due to these recent advancements in communicating with their customers, organizations and brand managers in particular are able to gain detailed insight into the behaviour of their consumers. Consequently, the data rich social media environment enables brands to provide a more personalized experience for their customers which increases levels of customer engagement resulting in improved brand performance outcomes [11, 25].

Besides tracking online metrics of customers' online behaviours towards brands, insights into motivations for these behaviours are also of high importance when examining the consumer base in a market. Segmenting their customers and the identification of 'customer typologies' is becoming a strategic priority amongst many organizations and brands with operations in the online environment [15, 30]. Of particular relevance for organizations, technology empowered and highly interconnected customers do not form one homogenous group in terms of their online behaviours but rather, customer segments that dynamically change over time [1, 36]. In the specific context of the social media environment, brand managers need to be able to segment their online customers appropriately with relevant metrics and rigorous analytical approaches [36]. The development of such analytical frameworks would allow an organization with a brand page in the social media environment to profile customers, divide them in groups sharing common characteristics and apply tailored online experiences to optimize the overall customer experience to improve customer–brand relationships, and ultimate sales opportunities.

In addressing the issue for distinct measurement approaches to guide managerial decision making, researchers have since stated that it is preferable to have an analytical framework without *a priori* knowledge in the identification of segments to data processing and pre-classification since such an approach fails at capturing possible interactions between the salient variables in a particular market setting [29]. In response, this study provides a novel clustering methodological framework to tackle segmentation and the identification of customer typologies problems in an online marketing context of the social media environment, which has been underexposed in the literature thus far.

Furthermore, Aviad and Roy [3] state results of clustering exercises often require considerable effort by the end user to interpret and find use in the results. Therefore, we develop cluster-specific models predicting customer engagement which identify how the respondents in different clusters differ in terms of their motivations for engaging with a brand through the social media platform. In addition to this, a new score, the CM1 score is used, which is an approach to identify the most salient features of particular clusters as introduced by Marsden et al. [31] in a study investigating Shakespearean era works.

From here on, this study provides a theoretical background to outline the relevant underlying literature in this area followed by a step-by-step description of our proposed methodology. After this, results of the computational experiments are presented and finally a discussion of results including future research directions is provided.

5.2 Theoretical Background

A theoretical background to the relevant areas in this study is provided. A brief overview of online consumer segmentation and existing clustering and Segmentation Methodologies is presented.

5.2.1 Online Consumer Segmentation

Customer segmentation has become a central concept in marketing and many organizations use segmentation to better serve and satisfy customer needs. As stated in Chap. 2, researchers have based the segmentation of markets on various factors, including cultural, geographic and socioeconomic variables as well as personality, life-style, user status and usage frequency. Customer segments based on these variables may be easy to understand and determine, but may not provide the best possible explanatory power [42]. As a consequence, marketing scholars highlight the need to account for heterogeneous customer perceptions and expectations in order to develop better marketing strategies and allocation of scarce organizational resources [13, 14]. Recent researchers have since argued that contemporary operative environments such as the Internet require sophisticated and sensitive segmentation methods and not ‘blindly’ follow the process of segmentation purely based on generic descriptors of consumers due to high levels of online consumer-to-consumer (C2C) interactions as well as more empowered consumers [30]. Given these underlying characteristics, researchers argue that segmentation should be seen as a tool in identifying factors with a causal relationship to future consumer behaviours towards a brand [4, 7]. Thus, consumers do not form one homogenous group when it comes to specific online consumer behaviours [1] and differences between these consumers need to be fully understood should marketers wish to target them.

As Bhatnagar and Ghose [4] argue, segmentation based on more than just demographic information has the capability to provide much more useful insights into consumers and understanding of specific consumer groups. Using only a single base for segmentation, such as age or gender, limits understanding of consumer groups [7]. This is particularly true for consumers' behaviours on the Internet. In previous Internet-based studies such as the investigation of web buyers, demographics cannot discriminate between different types of web buyers as demographics alone do not provide sufficient diagnostic information about web users [4]. In the social media and marketing literature in general, it has become a management imperative for the investigation of consumer behaviours and customer engagement with brands in more detail in the online environment [11, 12, 25]. However, Foster et al. [15] state that only a few studies have aimed to differentiate (or segment) between brand page users to uncover specific behavioural profiles along a continuum of participation. Furthermore, they continue to explain that the research dedicated to segmentation as a marketing tool in the online, and social media context more specifically, falls far behind the volume of research investigating more 'traditional' segmentation strategies. On this basis, a lag exists between academic literature investigating online consumer segmentation based on behaviours towards brands and what is practised in reality by organizations in the online marketplace.

Dividing online consumers into distinct groups (segments) with regard to their different needs, attitudes and behaviours is already commonly used in market research and is used as a basis to target these specific segments with tailored marketing programs [6]. However, findings from McKinsey & Company's (2012) paper on the 'iConsumer' and 'The World Gone Digital' when it comes to understanding digital behaviours, simple categorizations are 'not possible or wise'. In this sense, clustering consumers in the online and social media space needs to account for behaviours and attitudes across a full range of variables in order to find the full levels of diversity existent within the consumer market.

As previously discussed, interpreting and finding use in the results of clustering exercises often requires considerable effort by the user of these results [3]. It is also important to have more than just descriptive information on clusters of consumers as Bhatnagar and Ghose [4] state, diagnostic information should help practitioners make better decisions about their marketing strategies. In other words, having a true understanding of the consumer groups (or clusters) within a market is more useful to brand managers than simple demographic descriptions of multiple groups of people. It is for these reasons that it is important to find a method for describing and uncovering 'hidden' details within clusters. If a segmentation of a group of consumers is done based on a spectrum of behaviours, it is illogical to describe the resulting segments based solely on demographic information. Furthermore, predicting actual online behaviours within clusters cannot be conducted using information based on consumers' age, gender or income. It is for these reasons the current study employs novel methodologies such as symbolic regression modelling and the calculation of the CM1 score (discussed in the next section) to build cluster-specific behavioural models as well as describe the clusters more in depth in terms of behaviours in the social media platform.

5.2.2 *Clustering and Segmentation Methodologies*

When interested in segmenting and targeting a set or group of consumers with similar behaviours, it is a common practice in marketing research to employ clustering methods in order to group consumers [9]. Clustering consumers is not new in marketing as Klastorin in 1983 [27] discussed that finding homogeneous groups of consumers is beneficial for marketing practitioners and strategists. As Jain [26] (and Chap. 3 of this book) explains, clustering methodologies can be split into two different types: supervised (classification) or unsupervised (clustering). The goal of unsupervised data clustering is to discover the natural groupings of a set of objects or patterns [26]. This means that in order to find the natural groupings (or segments) within a group of consumers, limited to no parameters should be set prior to the clustering exercise; otherwise, the exercise would be closer to classification. As explained by Aviad and Roy [3], clustering activities attempt to partition a dataset into groups so that the entities in one group (segment) are similar to each other and are as different as possible from the entities in other groups (or segments).

A frequently used method in market segmentation is latent class analysis. It is one of the earliest methods adopted by social science and marketing researchers to separate a group of people into latent 'classes' or segments [17]. In more recent years, latent class analysis has been used to segment online consumers. Bhatnagar and Ghose [4] segment e-shoppers based on consumer perceptions and behaviour with respect to online commerce using a latent class analysis. Furthermore, Campbell et al. [7] segment 883 consumers based on their reactions to social network marketing using latent-class analysis and find a total of five segments with distinct characteristics. Other recent methods used in a market segmentation of consumers include k-Means cluster analysis, [4, 41], finite-mixture models in Structural Equation Modelling using Partial Least Squares [30] and a cluster analysis using a set number of variables using Ward's method for hierarchical clustering [19, 35]. However, what these methods all have in common is the number of algorithmic parameters that must be specified by the user. In other words, these methods become less 'unsupervised' when more parameters are introduced a priori. As Jain [26] explains about k-Means clustering methodology, for example, it is that the number of clusters (k) needs to be determined by the user as well as the cluster initialization and he goes on to explain that automatically determining the number of clusters has been one of the most difficult problems in data clustering. Aviad and Roy [3] also explain that in real-life data mining problems, there is no a priori classification making the division process into groups very difficult to define and construct. For these reasons, it is important to consider the number of parameters that are set prior to the clustering algorithm and attempt at limiting the number of user-defined parameters.

One advantage of data clustering consumers into homogeneous segments is that of the identification of central points of segments, which can be treated as ideal points to reflect the customer requirements of the consumers inside the segment [8]. In classification (supervised) problems, for example, a common objective is feature selection where the goal is to determine the best or most parsimonious set of variables for the algorithm. In clustering exercises, it is also important to find distinctive characteristics of the resulting groups but the goal is different to classification as the aim is to determine those features (or variables) that lead to a maximum differentiation of each cluster (segment) based on its members and their characteristics [3]. In the context of consumer segmentation in marketing, statistical analyses for significance such as an ANOVA test between clusters are conducted, e.g. [19, 41]; however, these methods are not capable of describing the clusters or finding those features that truly distinguish between clusters of consumers. In other scientific fields, various methods have been introduced to address this problem. One of these is the computation of the CM1 score which finds those features that are the most clearly identifiable characteristics of each cluster. This score has previously been applied in the identification of panels of biomarkers for breast cancer subtypes. In this instance it was used to identify the transcriptional state of genes that are consistently ‘over-expressed’ or ‘under-expressed’ in each subtype [32]. Analogously, it was used in a computational stylistic study that aimed to identify words which were used differently by Shakespeare and his contemporary peers [31]. In these studies, the CM1 score helped to analyse and describe the various clusters (or subtypes) in further detail and identify salient features of each group. We refer the reader to the publicly available online publication where the score was introduced [31] for its definition and its use in a different study. Methodological explanations of the use of this score in this context are provided in the following section.

5.3 Materials and Methods

This chapter uses the small online customer engagement dataset which is outlined in Sect. 26.2.4 of Chap. 26. In this dataset, 371 respondents answer questions about their online engagement behaviour with brands through Facebook. As stated in Sect. 5.1, insights are needed on top of online metrics in order to truly understand and capitalize on consumers’ motivations for engaging with brands and companies in the online environment. Here, the method of this study is outlined including the design, distance measure used for the clustering methodology, the CM1 score calculation and symbolic regression predictive modelling process.

5.3.1 Method Design

There are several stages to the methodology used in this study. Firstly, a brief outline will be given of the questionnaire tool construction and explanation of the dataset's basic characteristics. Then, the methodology for our clustering method is also conducted by going through several stages. Firstly, the Spearman's rank correlation coefficient is computed for all items for all respondents which results in the generation of a distance/similarity matrix. These distances will provide the basis for a graph which is a combination of a Minimum Spanning Tree (MST) and a k-Nearest Neighbourhood (kNN) Graph. The MST-kNN agglomerative algorithm and graph will provide the resulting clusters found by this study. Whilst generating these graphs, they will be visualized and analysed in order to describe and outline the segments found through our clustering method. In this, concepts such as 'node betweenness centrality', see Chap. 8 for the definition of this one and other measures. Basic layout algorithms are also used for visualization purposes. A brief inspection of the demographic and technology usage information will be done followed by the use of the CM1 score [31]. After this description of the clusters, each cluster will be analysed using symbolic regression analysis. The purpose of doing so is to identify mathematical models for average customer engagement levels to suit each cluster. This will further identify the characteristics that set each cluster apart, as it may be expected that different variables affect the level of customer engagement in each cluster.

5.3.2 Distance Measure

Before we continue with the clustering analysis in this chapter, a distance or similarity measure needs to be calculated for each of the data points. In this study, the Spearman's rank correlation coefficient was preferred. The Spearman's rank correlation coefficient is a non-parametric measure of statistical dependence between two variables which is appropriate for discrete variables, including ordinal ones. A perfect Spearman correlation between two variables may indicate that they are related via a monotonic function, whilst, in contrast, the Pearson correlation will only attain the maximum value when the two variables are related by a linear function. Herlocker et al. [21] have been critical about the assumption of linearity and since the Spearman's rank correlation does not rely on the assumption of linearity or other assumptions, it is preferred to be used in this case. Particularly due to the reason that we have a range of variables on a Likert scale we have preferred it as a measure of similarity [5].

5.3.3 Background of the MST-kNN Clustering Algorithm

Firstly, a brief introduction to Graph Theory is presented here in order to provide context. A simple undirected graph is denoted as $G(V, E)$ in which V is a non-empty set of vertices (also called nodes) and E is a set of unordered pairs of distinct elements of V called edges. An edge weighted graph is denoted as $G(V, E, W)$ in which V and E are defined as before but each edge now has associated a weight and W is a set of weights. We refer to the sets E and V as $E(G)$ and $V(G)$, to indicate that they are the set of edges of G and analogously, the set of nodes of G, respectively [18].

A path in a graph $G(V, E)$ is a sequence of edges which connects a sequence of vertices. In an undirected graph $G(V, E)$, we say that two nodes a and b are connected if the set of edges E contains a subset of them that form a path between nodes a and b. A graph is said to be connected if every pair of vertices in the graph is connected. A connected component of a graph is a maximal connected subgraph of G; in this case, each node and each edge belong to exactly one connected component. A simple undirected graph is a tree if in it any two vertices are connected by exactly one simple path. A graph is a forest if it is a disjoint union of graphs that are all trees, which means that in a forest all the connected components are trees. Given a connected, simple undirected graph $G(V, E)$, a spanning tree of that graph $(MST(G))$ is a subgraph that is a tree and connects all the vertices together. Given a graph G we can enumerate all its spanning trees and order them according to the total sum of weights of all edges of the tree. Accordingly, a tree is a Minimum Spanning Tree of G, denoted as $MST(G)$, or the Minimum Weight Spanning Tree if it is a spanning tree of G with the total sum of weights of its edges (its weight) being less than or equal to the weight of every other spanning tree of G.

Inostroza-Ponta et al. [22] have proposed a clustering algorithm known as MST-*k*NN; an agglomerative method combining the outputs given by a Minimum Spanning Tree and the k-Nearest Neighbour (*k*NN) algorithms. This method has been further explained in Chap. 3. The MST-*k*NN method has been tested on comprehensive studies on large-scale biological weighted networks and it has been successfully applied in various areas, see, for instance [2].

MST-*k*NN performs better than some other known classical clustering algorithms (e.g. k-Means and SOMs) in terms of homogeneity and separation [22, 24] in spite of not using an explicitly defined objective function, but using a clear stopping criterion instead. Due to its characteristics, it performs well even if the dataset has clusters of different mixed types (i.e. MST-*k*NN is not biased to 'prefer' convex clusters or when the data has clusters that are embedded in subspaces of different dimensionalities). Most importantly, the MST-*k*NN algorithm scales very well, allowing the possibility that the methods described in this paper can be extended to the analysis of very large datasets including questionnaires and other marketing datasets involving millions of consumers, online behaviours, brand pages or even brands on dedicated hardware.

The MST-kNN can be classified as a constructive heuristic that is not biased for the choice of a particular objective function, yet it provides a strong guarantee of optimality of a property of the final solution. We explain this property after we explain the algorithm. First, the algorithm's input can be either a distance matrix between all pairs of nodes or a weighted graph. In this study, a dissimilarity matrix that it is computed from the Spearman's rank correlation matrix is the input for the algorithm. Formally, if $r(a, b)$ is the Spearman's rank correlation between two respondents a and b over the set of questions, then the corresponding distance matrix $D = [d(a, b)]$ with each coefficient is calculated as $d(a, b) = 1 - r(a, b)$. Given this input matrix D, the output of the MST-kNN algorithm is a forest. This means that the MST-kNN generates a partition of a set of nodes given as an input using the information of similarities/dissimilarities between each pair. It not only gives a partition of the nodes but some of them are connected by edges.

We mentioned that the algorithm returns a forest that satisfies a property. The set of nodes are the ones that are part of the input. In the forest given as output, any edge of the forest that connects two nodes does so if the edge is one of the edges of the minimum spanning tree ($MST(G)$) and, at the same time, it is also an edge present in the set of edges of the k-nearest neighbour graph ($kNN(G)$). The k-NN graph is the graph that has one node per object and that has an edge between each pair of nodes, for example, a and b, if either a is one of the k nearest neighbours of b or if b is one of the k nearest neighbours of a, or both. We note that edges of the minimum spanning tree are not bound to have this property regarding 'k-neighbourness' and, the addition of this extra constraint has the effect of disconnecting the MST, thus creating a multi-tree forest and consequently leading to a natural partition of the set of nodes.

There are several variations of this scheme. In one of them, the value of k is set up to a relatively large value which is linked to the total number of nodes, and then, when the MST is fragmented in different components, a different value is selected for the different connected components using the same formula but now having for each of the connected components the number of nodes in each of them as input, thus leading to different values of k for each component. Another approach is the one we have used in this work, in which a value of k is fixed. In this paper we studied the cases of $k = 1$ and with the automatic selection of k in this study (i.e. $ln[n] = ln[371]$, since we have 371 samples in this dataset). Inostroza-Ponta et al. [22–24] outline the details of these methods and their applications to other real-world problems.

5.3.4 CM1 Score Calculation

After the clustering analysis, the clusters were investigated based on demographic and technology usage information. However, as stated, we also found that simple demographics do not provide intricate detail in the true underlying behavioural characteristics of each cluster. Therefore, in order to further describe and investigate the clusters in terms of online consumer behaviours, customer engagement and attitudes, we use the CM1 score to rank the answers of respondents that belong to two clusters. The score was recently introduced in a comprehensive study of the identification of word usage that would discriminate between authors. It was applied to produce models of authorship of a large group of plays from the Shakespearean era in [31]. Like the t-test, in order to calculate the CM1 score of the participants' responses to a question we first need to compute the difference between means of samples in the two groups of participants X and Y. Typically, X is the set of participants in a cluster and Y is the set of participants which are not in it. The distinctive characteristic of this score in comparison with the t-test is that it is moderated by the range of values of the set that has the largest set of samples, rather than the combined standard deviation of X and Y. We refer to Marsden et al. for details of this score [31].

5.3.5 Symbolic Regression Analysis

In this study the aim is to identify clusters (segments) of consumers that appear naturally based on their online behaviours, rather than examining descriptive information and without setting a priori parameters. To continue this data-driven approach, symbolic regression is used in order to find models for customer engagement for each of the clusters, which is a concept that has received increasing interest in recent years in the area of online consumer behaviour [25, 39]. Considering we are segmenting consumers based on their online behaviours, it is of considerable interest to generate cluster-specific models to predict online customer engagement with brands. As we have argued, interpreting results of clustering and segmentation exercises could include considerable effort by those marketing managers who are the end users of the clustering results. Therefore, we argue that investigating cluster-specific behavioural models assists marketing and brand managers in interpreting clustering and segmentation results.

Unlike numerical regression methods, in which model hypotheses are generated and fit to available data, symbolic regression discovers not only the coefficients within a structure, it also involves the discovery of the structure in the data and, consequently searches for the structure of the resulting models. Symbolic regression is defined as 'finding a mathematical expression, in symbolic form, that provides a good, best or even perfect fit between a given finite sampling of values of the independent variables and the associated values of the dependent variable(s)'. Stated more simply, the process of symbolic regression involves finding a model that best fits a given set of data. The main advantage of this method is that the researcher does not have to specify the structure of the regression model in advance [16].

In order to keep the method proposed in this study easily adoptable for future research by marketers and researchers, we use an open access software package named Eureqa [37]. Eureqa provides the user with a clear user interface, is free for academic use and the output is a Pareto optimal curve that trades model fitting for its complexity which aids the researcher in making the decision of this trade-off which is a significant problem as Smits and Kotanchek [38] point out. In this contribution, we have used the Eureqa Desktop package for the identification of models of average customer engagement in different clusters. We have restricted the search to only models that employ the basic 'building blocks' of 'addition', 'subtraction', 'multiplication', 'introduction of a constant value', 'the introduction of integer and real values' and 'the introduction of new input variables' in the expressions it generates. The two objectives of the Pareto Optimality Curve are the fitness of an error metric which represents the expression accuracy (which is user defined) and the 'complexity' of the model. Eureqa uses an ad hoc approach to define what the 'complexity' of a model is, which is the sum of the complexities values attributed to the use of each of the individual operations used to generate the formula. Our selected basic 'building blocks' have the minimum individual complexity. The 'error metric' that we have used was the 'Correlation coefficient', this means that during the evolutionary computation procedure Eureqa iteratively tries to find models that maximize the normalized covariance. Putting these things together, Eureqa helps to try to find a scale and offset invariant model that has the 'shape' of the data whilst at the same time uses as few input variables and mathematical operations as possible, giving a good trade-off of input selection and trend behaviour without risking over-fitting the data.

Through employing symbolic regression analysis, we aim to find a function for levels of online Customer Engagement within each cluster. In doing so, we iterate the same process for each of the clusters found by the MST-*k*NN clustering algorithm and subsequently select the best-fitting linear solution as found by Eureqa. This means that at the end of this process we have a function for Average Engagement (ENG) for each of the clusters that shows which of the input variables are relevant to that cluster. The results of this process are presented in the following section.

5.4 Results

As outlined in the materials and methods section, there are several stages to this study. Therefore, the results will be presented according to the order of these stages.

5.4.1 Distance Measure

As explained in the previous section, the basis for the clustering algorithm in this study is a Spearman-based distance matrix. As the sample for this study contains 371 respondents, this is a 371×371 matrix containing all the values for the Spearman correlation. In order to investigate this information, we identified those respondents who are the ‘closest’ in terms of Spearman correlation and those who are the ‘furthest’. In so doing, we also aim to emphasize the level of heterogeneity amongst customers in terms of their online behaviours with a brand focus.

Figure 5.1 shows two participants who have similar answer profiles of which one has selected an online clothing retailer (BlackMilk Clothing) and one a photography business (see Bliss Photography). These brands are two different types of businesses, an online fashion retailer and a photography service. However, these two respondents are the most similar in terms of their Spearman correlation in the whole sample, showing that those customers of the same brand do not necessarily have to be similar in their behaviours towards the brand. This highlights the focus of this study of finding homogenous groups of people in terms of their behaviours rather than their basic or demographic information. This also means that companies may need differentiated online strategies for their various customers across varying brand levels or sub-brands.

Figure 5.2 shows two participants who have selected a travel webpage (HIS Travel) and a clothing brand retailer (Portmans). These two figures illustrate that there exists wide heterogeneity in the responses of the participants regarding their online consumer behaviours and were therefore found to have the ‘furthest distance’ from each other based on Spearman correlation. As shown in Fig. 5.2, for the first half of the questions in the survey the respondent who had selected Trip H.I.S. answered significantly higher than the respondent who answered Portmans. Amongst these questions were questions about the respondent’s Usage Intensity of the branded social media page, their brand involvement with that brand, their level of self-brand congruency with that brand’s image as well as the levels of functional value and hedonic value of the brand’s Facebook page. It is in the questions regarding ‘flow’ and the level of perceived social value of the brand’s Facebook page where the respondents are closer in their responses. However, for the remainder of the questions, the respondent with the brand Portmans answers higher than the respondent with Trip H.I.S. These questions relate more to the actual experience on the brand’s Facebook page and their intention to interact and engage with the brand online in the future. We can speculate one possible reason for these differences in responses. The person who selected Trip H.I.S. may be highly involved with this brand but may not have such a good experience online, whereas the opposite may be true for the Portmans respondent.

As shown in both Figs. 5.1 and 5.2, heterogeneity exists amongst online consumers in terms of their behaviours and perceptions of the online experience. Therefore, we continue our analysis of this sample. After the Spearman-based distance matrix is computed, a minimum spanning tree is created. In this minimum spanning tree, the k-Nearest Neighbour algorithm is subtracted to reduce the number of edges which results in clusters (or, a subset of trees) that contain nodes that are only connected when they are nearest neighbours as explained in the previous section.

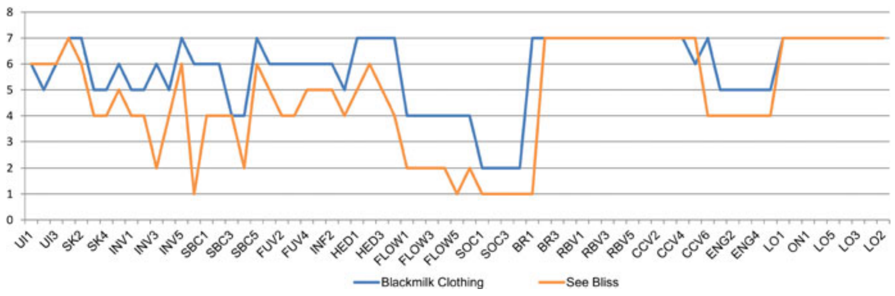


Fig. 5.1 Homogeneity between respondents—the pair of respondents that have shown the highest degree of homogeneity (closest in Spearman's Rank correlation) and their selection of two different brand choices

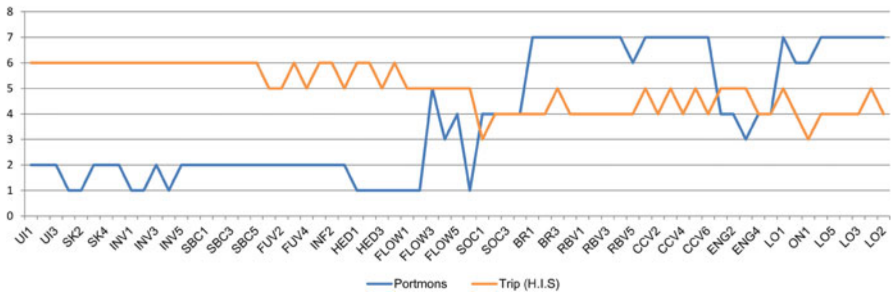


Fig. 5.2 Heterogeneity between respondents—the pair of respondents that have shown the highest level of heterogeneity in their responses ('furthest' in Spearman's Rank correlation) and their choices of brands

5.4.2 MST-kNN Clustering Results

In this section the results of the MST-kNN clustering method are presented that have been applied as described in our methodology. Firstly, we present the clustering method with the value of $k = 1$, followed by the automated value of k using the natural logarithm of n. Figures of these outcomes are displayed.

5.4.2.1 MST-*k*NN Algorithm with $k = 1$ Results

As stated, we have used the similarities produced by the Spearman's rank correlation method to create a distance matrix amongst all the respondents of the questionnaire. If we denote with $r(a, b)$ the Spearman correlation between two respondents a and b, then we have computed their distance as $d(a, b) = 1 - r(a, b)$. Using this distance matrix between respondents, we then applied the MST-*k*NN algorithm to find a clustering of the respondents in highly similar groups.

The results for the MST-*k*NN clustering method with $k = 1$ are presented in Fig. 5.3. We can see that all the respondents are grouped in 45 different clusters.

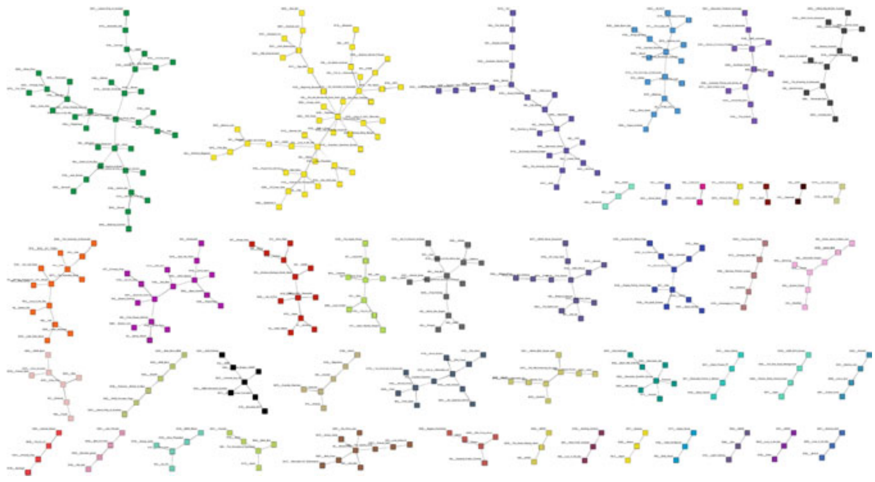


Fig. 5.3 Results of MST-*k*NN with $k = 1$. A result of 45 small clusters is found, which means that this result is likely to be the 'upper bound' of the heterogeneity in this dataset

Edges connect the respondents and each of these edges belong to a minimum spanning tree computed from the distance submatrix obtained by only taking into consideration the participants in that cluster.

An important property to highlight to the reader, when examining the figure, is that when the value of k is set to 1, in the shown solution of the MST-*k*NN, each edge that connects two respondents a and b indicates that either a is the most similar respondent to b in the entire dataset, or b is the most similar respondent to a in the entire dataset, or both. This means that this partition and visualization also offers an interesting alternative for data exploration and provides some insights on cluster structure. Results with $k = 1$ are a natural bound for this method, indicating that 45 is likely the upper limit of the heterogeneity present in the natural clusters in this dataset when this approach is used.

5.4.2.2 MST-*k*NN Algorithm with Automatic Setting of the Value of *k*

The alternative approach for computing a high-level description of the dataset, and a partition in clusters having a larger number of respondents is by direct application of the MST-kNN algorithm now with the automatic selection of k. The automatic selection of the value of k takes the natural logarithm of n, in this case $ln(371) = 5.92$ which is rounded up to $k = 6$. The algorithm will produce both a partition of the set of respondents in clusters, but also, like before, produce a set of edges connecting those respondents. An edge between two respondents a and b will indicate, in this case, that either person a is the closest (or up to the sixth closest) person of person b, or that b is the closest (or up to the sixth closest) person of person a, or both. The final result is found in Fig. 5.4 indicating five clusters (segments) of respondents.

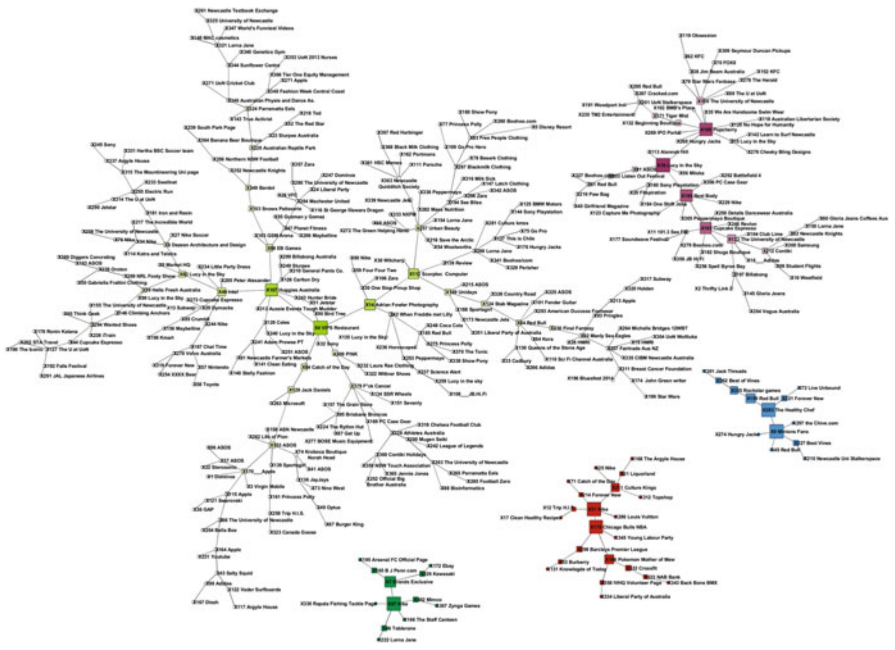


Fig. 5.4 Results of MST-kNN with the default setting for k. Five clusters are found when the value of k is automatically determined (and rounded up so $k = 6$ in this case). The size of the nodes indicates 'node betweenness centrality'. Different colours represent which cluster the nodes are part of and the shade of the node colours also represents 'node betweenness centrality'

The clustering with the automatic selection of k gives us five clusters of consumers in the data. One of these is clearly much larger than the other clusters; however, we are still able to examine these clusters in order to describe them. The largest cluster identified, Cluster One ($n = 250$) is shown in light green in Fig. 5.4 and has several large nodes based on computing the 'node betweenness' centrality measure.

The large nodes at the very centre of Cluster One represent respondents who selected a restaurant, a nappy brand, a photographer's brand page and a consumer electronics brand page in the questionnaire survey. The second largest cluster identified is Cluster Two ($n = 68$) and is shown in purple in Fig. 5.4. The largest nodes in this cluster represent respondents who selected a cafe, two online shopping retailers and a fitness page. Cluster Three ($n = 12$) is shown in dark green in which the two central largest nodes stand for a respondent who answered in reference a sports apparel brand and another respondent who answered in reference to an online shopping retailer.

Cluster Four ($n = 13$) is shown in blue with the central nodes in this cluster representing respondents who selected an energy drink brand, a healthy food brand page and a movie brand fan page. Finally, Cluster Five ($n = 22$) shown in red and labelled has several large nodes including respondents who answered a 'street wear' clothing brand, a sports apparel brand, an American basketball team page and a commercial television series brand page. Again, the central nodes in these clusters show that people who are homogenous in terms of their online behaviours and ways in which they communicate and interact with their 'favourite' brands online could come from a very heterogeneous group in terms of actual brand preferences and other descriptive information.

5.4.3 Describing the Cluster Results

In order to describe and analyse the clusters found by the MST-kNN algorithm we will conduct several steps. Firstly, each of the five clusters will be described in terms of their demographic information and their technology usage profile. Secondly, we will compute the CM1 score to identify those behaviours (variables) that are of particular importance in identifying each cluster. Finally, the symbolic regression analysis is conducted as described in our proposed methodology. Following this, we finish with a discussion of our results and a guide for future research suggestions.

5.4.3.1 Basic Description of Clusters

As shown in Fig. 5.4, Cluster One is the largest cluster comprising 250 respondents. Within this cluster, the age ranged from 17 to 49 years old and consisted of 41.8% males and 58.2% females. Furthermore, 73.8% of respondents in this cluster chose a service brand in the survey questions and 26.2% a product brand. In the technology usage profile of this cluster, we can find that the majority are experienced Facebook users having had an account for over 3 years (82.0%) who have 'liked' the page they selected for 6 months or more (62.5%). What is also interesting about these experienced social media users is that the majority of respondents access the Facebook platform from a mobile telephone or device. Consumers accessing Facebook from a laptop, mobile tablet or mobile telephone account for 86% of the sample.

Figure 5.4 also shows that Cluster Two is the second largest cluster found in the data. In this cluster, the age ranges from 18 to 45 years old with an average age of 21.38 years old. This cluster comprises of 33.8% male respondents and 66.2% female respondents. Furthermore, in cluster two, 83.8% of respondents selected a service brand and only 16.2% selected a product brand. In examining the technology usage profile of cluster two, it is clear that this cluster has similarities with cluster one. The majority of respondents in this cluster have had a Facebook account for longer than 3 years (80.9%) and have used and interacted with the Facebook brand page they selected for more than 6 months (55.9%). This shows that clusters may not necessarily be identified solely based on basic information. Furthermore, the majority of the cluster accesses the Facebook brand page from a mobile phone device (52.9%) and state that they are signed in all the time (45.6%). On this basis, we can deduce that this cluster forms an experienced group of Facebook users who are 'tech-savvy' accessing the social media platform mostly from a mobile device. This may have implications in the way that they interact with brands through social media due to the different functions and displays of mobile devices to desktops.

An examination of Cluster Three's descriptive information indicates similarity to the whole sample and other clusters, where the average age in Cluster Three is 21.25 years old ranging from 19 to 30 years. Furthermore, there was an exact 50% split between male and female respondents in this cluster and 53.8% of those respondents selected a service brand, whilst 41.7% of respondents selected a product brand. The technology usage profile of cluster three shows that although overall this cluster is still similar in terms of average age, duration of having a Facebook account and so forth, there are a few differences in this cluster. For example, 25% of this cluster accesses the Facebook brand page from a home desktop PC which is higher than the other clusters. This may impact on the way in which these consumers interact online and the way in which they like to use the social media site. Moreover, only 33.3% of this cluster indicate they are 'signed in all the time' through a mobile device which is also lower than the other clusters.

An examination of Cluster Four indicates an average age of 21 years old with the age ranging from 18 to 30 years. 61.5% of the respondents in this cluster are male and 38.5% are female. Similar to the other clusters, more respondents selected a service brand (76.9%) than a product brand (23.1%).

Analysing Cluster Four's technology usage shows that these respondents are experienced Facebook users with 84.6% having had a Facebook account for 3 years or more and 61.5% of the respondents indicate being 'signed in all the time'. This figure is almost twice in size as that of Cluster Three where only 33.3% of respondents indicate being signed in all the time. Furthermore, consistent with the respondents of Cluster Four being signed in all the time, 53.8% of these respondents access Facebook from a mobile device.

An examination of Cluster Five ($n = 25$) indicates an average age of 22 years old with the age ranging from 18 to 41 years which is slightly older than Cluster Three and 4. In this cluster, 50% are male and 50% female. Again, more respondents chose a service brand in their responses (72.2%) than those who chose a product brand (27.3%). Cluster Five again shows a group of respondents who are experienced Facebook users with 81.8% of respondents having had an account for 3 years or more and 59.1% indicating that they are signed in all the time. Exactly 50% of this sample answer that they access the brand's Facebook page from a mobile device and 40.9% from a laptop. This may indicate that these respondents engage in these online activities on the go which may impact on their actual online behaviours towards brands.

Based on the above analysis, the demographic information is not providing novel insights into understanding each cluster and its characteristics. This being the case, we propose the use of the CM1 score in the context of online consumer behaviour and segmentation studies as a better approach to investigate and describe clusters (or segments) of consumers.

5.4.3.2 CM1 Score Results for Clusters

The characterization of the major differences between the clusters is presented using the CM1 score to describe and label the clusters in further detail and provide further insights. Figure 5.5 shows the curves of the CM1 scores ordered from smallest to largest values which show the 'lowest' to the 'highest' scoring identifiable features for each cluster. At the very bottom and very top of each of the bar charts, several features protrude further than other bars which we call the 'shoulders' of each of the CM1 score curves.

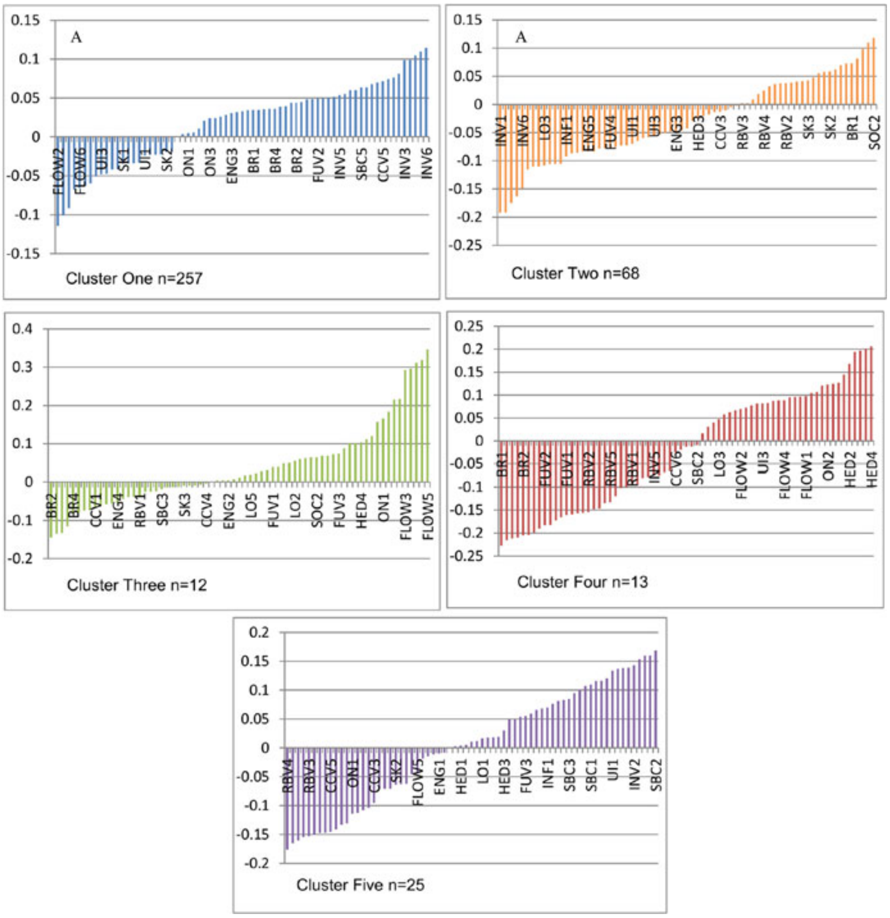


Fig. 5.5 CM1 Score computations for each cluster—the CM1 score has been computed for each cluster. The values were sorted from lowest to highest showing the 'bottom' to 'top' CM1 scores for each cluster. They are presented in the five graphs with data from Clusters 1 to 5, respectively. For most clusters bottom and top 'shoulders' can be seen in which there are several features that are higher or lower than the general trend in each curve

These features are also shown in Table 5.1 where each of the bottom and top five features are shown for all clusters. Those features that have formed a 'shoulder' in the curves of Fig. 5.5 are shown in bold in Table 5.1. Some clusters did not have a 'shoulder' on one or both ends and for these clusters the bottom five or the top five features are used for analysis.

Starting with Cluster One, we see that all bottom five features include the features relating to the Flow construct. When a person experiences 'flow' when using a web page, social media page or is engaged in an activity that person is in a perceived state of effortless action, loss of time and sense that the experience stands out as being

Table 5.1 Five bottom and top CM1 Scores for each cluster—highlighted in bold are those features that represented a 'shoulder' in the CM1 score graph for that cluster

Cluster 1	Cluster 2	Cluster 3	Cluster 4	Cluster 5
Five 'lowest' CM1 scores				
FLOW2	**INV1**	**BR2**	BR1	RBV4
FLOW5	**INV4**	**CCV5**	SOC1	RBV2
FLOW1	**INV2**	**BR1**	INF2	RBV5
FLOW4	**INV3**	**SK2**	INV6	BR2
FLOW6	**INV6**	BR4	BR2	RBV3
Five 'highest' CM1 scores				
INV3	BR1	**FLOW3**	**HED2**	INV2
INV2	SK5	**FLOW4**	**HED1**	**UI3**
INV1	**SOC4**	**FLOW2**	**FLOW5**	**UI2**
INV4	**SOC1**	**FLOW1**	**ON1**	**INV4**
INV6	**SOC2**	**FLOW5**	**HED4**	**SBC2**

exceptional when compared to 'normal' activities [34]. For example, the question relating to FLOW2 (the very most bottom CM1 score) stated: 'When I am visiting (using/operating) my favourite Facebook brand page: I lose track of time' (rated on a scale 1–7). For this cluster this means that these consumers, on average, have lower scores for questions relating to flow than all other clusters. When inspecting the technology usage profile of this cluster, it may provide a reason why this cluster scores low on flow. Nearly 60% (59.4%) of this cluster indicated that they engage with the Facebook brand page through a mobile telephone or a mobile tablet. For a consumer to experience flow, the environment needs to be appropriate to 'lose time and sense of self' which means that this may not be possible on small mobile device screens to deliver a media rich experience for flow to occur. Furthermore, the top features of this cluster show only variables relating to the 'Brand Involvement' construct which aimed at gathering an understanding of the respondents' previous (offline) involvement with the brand. For example, the very most top feature, INV6 stated: 'I am involved in/with this brand'. Considering that the top CM1 features only included brand involvement variables, we can understand that the consumers in this cluster engage with this brand online as they feel as though they are 'involved' in/with this brand, that the brand is significant and important to them or whether the brand means a lot to them and not just to entertain themselves and 'lose track of time' online. This being the case, we label this cluster as 'brand rationalists' since members of this group concentrate their time in consuming brand pages via a mobile device for convenience and given they are more highly involved fans of the brand than other respondents, content containing cognitive aspects of a utilitarian nature can be argued to be of greater importance to this segment than other respondents.

As can be seen in Table 5.1, Cluster Two shows an interesting contrast where all bottom CM1 scores are exactly those variables that are in the top of Cluster One's CM1 scores. That is, five of the six items relating to the '**Brand Involvement**' construct. This means that the respondents in this cluster have, on average, a lower 'involvement' with the brand they engage with online than all other clusters. The three most salient features for this cluster, and those that represent a ‘shoulder’ in the CM1 curve all relate to the ‘Social Value’ construct. The strongest variable, SOC2 stated ‘I can meet new people like me on this Facebook brand page’ (rated from 1 to 7). This indicates that these respondents perhaps use the Facebook brand page in a different way; to socialize and interact, rather than become truly involved in the brand. This being the case, we label this cluster as ‘passive socializers’ since members of this group concentrate their time interacting with other consumers on the brand pages via a laptop or mobile device at least once a week. Given these consumers are lowly involved fans of the brand, content of a hedonic, utilitarian and co-creative nature is of greater importance to this segment than other respondents.

Cluster Three also provides an interesting case to compare to Cluster One as all the items relating to ‘Flow’ that are found in Cluster One’s bottom features appear in Cluster Three’s top CM1 score features. Oppositely to Cluster One, this means that these respondents, on average, have higher scores for questions relating to ‘flow’ than all other respondents. Interestingly, 50% of this cluster responded that they access the Facebook brand page from either a home desktop PC or a laptop computer, as shown in this cluster’s technology usage profile. As stated, for ‘flow’ to occur, the environment has to be suitable and this could be a reason why this cluster is more likely to experience ‘flow’ when using the Facebook brand page.

An examination of Cluster Three’s bottom CM1 scores indicates a small variety of additional features. For instance, BR2, the lowest CM1 score belongs to the ‘Brand Interaction Value’ construct and specifically the item ‘I can communicate with the brand on this Facebook brand page’. The second lowest CM1 score, CCV5 comes from the ‘Co-creation Value’ construct and the item: ‘The Facebook brand page allows my involvement in providing services to me to get the experience that I want’. This suggests that these respondents are perhaps less likely to seek direct interaction and dialogue with the brand and more likely to ‘browse’ and ‘lose track of time’ whilst using and consuming content on the Facebook brand page. These stark contrasts between clusters provide great insights to the motivations and reasons for consumers interacting and engaging with the Facebook brand pages. This being the case, we label this cluster as ‘immersers’ since members of this group concentrate their time interacting with the brand page from home desktop PC, laptop and mobile device at least once a week and given they are lowly involved fans of the brand, content of a utilitarian and hedonic nature to induce flow experiences is of greater importance to this segment than other respondents.

Cluster Four and Five show different and unique cases. Cluster Four specifically does not have a conspicuous bottom ‘shoulder’ of CM1 scores; however, when analysing the bottom five features, two ‘BR’ and two ‘INF’ items appear. As stated, BR related to ‘Brand Interaction Value’, whereas INF relates to ‘Informational Value’. Similarly to Cluster Three, perhaps these respondents are less concerned about having direct interaction with the brand through the Facebook page and less likely than other respondents to seek valuable information through the Facebook platform. The top CM1 scores for Cluster Four include three items from the 'Hedonic Value' construct, one 'Flow' item and one 'Online Loyalty Behaviour' (ON) item. Hedonic value relates to how entertaining and 'fun' that respondent perceived the Facebook brand page to be. For example, the very top CM1 score, HED4 stated: 'The content of the Facebook brand page is entertaining'. One could argue that when a person has fun, or is entertained by a certain online activity, it is more likely for that person to enter a 'state of flow' which could explain the reason for these items appearing in this cluster's top CM1 scores together. The item relating to online 'loyalty' behaviours, ON1 specifically stated: 'I will share this brand's Facebook page content in the future'. Logically, if a person thinks that specific content is 'fun' and 'entertaining' they are more likely to want to share this with their friends. This being the case, we label this cluster as 'hedonic sharers' since members of this group concentrate their time interacting with the brand page for the enjoyment, fun and entertainment of the social media experience on the brand page whilst sharing content with others to support the need for enjoyment and hedonic gratification. Therefore, content predominately hedonic in nature combined with social and utilitarian content is of greater importance to this segment than other respondents.

Finally, Cluster Five also does not possess a specific bottom 'shoulder' of CM1 scores, which is why we investigate the bottom five features. Four of these features relate to the 'Relationship Building Value' construct (RBV) and one 'Brand Interaction Value' construct. These two constructs include statements about interaction, relationships and direct communication between the respondent and the brand on Facebook. For example, the most bottom feature (RBV4) stated 'The Facebook brand page is committed to delivering add-in values (e.g. special offers, member programs) to keep me loyal to the brand'. This shows that these respondents in Cluster Five score, on average, lower in these questions than all other respondents. On the other hand, the top CM1 scores of this cluster include four features that form a top 'shoulder' and include SBC2 which is part of the 'Self-Brand Congruency' construct, INV4 (Brand Involvement) and UI2 and UI3 which are part of the 'Usage Intensity' construct. Usage intensity questions aimed at gaining an idea how often consumers 'used' the Facebook brand page. For example, UI3 stated 'I regularly use the Facebook brand page' (rated from 1 to 7). This indicates that although these respondents may value interactions with the brand through Facebook less than other clusters, they may engage with those brands they feel that are congruent to their own personality and they so at a higher usage intensity than other clusters. This being the case, we label this cluster as 'active participators' since members of this group identify themselves and their self-concept with the brand and regularly visit and use the brand page, which is reflected in the high result for how often the brand uploads new content on its Facebook page (i.e. to facilitate regular interaction and revisits to the brand page), how these consumers visit once a week or more and the high percentage of consumers who are signed in all of the time into Facebook. Therefore, regular posting of content is of greater importance to this segment to the sampled population to than other respondents.

From this CM1 score analysis, we see varying clusters of consumers forming in terms of their online behaviours and specific characteristics. Whilst Cluster One [brand functionalists] respondents are more 'involved' with the brand where they choose to interact with through social media, Cluster Two [passive socializers] respondents are the exact opposite and place more value on the social experience online. Furthermore, Cluster One is the least likely out of the whole sample to experience a state of 'flow' owing to the need for convenience and access to utilitarian oriented content, whilst for Cluster Three [immersers], flow was the highest scoring CM1 score.

We now take one step further in investigating the clusters found in this study by mathematically modelling cluster-specific customer engagement models. That is, using a data-driven symbolic regression analysis, we aim to find predictive models for online Customer Engagement for each cluster separately. In doing so, we provide a basis for future targeted marketing strategies with the objectives of driving higher levels of customer engagement with the brand. Using all other variables in the study, models to predict Customer Engagement are found that are specific to each cluster which go above and beyond the description and explanation of clusters using basic information.

5.4.4 Symbolic Regression Analysis: Cluster-Specific Behavioural Model Building

In this section, we provide an identification of models for Customer Engagement for each of the clusters. As stated, using and interpreting the findings of a clustering or segmentation analysis remains a challenge in practical settings. As we have highlighted, online customer engagement has been of considerable interest to marketing scholars and practitioners alike in recent years. This provides the motivation for cluster-specific model building in order to predict customer engagement in the social media environment. After segmenting consumers into distinct segments, the next logical step is to prepare targeted marketing strategies. However, in order to achieve this, the marketer needs to thoroughly understand their consumers. Whereas the objective of using the CM1 score was to describe the clusters, here we aim to find predictive models of online customer engagement with brands to provide greater insight into each of the clusters. These predictive models could subsequently be used for guiding targeted marketing strategies.

As the theoretical construct of Customer Engagement contained five items, the average of these items for each respondent has been taken to create one variable 'AVENG' to be modelled by our method. As explained, Eureqa builds a Pareto optimality curve to plot the trade-off between complexity and the error value of each model it finds in the data. This curve assists the user in selecting the best model found, considering the appropriate level of complexity as well as accuracy depending on the selected error metric. Furthermore, as we explained about Eureqa's 'building blocks', we restricted the search to only use models that involve the mathematical operations of addition, subtraction, multiplication of variables and the introduction of an integer constant. These integer constants can then appear as a coefficient multiplying a variable or as an additive term somewhere in a formula. The guiding function used for the search is the Correlation Coefficient, this means that the program's task is to find a model of the Average Engagement (labelled AVENG) that highly correlates with the observed values, rather than, for example, minimizing actual error. Here we report the best simple linear models that do not include or integer constants found by Eureqa and provide some initial insights. The meaning of results for each cluster is explained further in the next section.

For all clusters, the best (in terms of error metric) simple linear models that do not include any integers constants are shown in Table 5.2. As can be seen, varying qualities of models are found across the five clusters all with considerably low complexity levels. Included in the table are the correlation coefficients for each of the models as provided by Eureqa.

Table 5.2 Best 'simple' linear models for AVENG—for each cluster, the Pearson correlation coefficient (*Corr. Coef.*) and complexity value (*Compl.* as defined by Eureqa default values) are shown as well as the best model found by Eureqa

	Corr. Coef.	*Compl.*	Symbolic regression model
Cluster 1: Brand rationalists	0.59	3	AVENG = UI3 + SOC2
Cluster 2: Passive socializers	0.64	5	AVENG = UI1 + CCV5 + LO1
Cluster 3: Immersers	0.99	7	AVENG = UI3 + BR1 + RBV2 − FUV1
Cluster 4: Hedonic sharers	0.98	5	AVENG = SK2 + SK5 + FUV3
Cluster 5: Active participators	0.92	5	AVENG = CCV5 + LO1 + ON3

Specifically, from these simple models we can gather information about what type of variables are important in each cluster to predict online customer engagement. Starting with Cluster One—brand rationalists, although this model does not have a perfect fit and not a very high correlation coefficient (0.59), it provides us with two variables that Eureqa has found to be in a model which is correlated to AVENG. UI3 is part of the Usage Intensity construct and SOC2 as part of the Social Value construct. This shows us that these two variables correlate with customer engagement in our largest cluster, Cluster One. Having the relationship of addition between these two variables also indicates that these two activities need to be happening together to predict AVENG and drive higher values of customer engagement. Therefore, as the consumer uses the Facebook page more intensely, they are likely to derive higher social value from the experience which combinedly explain the level of customer engagement with the brand in this cluster of 'brand rationalists'.

For Cluster Two—passive socializers, a model was found with a slightly higher correlation coefficient (0.64) and complexity level 5. The variables in this model are all found to be positively correlated with AVENG and they are as follows: UI1, CCV5 and LO1. Like Cluster One, Usage Intensity is again part of a linear model which is correlated to AVENG. This could be expected as it is likely that those customers who use the brand page more intensely are more likely to be engaged with the brand online. The other two variables belong to the theoretical constructs of 'Co-Creation Value' and 'Loyalty', respectively. As with Cluster One that means these three variables collectively contribute to AVENG in this cluster.

Cluster Three, immersers, has a model for AVENG with the highest correlation coefficient (0.99). There are four variables, three of which Eureqa found to positively contribute to a linear model that correlates with AVENG; however, one variable, FUV1 (Functional Value) is found to be contributing in an inverse sense to the model of customer engagement. A possible interpretation of this is that consumers who are part of this cluster perhaps do not like 'simple', useful and functional information on the Facebook page and do not use the brand page to search this kind of information. Again, Usage Intensity is found in this cluster together two variables relating to interaction and relationship with the brand: BR1 (Brand Relationship Value) and RBV2 (Relationship Building Value). This means that, rather than visiting and using the Facebook brand page for functional reasons, these customers prefer to have interactions with the brand and feel like they are creating a brand relationship.

The AVENG model for Cluster Four, Hedonic sharers, also has a high correlation coefficient (0.98) and is also quite simple. Three variables are found to positively correlate with customer engagement: SK2 and SK5 which belong to the Subjective Knowledge construct and FUV3 (Functional Value). This is the only cluster for which subjective knowledge is found to predict AVENG by Eureqa. These variables are about how 'knowledgeable' and confident the respondent feels about their skill and expertise in using Facebook and the brand page. It is logical to expect that when a consumer feels more confident about how to use a certain type of technology to engage with a brand, the more they will display behaviours towards that brand. However, considering these variables were not included in any other cluster, it shows that perhaps, it matters more for these respondents than others. Furthermore, conversely to Cluster Three, a variable of the Functional Value construct is found to be positively correlated with AVENG for Cluster Four. That means that these respondents do like to find useful and functional information on the Facebook brand page and that they are more likely to engage with the brand online if this is available to them.

Finally, a simple, positive linear model for Cluster Five, active participators, was found with a correlation coefficient of 0.92. The three variables are CCV5 (Co-Creation Value), LO1 (Loyalty) and ON3 (Online Loyalty Behaviours). Considering that two variables relating to loyalty behaviours are found in a predictive model for customer engagement means that certain customers may need to be loyal customers prior to them engaging with the brand through social media. That is, once consumers have developed a brand relationship in the offline environment, they then continue to pursue this relationship in the online environment by interacting with the brand via social media. Furthermore, the co-creation value items all asked respondents how they feel the brand tries to create value for them in using the online social media platform. Naturally, if the brand provides an effective process in enabling co-creative interactions, a customer is more likely to engage with that brand online. The implications of these findings for marketing practitioners will be discussed in the following section.

5.5 Discussion

In this paper, we apply a novel and innovative methodology to the objective of clustering consumers in terms of online behaviours towards brands. More specifically, we examine a seemingly homogeneous sample obtained through survey research of customer's behaviours towards brands through a social media platform. We identify the existence of heterogeneity in the data amongst customers and the need to cluster customers in terms of their behaviours and attitudes rather than their demographic descriptive information or by the 'type' of brand they like and follow online. Through this study, we advance several contributions, both to literature and practice. In this section, we outline these contributions as well as recognize the limitations of this study, provide a guide for future research and present final conclusions.

The principal contribution in this paper is to propose a viable alternative to existing market segmentation and clustering methodologies. In doing so, we provide an ability to better describe and understand resulting clusters for the purposes of informing strategy formulation for facilitating customer engagement with specific market segments which enable insights for guiding target marketing strategies. Specifically, in this study, we have taken a data-driven approach with its roots in the natural sciences and applied it in this social scientific context. We show that the novel MST-kNN method is useful for finding clusters of consumers that are similar in their behavioural profile, rather than their demographic information. Here, we elaborate on our findings further and in particular, articulate what the findings for each cluster means for marketing practitioners.

Our findings entice marketing and brand managers to pursue effective segmentation and targeting strategies by using their behavioural and psychographic profiles rather than only relying on their more basic or demographic behaviours. Furthermore, the consumer behavioural models found by our data-driven approach go a step further in describing and interpreting the clustering results and provide a guideline for practitioners targeted marketing strategies. Starting with Cluster One (i.e. the *'brand rationalists'*) we have observed consumers are already involved with the brand prior to online interactions. This information would not have been available if the segmentation process had used demographic variables only. Furthermore, from the symbolic regression analysis, we have seen that social interaction is important for predicting online engagement with the brand for these consumers. What this means for marketers is that they need to provide a social online atmosphere by, for example, allowing their customers to comment, like and share with the brand, and more importantly for this cluster, with each other on the social media brand page. Eliciting conversations between different users on the brand page and the encouragement of active participation in conversations will generate higher levels of customer engagement for Cluster One consumers, the ‘brand rationalists’. Considering the characteristics of this cluster, targeting those consumers who are already involved with the brand would yield more successful outcomes of their online customer engagement strategies. This can be done by, for example, targeting those users who have followed the brand online for a longer period of time or those consumers in existing databases of the brand and then providing supporting processes on the social media brand page to enable socialization.

Moving to Cluster Two (i.e. the ‘passive socializers’), conversely to the ‘brand rationalists’ (in Cluster One), consumers from this group are more likely to derive ‘Social Value’ in engaging with a branded social media page as these variables were part of the highest CM1 scores. Furthermore, they are also less involved with the brand than other clusters. The results of the symbolic regression modelling give us information to guide targeted strategies for these consumers. For instance, what drives customer engagement with brands in the social media environment in this cluster is when those customers feel like they are allowed to be involved in the provision of services to them and that they can customize the online experience they want (as the Co-Creation construct appears in the average Customer Engagement model). This insight, together with the finding that when customers feel like recommending the brand to others as well as higher usage rates, drives higher levels of engagement from these consumers. For marketing and brand managers, this means that they need to provide customers an opportunity and supporting processes to co-create with the brand page to receive the consumption experience they want. For example, supporting processes include responding to questions about the product or service via the social media page, requesting feedback or input from customers, and allowing mechanisms for customization of the social media experience will all lead to higher levels of customer engagement with brands from this cluster.

Cluster Three (i.e. the ‘Immersers’) was found to be the cluster that is more likely to experience a state of flow when using a branded social media page than other clusters. There could be various reasons for this such as that they use the social media platform from a larger screen such as a computer or tablet (50% of them report that they access Facebook from a desktop or laptop computer). Other reasons could be that they spend more time on the branded social media page or that they enjoy it more than others. Together with the outcomes from the symbolic regression analysis, these findings indicate that the online atmosphere delivered via the social media platform needs to facilitate this type of online experience. In doing so, these customers are likely to ‘lose track of time’ during their usage when their skill and challenge presented to them are in balance, and the more they experience a flow state, the more likely high levels of engagement will occur. For the brand to receive higher levels of engagement on the social media platform from these consumers, online material needs to be existent for customers to enable a flow state to occur whilst customers are using the Facebook page. Furthermore, 'Relationship Building Value' and 'Brand Interaction Value' are also found to be predictive factors of online customer engagement in this cluster, which are important objectives marketers can take into consideration when targeting this cluster of consumers. For instance, this means actively building rapport and a relationship (e.g. special offers, useful value adding content, member programs and benefits) with consumers through the online social media page is likely to lead to higher levels of online customer engagement with the brand for this cluster as well as actively interacting with brand-related communication (e.g. video showcasing the brand) and encouraging conversations and dialogue specifically relating to the brand via the social media page.

Next is Cluster Four: the 'hedonic sharers'. This cluster is the only cluster in which their knowledge of in using social media technology, and their confidence on it, affects their customer engagement levels with brands. This, together with functional value (how functional the content of the Facebook page is) predicts and leads to higher levels of customer engagement in this cluster. What this means for the brand is that perhaps they need to educate those consumers in this cluster on the usage of new technologies. This may not only apply to the Facebook platform, but other new technological advances in customer brand interactions. A great deal of research has been conducted in this field using, for example, 'Technology Acceptance Models' to examine consumers' 'readiness' and knowledge to use new technology, see, for instance [28, 40]. As such, this type of research can aid those marketers who are seeking to better serve consumers who are not yet knowledgeable or confident in using new technology. Mechanisms that allow these consumers to become more confident, which might be needed in the case of Cluster Four, will lead to higher levels of online customer engagement with the brand for them. Furthermore, the fact that all four items of Hedonic Value in Cluster Four were in the top CM1 scores means that these respondents derive, on average, greater hedonic (fun, entertaining) value from using a branded Facebook page. If brands continue to provide these respondents with fun and entertaining content, together with a useful experience, combined with education mechanism for those consumers who need it on using new technologies, they can expect higher levels of customer engagement with brands from cluster four consumers.

Finally, Cluster Five (i.e. 'active participators') consisted of respondents already involved with the brand who responded, on average, higher to a question relating to their Self-Brand Congruency. This means that these customers are more likely to feel that their personality, and the 'personality' that the brand they like online portrays is congruent. From the symbolic regression model we find that, as with Cluster Two, Co-Creation Value is a driver of customer engagement for this cluster. Also predicting customer engagement for this cluster are a loyalty item and an online loyalty behaviour item. This means that for the consumers in this cluster to engage with a brand online, they may need to be loyal customers first. Commonly in customer engagement research, customer engagement with a brand is modelled as a driver of loyalty, rather than loyalty to engagement [11, 25]. However, for this small cluster this relationship may actually occur from loyalty to engagement which is an issue that should be taken into consideration by marketers and could be explored in future research studies.

5.5.1 Limitations and Future Research

To clarify our perspective on the findings and as a possible guide for future research in this field we start by outlining some of the known limitations of our study. First, the relatively small student sample for this study poses some limitations. Even though it has been contended that a student sample is deemed appropriate for online and social media studies [10], future research should consider using a larger and more varied sample in order to improve generalizability. Second, this data was collected using a paper-and-pen survey in an offline setting. To further advance research in this field, 'real-time' behavioural data could be used as well as online surveys presented to a consumer as they are engaging in behaviours towards, and interacting with, brands online.

Third, in the analysis of the symbolic regression results, we have presented only those linear models that were both present in the Pareto frontier and have the maximum number of variables (thus they are the most fitting linear models). This is a self-imposed limitation; however, these models were competing with much more complex non-linear models that perhaps tried to 'over-fit' the data rather than predict the trend. An advantage of selecting these linear models stems from the ease of interpretation as they could easily be converted into real advice to marketing practitioners. In large-scale studies however, some more complex models could be used to analyse each of the clusters as they would provide the researcher with further information and understanding of the drivers for customer engagement in that cluster.

Fourth, we have based our study using consumer-reported data with brands only through one online platform, Facebook. Whilst this one is indeed an omnipresent social media platform, we could also argue that we may have not gathered all possible online interactions consumers could have with brands as other social media platforms allow for close interactions between brands and consumers. Consumer's brand interactions on other platforms such as Twitter, Foursquare or Pinterest would need to be investigated in the future to gather consumer insights on a broader spectrum as they may lead to other personalization strategies. This would also approximate what marketing and brand managers attempt to do in real-life more closely as practitioners need to manage their relationships with consumers across all online and offline communication platforms.

Other areas of future research include the use of community detection algorithms to address segmentation problems such as the one in this study. We have recently shown the application of a community detection algorithm in an analysis of consumer behaviour in the charitable and non-for-profit sector where we compared it to clustering methods [33]. Amongst the currently existing community detection algorithms, some could provide partitioning results of the set of consumers in groups ('communities') that in turn could be matched against and compared to the five clusters described in this study. Such a study on 'ensemble segmentation' (the use of different partitioning algorithms in one study and followed by an analysis of observed commonalities) could be useful for further characterization of 'core' subgroups of people with even better defined saliency features. In addition, other types of algorithms for community detection allow the possibility of having 'overlaps', allowing fuzzy membership of consumers to more than one community [43] or allowing a 'percentage membership' to clusters in studies of fuzzy clustering algorithms [8]. The use and comparison of such 'fuzzy' methods would in turn give new insights for marketers wishing to target specific clusters or groups. Together with the use of hierarchical clustering methodologies, this would allow companies (which may operate under budget constraints) to identify similarities between clusters and design a relatively smaller set of marketing strategies to target their consumer base.

5.5.2 Final Conclusions

In summary, we have presented a comprehensive process for segmenting, analysing and guiding personalized marketing strategies to target consumers. It is important to remember that truly understanding consumer segments, based on their actual characteristics is what will guide marketers' targeting strategies and what will determine their successfulness in terms of positive outcomes for the brand. In this work, we have shown that segmenting consumers based on their behavioural profile, specifically in this case, their online behaviours towards a brand, and subsequently analysing these segments using more than just demographic information gives more insights and knowledge about consumers than demographics alone. As we have stated, the method used in this study is scalable to very large datasets making it feasible to scale this study to millions of consumers to segment and target brands' customers based on their real-time online behaviours. We have also shown the benefits of analysing clusters using symbolic regression modelling which is a method that is transferrable to many other instances and is extremely flexible with its options.

Acknowledgements We would like to thank Dr. Ahmed Shamsul Arefin for his help in providing the clustering result used in this study. Pablo Moscato acknowledges previous support from the Australian Research Council Future Fellowship FT120100060 and Australian Research Council Discovery Projects DP120102576 and DP140104183.

References

1. M Aljukhadar and S Senecal. Segmenting the online consumer market. *Marketing Intelligence & Planning*, 29(4):421–435, 2011.
2. AhmedShamsul Arefin, Mario Inostroza-Ponta, Luke Mathieson, Regina Berretta, and Pablo Moscato. *Clustering Nodes in Large-Scale Biological Networks Using External Memory Algorithms*, volume 7017 of *Lecture Notes in Computer Science*, book section 36, pages 375–386. Springer Berlin Heidelberg, 2011.
3. B Aviad and G Roy. A decision support method, based on bounded rationality concepts, to reveal feature saliency in clustering problems. *Decision Support Systems*, 54(1):292–303, 2012.
4. Amit Bhatnagar and Sanjoy Ghose. A latent class segmentation analysis of e-shoppers. *Journal of Business Research*, 57(7):758–767, 2004.
5. C Blattberg, Robert, Byung-Do Kim, and A Neslin, Scott. Database management: Analyzing and managing customers, 2008.
6. Petter Bae Brandtzæg, Jan Heim, and Amela Karahasanović. Understanding the new digital divide—a typology of internet users in Europe. *International Journal of Human-Computer Studies*, 69(3):123–138, 2011.
7. Colin Campbell, Carla Ferraro, and Sean Sands. Segmenting consumer reactions to social network marketing. *European Journal of Marketing*, 48(3/4):432–452, 2014.
8. Kit Yan Chan, C. K. Kwong, and B. Q. Hu. Market segmentation and ideal point identification for new product design using fuzzy data compression and fuzzy clustering methods. *Applied Soft Computing*, 12(4):1371–1378, 2012.
9. Anil Chaturvedi, J. Douglas Carroll, Paul E. Green, and John A. Rotondo. A feature-based approach to market segmentation via overlapping k-centroids clustering. *Journal of Marketing Research (JMR)*, 34(3):370–377, 1997.
10. S. C Chu and Y Kim. Determinants of consumer engagement in electronic word of mouth (eWOM) in social networking sites. *International Journal of Advertising*, 30(1):47–75, 2011.
11. Natalie Jane de Vries and Jamie Carlson. Examining the drivers and brand performance implications of customer engagement with brands in the social media environment. *J Brand Manag*, 21(6):495–515, 2014.
12. Natalie Jane de Vries, Jamie Carlson, and Pablo Moscato. A data-driven approach to reverse engineering customer engagement models: Towards functional constructs. *PLoS ONE*, 9(7):e102768, 2014.
13. Wayne S. DeSarbo, Kamel Jedidi, and Indrajit Sinha. Customer value analysis in a heterogeneous market. *Strategic Management Journal*, 22(9):845–857, 2001.
14. Arne Floh, Alexander Zauner, Monika Koller, and Thomas Rusch. Customer segmentation using unobserved heterogeneity in the perceived-value–loyalty–intentions link. *Journal of Business Research*, 67(5):974–982, 2014.
15. Mary Foster, Bettina West, and Anthony Francescucci. Exploring social media user segmentation and online brand profiles. *Journal of Brand Management*, 19(1):4–17, 2011.
16. O Giustolisi and D. A Savic. A symbolic data-driven technique based on evolutionary polynomial regression. *Journal of Hydroinformatics*, 8(3):207–222, 2006.
17. Paul E. Green, Frank J. Carmone, and David P. Wachspress. Consumer segmentation via latent class analysis. *Journal of Consumer Research*, 3(3):170–174, 1976.
18. J. L Gross and J Yellen. *Handbook of Graph Theory*. Discrete Mathematics and its Applications. CRC Press LLC, Boca Raton, Florida, 2004.
19. Chris Hand, Francesca Dall'Olmo Riley, Patricia Harris, Jaywant Singh, and Ruth Rettie. Online grocery shopping: the influence of situational factors. *European Journal of Marketing*, 43(9/10):1205–1219, 2009.

20. Thorsten Hennig-Thurau, Edward C. Malthouse, Christian Friege, Sonja Gensler, Lara Lobschat, Arvind Rangaswamy, and Bernd Skiera. The impact of new media on customer relationships. *Journal of Service Research*, 13(3):311–330, 2010.
21. J. L Herlocker, J. A Konstan, A Borchers, and J Riedly. An algorithmic framework for performing collaborative filtering. In *Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 230–237. ACM New York, 1999.
22. Mario Inostroza-Ponta, Regina Berretta, Alexandre Mendes, and Pablo Moscato. *An automatic graph layout procedure to visualize correlated data*, pages 179–188. Springer, 2006.
23. Mario Inostroza-Ponta, Regina Berretta, and Pablo Moscato. QAPgrid: A two level QAP-based approach for large-scale data analysis and visualization. *PLOS One*, 6(1):e14468, 2011.
24. Mario Inostroza-Ponta, Alexandre Mendes, Regina Berretta, and Pablo Moscato. *An integrated QAP-based approach to visualize patterns of gene expression similarity*, pages 156–167. Springer, 2007.
25. Benedikt Jahn and Werner Kunz. How to transform consumers into fans of your brand. *Journal of Service Management*, 23(2):344–361, 2012.
26. Anil K. Jain. Data clustering: 50 years beyond k-means. *Pattern Recognition Letters*, 31(8):651–666, 2010.
27. T. D Klastorin. Assessing cluster analysis results. *Journal of Marketing Research (JMR)*, 20(1):92–98, 1983.
28. Paul Legris, John Ingham, and Pierre Collerette. Why do people use information technology? a critical review of the technology acceptance model. *Information & Management*, 40(3):191–204, 2003.
29. E. C. Malthouse, M. Haenlein, B. Skiera, E. Wege, and M. Zhang. Managing customer relationships in the social media era: Introducing the social CRM house. *Journal of Interactive Marketing*, 27(4):270–280, 2013.
30. R Mancha, M. T Leung, J Clark, and M Sun. Finite mixture partial least squares for segmentation and behavioral characterization of auction bidders. *Decision Support Systems*, 57(1):200–211, 2014.
31. J Marsden, D Budden, H Craig, and P Moscato. Language individuation and marker words: Shakespeare and his Maxwell's demon. *PLOS One*, 8(6):1–12, 2013.
32. Heloisa Helena Milioli, Renato Vimieiro, Carlos Riveros, Inna Tishchenko, Regina Berretta, and Pablo Moscato. The discovery of novel biomarkers improves breast cancer intrinsic subtype prediction and reconciles the labels in the METABRIC data set. *PLOS ONE*, 10:e0129711, 2015.
33. Leila M. Naeni, Natalie Jane de Vries, Rodrigo Reis, Ahmed Shamsul Arefin, Regina Berretta, and Pablo Moscato. Identifying communities of trust and confidence in the charity and not-for-profit sector: a memetic algorithm approach. In *The 7th IEEE International Conference on Social Computing and Networking (Socialcom)*. IEEE, 2014.
34. A O'Cass and J Carlson. Examining the effects of website-induced flow in professional sporting team websites. *Internet Research*, 20(2):115–134, 2010.
35. H Ouwersloot and G Odekerken-Schröder. Who's who in brand communities – and why? *European Journal of Marketing*, 42(5/6):571–585, 2008.
36. K. Y Peters, A. M Chen, B. O Kaplan, and K Pauwels. Social media metrics—a framework and guidelines for managing social media. *Journal of Interactive Marketing*, 27(4):281–298, 2013.
37. M Schmidt and H Lipson. Eureqa, 2013.
38. G. F Smits and M Kotanchek. *Pareto-front exploitation in symbolic regression*, book section 17, pages 283–299. Springer, US, 2005.
39. Jenny van Doorn, Katherine N. Lemon, Vikas Mittal, Stephan Nass, Doreén Pick, Peter Pirner, and Peter C. Verhoef. Customer engagement behavior: Theoretical foundations and research directions. *Journal of Service Research*, 13(3):253–266, 2010.

40. Viswanath Venkatesh and Fred D. Davis. A theoretical extension of the technology acceptance model: Four longitudinal field studies. *Management Science*, 46(2):186–204, 2000.
41. Elaine Wallace, Isabel Buil, Leslie de Chernatony, and Michael Hogan. Who "likes" you ... and why? a typology of Facebook fans. *Journal of Advertising Research*, 54(1):92–109, 2014.
42. Michel Wedel and Wagner A. Kamakura. *Market Segmentation: Conceptual and Methodological Foundations*. International Series in Quantitative Marketing. Kluwer Academic Publishers, Norwell, Massachusetts, USA, 2 edition, 2000.
43. Jierui Xie, Stephen Kelley, and Boleslaw K. Szymanski. Overlapping community detection in networks: The state-of-the-art and comparative study. *ACM Comput. Surv.*, 45(4):1–35, 2013.

Chapter 6
Frequent Itemset Mining

Massimo Cafaro and Marco Pulimeno

Abstract We present a survey of the most important algorithms that have been proposed in the context of the frequent itemset mining. We start with an introduction and overview of basic sequential algorithms, and then discuss and compare different parallel approaches based on shared-memory, message-passing, map-reduce, and the use of GPU accelerators. Even though our survey certainly is not exhaustive, it covers essential reference material, since we believe that an attempt to cover everything will instead fail to convey any useful information to the interested readers. Our hope is that this work will help interested researchers and practitioners, in particular those coming from a business-oriented background, quickly enabling them to develop their understanding of an area likely to play an ever more significant role in coming years.

Keywords Accuracy · Association rule · Association rule mining · Customer churn prediction · FP-array · FP-tree · Frequent itemset mining · Market basket analysis · Transaction

6.1 Introduction

Frequent itemset mining (FIM) was first introduced in the context of market-basket analysis as a first step towards the construction of association rules, i.e., rules that capture a relationship among sets of products sold to customers [4]. An association rule takes the form of an implication $A \implies B$, where A and B are sets of products. As an example in market basket analysis, finding that the dataset reflects the rule $\{\text{bread, butter}\} \implies \{\text{milk}\}$ means that a customer who buys *bread* and *butter* is highly likely to also buy *milk* [2, 3, 38].

M. Cafaro (✉) · M. Pulimeno
University of Salento, Lecce, Italy
e-mail: massimo.cafaro@unisalento.it; marco.pulimeno@unisalento.it

P. Moscato, N. J. de Vries (eds.), *Business and Consumer Analytics: New Ideas*,
https://doi.org/10.1007/978-3-030-06222-4_6

This chapter is organized as follows. In Sect. 6.2, we provide interested readers with an overview of some applications of FIM and an intuitive description of the main concepts and how they apply to different contexts. A formal definition of frequent itemsets and related concepts is given in Sect. 6.3. Then, we recall in Sect. 6.4 the most important sequential algorithms that have been designed to tackle the association pattern mining problem. This provides the foundation to better understand the parallel algorithms surveyed in Sect. 6.5. In particular, we discuss shared-memory, message-passing, map-reduce, and accelerator based parallel algorithms. In Sect. 6.6 we provide information regarding selected, publicly available implementation repositories, in order to allow interested readers to start up quickly. We draw our conclusions in Sect. 6.7.

6.2 FIM Applications

In this section, we discuss a few possible applications of FIM, in order to provide interested researchers and practitioners (in particular, those coming from a business-oriented background) with a better understanding of why these algorithms are relevant to them and how they can be used.

Before delving into the details, it is better clarifying that while sequential algorithms are perfectly fine for dealing with small sized datasets, to be processed on traditional desktop workstations or even laptops, they cannot cope with the dramatic exponential growth of datasets which nowadays characterizes many business-oriented contexts (e.g., consumer analytics).

On the other hand, parallel algorithms (which are able to simultaneously take advantage of multiple processors/cores of execution) are able to solve compute-intensive problems in a fraction of the time required by a sequential algorithm. An infeasible problem (from a time required perspective) suddenly becomes feasible, the time required for design is significantly reduced, etc.

Reducing the time required to obtain a solution to a problem of interest of fixed size is the main reason that lead people adopting parallelism. However, a slightly different use of parallelism has also become mainstream: solving a larger and/or data intensive problem in the same amount of time required to solve a much smaller instance of the problem. For instance, one may be interested to solve a problem with higher accuracy and/or characterized by a huge dataset and is willing to wait at most the same amount of time required to solve a much smaller instance of the problem.

In both cases, parallelism allows to gain a distinct, competitive advantage (think about reducing the so-called time to market, etc.), and companies are therefore continuously evolving their approaches to include parallel computing as one of the key factors required to remain competitive.

Having discussed the key role of parallelism, we now present a few possible applications of FIM.

6.2.1 Market-Basket Analysis

Retailers collect data about costumer purchases in the form of transaction datasets, where each transaction represents a shopping cart, i.e., the set of items bought together by a customer during the same visit to the store. In this context, a frequent itemset is a set of items that appear together in many transactions. Finding the frequent itemsets allows to determine the sets of items frequently bought together and, as a subsequent step, to compute interesting and unexpected association rules.

The knowledge of association rules, in particular unexpected ones, is of great utility to decision makers in order to determine the right selling strategies, such as price promotions or better shelf placements of products. A famous and unexpected association rule, often mentioned in the context of market basket analysis, is the $\{\text{diapers}\} \implies \{\text{beer}\}$, which states that diapers and beer are often bought together.

A possible explanation is that who buys diapers is likely to have children and to stay at home instead of going out for a drink, so he also buys beer. Knowing that, a manager could decide to locate the beer next to the diapers to facilitate the purchase, or he could promote the diapers but increase the beer price. It is unlikely that a customer attracted to the store by the good price of diapers goes to another supermarket for buying the beer, even though its price is higher.

6.2.2 Document Analysis

Over time, since the first introduction in market basket analysis, FIM has been employed in other contexts and also as an intermediate step for other data mining tasks, such as classification and clustering.

Depending on the specific application context, the meaning assigned to transactions, frequent itemsets and association rules will change to reflect the aims of the analysis.

In document sets analysis [9, 10, 16], for example, one can consider each document as a transaction and the words in the document as items. In this case, frequent itemsets are sets of words common to many documents. This kind of analysis can be useful for clustering, since documents containing many words in common are likely to deal with the same topics.

Indeed, FIM allows to better cope with some peculiar features of document clustering, such as high dimensionality, very large datasets and the need for understandable cluster labels.

6.2.3 Web Log Analysis

Web servers maintain log data about user accesses to hosted web pages. This data can be processed by data mining techniques in order to extract useful information on users' behaviour. In this context, FIM can be used, for example, to find the set of pages more likely to be accessed together during the same session by a user [20, 30]. This knowledge can help web administrators improving the web site structure, or web organizations to design better promotions tailored to users' habits.

6.2.4 Recommendation Systems

Frequent itemset analysis has also been successfully applied to the design of recommender systems [25, 29]. A recommendation system tries to predict the preferences of a user, such as the movies he is likely to see or the products he is likely to buy, based on his past behaviour and the behaviour of other similar users (collaborative filtering). In this context, frequent itemsets can be employed both to cluster users in segments with same taste and to determine good candidate products to be recommended.

6.2.5 Anomaly and Intrusion Detection

Some anomaly detection systems are based on data mining techniques, which allow to analyse system audit data about users and programs behaviour in order to learn how the system behaves in normal conditions and, consequently, to be able to detect deviations from usual behaviours. FIM and association rules can be employed in this context, for example, to learn a model of normal users and programs behaviour [22, 23]. Transactions, in these cases, take the form of audit data, for example, the list of shell commands issued by a user in a shell session.

6.2.6 Bioinformatics

FIM has also been applied to medical and biological data [7]. For example, one can consider the blood markers, genetics data, and disease characteristics of a patient as a *basket* and mine a group of patients, looking for the most frequent set of markers which are associated with a particular disease. This information could be useful for better disease diagnosis, or to improve the knowledge about what causes the disease.

In general, biological datasets differ from traditional market datasets, both in structure and dimensions. Gene-expression datasets, for example, are very high dimensional and can contain a hundred of thousands columns in a single *transaction*. That requires specialized algorithms and long pattern mining techniques, such as closed and maximal frequent itemset mining. We will discuss these methods and formally define closed and maximal frequent itemsets in the following sections.

6.3 Preliminary Definitions

Frequent itemset mining requires processing a dataset which is a *database* $\mathcal{T}$ of *transactions* $T_1, \ldots, T_n$. Each transaction T_i contains items drawn from a universe set of items $U = \{i_1, i_2, \ldots, i_d\}$. A transaction is represented by a unique identifier T_i called *transaction ID* (or TID), and refers to a set of items. A set consisting of transaction IDs is called a *tidset*. An *itemset* is a set of items, and a k-itemset is a set of items of cardinality k.

The previous database representation is called *horizontal*, i.e., a horizontal representation of the database consists of pairs $(T_i, I(T_i))$ in which T_i is a transaction and $I(T_i)$ is the itemset containing all of the items in T_i. Some algorithms require another representation of the database, called *vertical*. In this case, the database consists of pairs $(i, T(i))$, in which i is an item, and $T(i)$ is a tidset containing all of the transactions in which the item i is present.

Definition 6.1 Given an itemset I, $sup(I) = |\{T_i : I \subset T_i, T_i \in \mathcal{T}\}|$ is the number of transactions in $\mathcal{T}$ that contain I as a subset. The quantity $sup(I)$ is called the support of itemset I.

Alternatively, $sup(I)$ can be defined as the fraction of transactions that contain I as a subset.

In FIM, we are interested in discovering highly correlated items. Of course, those items will naturally occur together in the transactions and therefore will have high support. As a consequence, it makes sense to fix an application dependent threshold that we shall call *minimum support* and denote by $minsup$. An itemset I is said to be *frequent* in $\mathcal{T}$ if $sup(I) \geq minsup$.

It is worth noting here that $minsup$ is inversely proportional to the number of frequent itemsets: a lower minimum support yields a large number of frequent items and, vice-versa, a higher minimum support yields a few or even no frequent itemsets at all. We are now ready to formally state the problem of FIM.

Definition 6.2 Given a database $\mathcal{T}$ of transactions, the problem of frequent itemset mining entails determining all of the itemsets I such that I occurs in at least $minsup$ of the transactions of $\mathcal{T}$.

Table 6.1 Example of a horizontal database

Transaction ID	Itemset	Binary representation
1	{milk, bread}	1100
2	{bread, butter}	0110
3	{beer}	0001
4	{milk, bread, butter}	1110
5	{bread, butter}	0110

Table 6.2 Example of a vertical database

Item	Tidset	Binary representation
Milk	$\{1, 4\}$	10010
Bread	$\{1, 2, 4, 5\}$	11011
Butter	$\{2, 4, 5\}$	01011
Beer	$\{3\}$	00100

As an example, borrowed from the supermarket domain, consider the universe set of items $U = \{\text{milk, bread, butter, beer}\}$ and a small database containing the items whose horizontal and vertical representations are shown, respectively, in Tables 6.1 and 6.2. Alternatively, the set of items related to a transaction in the horizontal representation can also be stored in binary form, in which an item in U is considered as a binary variable whose value is one if the item belongs to the itemset corresponding to the transaction and zero otherwise. For the vertical representation, the binary form refers to the tidsets, therefore a transaction is considered as a binary variable whose value is one if the corresponding item in U belongs to the transaction and zero otherwise.

An association rule could be $\{\text{milk, bread}\} \Rightarrow \{\text{butter}\}$, meaning that if milk and bread are bought, customers also buy butter. The itemset $I = \{\text{milk, bread}\}$ has support $sup(I) = 2$ (alternatively, its support is $2/5 = 0.4$ since the itemset occurs in 40% of all of the transactions, i.e., 2 out of 5 transactions).

If an itemset I occurs in a transaction, then all of its subsets will also be contained in the transaction. This simple observation leads to the following property, referred to as the *support monotonicity property*.

Property 6.1 The support of any subset J of an itemset I is always at least equal to the support of I: $sup(J) \geq sup(I) \quad \forall J \subseteq I$.

An immediate consequence of monotonicity of support is that every subset of a frequent itemset is also frequent. This property is referred to as *downward closure*.

Property 6.2 Every subset of a frequent itemset is also frequent.

The downward closure allows to efficiently prune the search space (i.e., it can be used to discard itemsets which are not frequent, owing to the fact that if an itemset I is not frequent, then any superset $K \supseteq I$ cannot be frequent). Moreover, it can also be used in those cases in which we are interested in determining a specific class of frequent itemsets, namely the *maximal* frequent itemsets.

Definition 6.3 Given a minimum support *minsup*, a frequent itemset I is maximal if $sup(I) \geq minsup$ (i.e., I is frequent), and no superset of I is frequent.

Besides maximal frequent itemsets, we may be interested in some applications in determining the so-called *closed* frequent itemsets.

Definition 6.4 Given a minimum support *minsup*, a frequent itemset I is closed if $sup(I) \geq minsup$ (i.e., I is frequent), and none of its supersets have exactly the same support $sup(I)$.

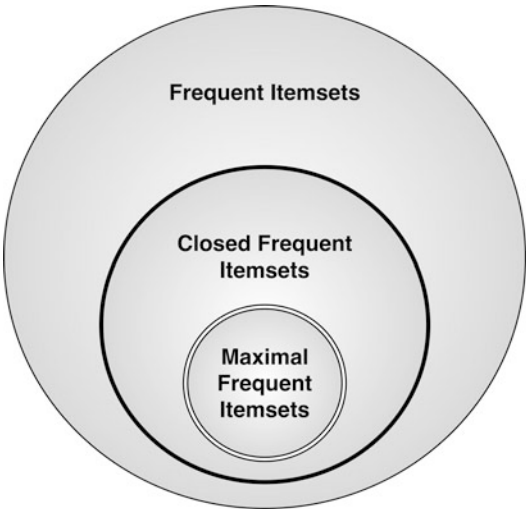


Fig. 6.1 Set relationship among frequent, closed and maximal itemsets

Figure 6.1 shows the relationship among frequent itemsets, closed frequent itemsets and maximal frequent itemsets.

There exists a simple graphical representation of itemsets, called a *lattice*, shown in Fig. 6.2. A lattice is a graph $G = (V, E)$ in which V is the set of vertices, i.e., itemsets, with $|V| = 2^{|U|}$, and E is the set of edges, with an edge connecting a pair of vertices *iff* (if and only if) the corresponding itemsets differ by exactly one item. The lattice is a natural representation of the search space of frequent patterns. Since the number of itemsets in the lattice is exponential (being the power set of the universe U), a brute-force frequent pattern mining algorithm trying to explicitly traverse the lattice cannot be efficient. Pruning the search space by traversing the lattice implicitly is therefore a common feature of all of the frequent pattern mining algorithms. Figure 6.2 refers to the set of transactions in Table 6.3: for each itemset the IDs of the transactions containing it are indicated. Grey nodes in Fig. 6.2 are frequent itemsets, while grey nodes with a marked border are closed frequent itemsets; nodes with a double border are maximal frequent itemsets.

Enumerating the itemsets in the lattice can be done using either breadth-first (BFS) or depth-first (DFS) search on the *prefix tree*, where two itemsets A, B are connected by an edge *iff* A is an immediate subset and prefix of B. In practice, the prefix tree provides an enumeration of the lexicographically ordered string representation of the itemsets. This way, itemsets can be enumerated starting with the empty set, and adding one item at a time. An item used to extend a node to its child in the prefix tree is referred to as a *tree extension*, or simply an extension.

Table 6.3 Example of transaction database

Transaction ID	Itemset
1	{a, c, d}
2	{a, b, e}
3	{c, d, e}
4	{a, b, c, e}
5	{a, b, c, d, e}
6	{c, e}
7	{a, b, c, e}

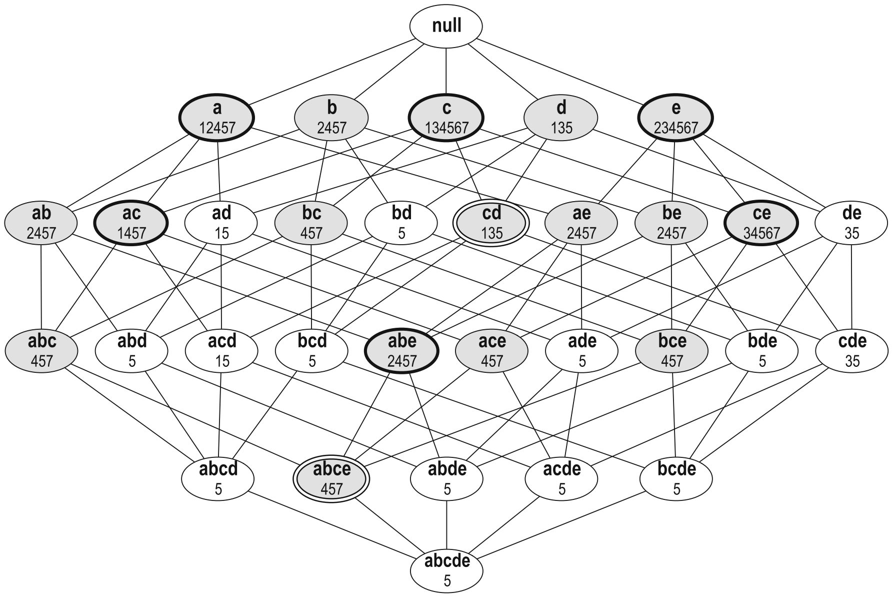


Fig. 6.2 Example of a lattice: grey nodes are frequent itemsets, marked grey nodes are closed frequent itemsets, double bordered grey nodes are maximal frequent itemsets

In order to use frequent itemsets to generate association rules, we must take into account two measures known as *support* and *confidence*.

Definition 6.5 Given an association rule $A \implies B$, the support of the rule is defined as $sup(A \implies B) = \frac{sup(A \cup B)}{N}$, where $N = |\mathcal{T}|$.

Definition 6.6 Given an association rule $A \implies B$, the confidence of the rule is defined as $conf(A \implies B) = \frac{sup(A \cup B)}{sup(A)}$.

For an association rule, the support metric is an indication of how often the rule can be applied to a given input dataset. A rule whose support is very low may occur in the input dataset only by chance. Moreover, low support rules are likely to be, in general, not interesting (from a business oriented perspective, it may not be profitable promoting items rarely bought together by customers).

On the other hand, the confidence metric determines how often items belonging to the itemset B appear in transactions containing the itemset A. Therefore, it measures how reliable the inference made by a rule is. For a given association rule $A \implies B$, a higher confidence value means that it is more likely for B to be present in transactions that contain A. By its definition, the confidence of a rule can also be interpreted as an estimate of the conditional probability of B given A.

The itemsets A and B are, respectively, the *antecedent* and the *consequent* of the rule. In order to generate the most interesting and relevant association rules, a minimum, application dependent, confidence threshold *minconf* is usually defined.

Definition 6.7 Let A and B be two sets of items. Then, the rule $A \implies B$ is an association rule at a minimum support (*minsup*) and minimum confidence (*minconf*), if it satisfies the following two criteria:

1. The support of the itemset $A \cup B$ is at least minsup;
2. The confidence of the rule $A \implies B$ is at least minconf.

The generation of association rules requires two steps. These steps correspond to the two criteria appearing in Definition 6.7, representing, respectively, the support and confidence constraints. In the initial step, all of the frequent itemsets with support greater than or equal to $minsup$ are generated. In the second step, the association rules are generated from the frequent itemsets at the minimum confidence level of $minconf$.

The first step is, in general, computationally intensive and, therefore, it has attracted the attention of researchers, being the most interesting part of the whole process. The second step is, indeed, rather straightforward. We will defer the discussion of sequential FIM algorithms to Sect. 6.4, and the discussion of parallel algorithms to Sect. 6.5. Here, we provide a brief discussion of the second step, without delving into the details.

The output of an algorithm, either sequential or parallel, solving the FIM problem, is a set of frequent itemsets F. For each itemset $I \in F$, we can generate association rules by partitioning I into all of the possible combinations of the sets X and $Y = I \setminus X$, such that $I = X \cup Y$. The confidence of each rule $X \implies Y$ can then be determined, and it can be retained if it satisfies the minimum confidence requirement. Association rules also satisfy a confidence monotonicity property.

Property 6.3 Let X_1, X_2 and I be itemsets such that $X_1 \subset X_2 \subset I$. Then the confidence of $X_2 \implies I \setminus X2$ is at least that of $X_1 \implies I \setminus X_1$: $conf(X_2 \implies I \setminus X2) \geq conf(X_1 \implies I \setminus X_1)$.

Confidence monotonicity follows straight from the definition of confidence and the property of support monotonicity.

6.4 Sequential Algorithms

In this section, we review some of the most important sequential algorithms that have been designed to solve the FIM problem. The presentation is based on [51]. We recall here the following algorithms: *Apriori* [4], *Eclat* [49] and *FP-growth* [19].

Other algorithms for mining maximal itemsets include *Max-Miner* [8], *MAFIA* [11], *Gen-Max* [18] and *IBE* [40]. Relevant algorithms for closed itemsets are *LCM* (Linear time Closed itemset Miner) [45], *A-Close* [35], *CLOSET* [36], *CHARM* [52] and *Titanic* [42].

Finally, we also discuss briefly some recently proposed sequential algorithms for mining disjunctive patterns: *TitanicOR* [46], *Disclosed* [47] and *QCEP*. Among the algorithms in this class, we recall here *BLOSOM* [55], *DEPMiner* [28], *MBD-LLBorder* [14] and *TW* [44].

6.4.1 Apriori

The *Apriori* algorithm is based on a level-wise or breadth-first exploration of the itemset lattice. *Apriori* works by pruning all of the supersets of any infrequent candidate itemset, since, as already discussed, no superset of an infrequent itemset can be frequent. Another optimization regards candidate itemset generation. Indeed, *Apriori* avoids generating candidate itemsets that have infrequent subsets. Moreover, the algorithm computes the support count of all of the valid candidates of size k that it finds in level k when exploring the prefix tree.

The pseudo-code for *Apriori* is shown in Algorithm 1. Let $\mathcal{C}^{(k)}$ be the prefix tree comprising all of the candidate k-itemsets. *Apriori* starts by inserting all of the single items into an initially empty prefix tree, populating $\mathcal{C}^{(1)}$. In the *while* loop, the algorithm determines the support count of the current set of candidates at level k by invoking the COMPUTESUPPORT procedure. Here, k-subsets of each transaction in the database $\mathcal{T}$ are generated, and for each subset *Apriori* increments the support count of the corresponding candidate itemset in $\mathcal{C}^{(k)}$ if it exists. Therefore, the algorithm performs just one database pass per level, and the supports of all of the candidate k-itemsets are incremented during that pass.

Itemsets whose support count is greater than or equal to $minsup$ are added to $\mathcal{F}$; otherwise, they are not frequent and can be discarded and removed from $\mathcal{C}^{(k)}$. Then, $\mathcal{C}^{(k)}$ is used to generate the candidate $(k+1)$-itemsets belonging to the next level $k+1$.

Given two frequent k-itemsets X_a and X_b sharing a common $k-1$ length prefix, that is, given two sibling leaf nodes with a common parent, the EXTENDPREFIX-TREE procedure generates the $(k+1)$-length candidate itemset $X_{ab} = X_a \cup X_b$, which is retained if it has no infrequent subset, otherwise is discarded.

In order to guarantee that in $\mathcal{C}^{(k)}$ all of the leaves are at level k, if a k-itemset X_a cannot be extended, it is pruned from the prefix tree, along with any of its ancestors with no k-itemset extension. The while loop continues until no new candidate itemsets are added to $\mathcal{C}^{(k+1)}$.

6.4.2 Eclat

The *Eclat* (Equivalence CLAss Transformation) algorithm requires a vertical representation of the database, and is based on indexing the database so that fast frequency computations are possible. Therefore, it significantly improves support counting with regard to *Apriori*. Indeed, the level-wise approach of *Apriori* requires generating the subsets of each transaction and checking if these subsets are present in the prefix tree. However, since not all of the generated subsets will be in the prefix tree, this is a computationally expensive step.

Eclat exploits the tidsets for support counting. The underlying reason is that it is possible computing the support of a candidate itemset by intersecting the tidsets of suitably chosen subsets. Let $t(A)$ and $t(B)$ be, respectively, the tidsets corresponding to two frequent itemsets A and B. Then, $t(AB) = t(A) \cap t(B)$, and, for the candidate itemset AB, $sup(AB) = |t(AB)|$.

The algorithm intersects tidsets if the frequent itemsets share a common prefix. *Eclat* explores the prefix tree using depth first search, determining and processing groups of itemsets that share a common prefix (these groups are called *prefix equivalence classes*).

The pseudocode for *Eclat* is shown in Algorithm 2. A variant in which the performances of *Eclat* are improved through a careful shrinking of the size of the intermediate tidsets is called *dEclat*. This algorithm keeps track of the differences in the tidsets as opposed to the full tidsets.

6.4.3 FP-Growth

The *FP-growth* algorithm is also based on the idea of indexing the database so that fast frequency computations are possible. To this end, it uses an augmented prefix tree called FP-tree (frequent pattern tree).

The vertices of the tree are labelled with a single item, and each child vertex represents a different item. Vertices also store the support information related to the itemset comprising all of the items on the path from the root to a given vertex. To build the FP-tree, we start with a tree containing as root the null item $\emptyset$. For each pair $(T_i, X) \in \mathcal{T}$, where $X = I(T_i)$, the itemset X is inserted into the FP-tree, and the count of all of the vertices along the path representing X is incremented. When X shares a common prefix with some previously inserted transaction, X will follow

Algorithm 1: *Apriori* algorithm

Input: $\mathcal{T}$, a database of transactions; U, a universe set of items; $minsup$, the minimum support

Output: the set of frequent items in $\mathcal{T}$

```
1  procedure APRIORI($\mathcal{T}$, $U$, $minsup$)
2      $\mathcal{F} \leftarrow \emptyset$
3      $\mathcal{C}^{(1)} \leftarrow \{\emptyset\}$
4      foreach $i \in U$ do
5          add $i$ as child of $\emptyset$ in $\mathcal{C}^{(1)}$ with $sup(I) \leftarrow 0$
6      end foreach
7      $k \leftarrow 1$
8      while $\mathcal{C}^{(k)} \neq \emptyset$ do
9          COMPUTESUPPORT($\mathcal{C}^{(k)}$, $\mathcal{T}$)
10         foreach $leaf\ X \in \mathcal{C}^{(k)}$ do
11             if $sup(X) \geq minsup$ then
12                 $\mathcal{F} \leftarrow \mathcal{F} \cup \{(X, sup(X)\}$
13             else
14                 remove $X$ from $\mathcal{C}^{(k)}$
15             end if
16         end foreach
17         $\mathcal{C}^{(k+1)} \leftarrow$ EXTENDPREFIXTREE($\mathcal{C}^{(k)}$)
18         $k \leftarrow k + 1$
19     end while
20     return $\mathcal{F}$
21 end
22 procedure COMPUTESUPPORT($\mathcal{L}$, $\mathcal{T}$)
23     foreach $T_i \in \mathcal{T}$ do
24         foreach k-subset $X \subseteq T_i$ do
25             if $X \in \mathcal{L}$ then
26                 $sup(X) \leftarrow sup(X) + 1$
27             end if
28         end foreach
29     end foreach
30 end
31 procedure EXTENDPREFIXTREE($\mathcal{L}$)
32     foreach $leaf\ X_a \in \mathcal{L}$ do
33         foreach $leaf\ X_b \in SIBLING(X_a)$ such that $b > a$ do
34             $X_{ab} \leftarrow X_a \cup X_b$
35             if $X_j \in \mathcal{L}$ then
36                 foreach $X_j \subset X_{ab}$ such that $|X_j| = |X_{ab}|$ - 1 do
37                     add $X_{ab}$ as child of $X_a$ with $sup(X_j) \leftarrow 0$
38                 end foreach
39             end if
40             if no extensions from $X_a$ then
41                 remove $X_a$ and its ancestors with no extensions from $\mathcal{L}$
42             end if
43         end foreach
44     end foreach
45     return $\mathcal{L}$
46 end
```

Algorithm 2: *Eclat* algorithm

```
   Input: a vertical database of transactions; $minsup$, the minimum support
   Output: the collection of frequent itemsets in $\mathcal{F}$
   // Initial call: $\mathcal{F} \leftarrow \emptyset, P \leftarrow \{(i, T(i)) : i \in U, |T(i)| \geq minsup\}$
1  procedure ECLAT($P, minsup, \mathcal{F}$)
2      foreach $(X_a, T(X_a)) \in P$ do
3          $\mathcal{F} \leftarrow \mathcal{F} \cup \{(X_a, sup(X_a))\}$
4          $P_a \leftarrow \emptyset$
5          foreach $(X_b, T(X_b)) \in P,\ with\ X_b > X_a$ do
6              $X_{ab} \leftarrow X_a \cup X_b$
7              $T(X_{ab}) \leftarrow T(X_a) \cap T(X_b)$
8              if $sup(X_{ab}) \geq minsup$ then
9                  $P_a \leftarrow P_a \cup \{(X_{ab}, T(X_{ab}))\}$
10             end if
11         end foreach
12         if $P_a \neq \emptyset$ then
13             ECLAT($P_a, minsup, \mathcal{F}$)
14         end if
15     end foreach
16 end
```

the same path until the common prefix. Regarding the remaining items in X, new vertices are inserted under the common prefix, and their count is initialized to 1. The FP-tree construction is complete when all of the transactions have been processed and inserted.

The usefulness of the FP-tree lies in being a prefix compressed representation of the database $\mathcal{T}$. In order to compact the FP-tree as much as possible, so that the most frequent items are found at the top, near the root, the algorithm reorders the items in decreasing order of support. Starting from the initial database, *FP-growth* determines the support of all of the single items $i \in U$. Then, infrequent items are discarded, and frequent items sorted by decreasing support. Finally, each pair $(T_i, X) \in \mathcal{T}$ is inserted into the FP-tree after reordering X by decreasing item support.

The algorithm now uses the FP-tree as an index in place of the original database. The frequent itemsets are mined from the tree as shown in Algorithm 3. The algorithm takes as input a FP-tree R built from the database $\mathcal{T}$, and the current itemset prefix P, which is initially empty.

FP-growth builds projected FP-trees for each frequent item i in R in increasing order of support. The projection of R on item i is determined by finding all of the occurrences of i in the tree, and for each occurrence, computing the corresponding path from the root to i. The count of each item i on a given path is stored in the variable $cnt(i)$. The path is then inserted into the new projected tree RX, where X is the itemset obtained by extending the prefix P with the item i.

When a path is inserted, the count of each vertex in RX along the given path is incremented by the corresponding path count $cnt(i)$. The item i is omitted from the path, since it now belongs to the prefix. The derived FP-tree is the projection of the itemset X that includes the current prefix extended with item i. The algorithm is then recursively invoked with projected FP-tree RX and the new prefix itemset X as input parameters. The recursion ends when the input FP-tree R is just a single path. In this case, all of the itemsets that are subsets of the path are enumerated, and the support of each such itemset is determined by the least frequent item in it.

Algorithm 3: *FP-growth* algorithm

```
Input: $\mathcal{T}$, a database of transactions; $minsup$, the minimum support
Output: the collection of frequent itemsets in $\mathcal{F}$
// Inital call: $R \leftarrow$ FP-tree($\mathcal{T}$), $P \leftarrow \emptyset, \mathcal{F} \leftarrow \emptyset$
1  procedure FP-GROWTH($R$, $P$, $\mathcal{F}$, $minsup$)
2      Remove infrequent items from $R$
3      if ISPATH($R$) then                    // insert subsets of $R$ into $\mathcal{F}$
4          foreach $Y \subseteq R$ do
5              $X \leftarrow P \cup Y$
6              $sup(X) \leftarrow min_{x \in Y}\{cnt(x)\}$
7              $\mathcal{F} \leftarrow \mathcal{F} \cup \{(X, sup(X))\}$
8          end foreach
9      else // process projected FP-trees for each frequent item $i$
10         foreach $i \in R$ in increasing order of $sup(i)$ do
11             $X \leftarrow P \cup \{i\}$
12             $sup(X) \leftarrow sup(i)$   // sum of $cnt(i)$ for all nodes labelled
               $i$
13             $\mathcal{F} \leftarrow \mathcal{F} \cup \{(X, sup(X))\}$
14             $R_X \leftarrow \emptyset$                    // projected FP-tree for $X$
15         end foreach
16     end if
17     foreach $path \in$ PATHFROMROOT($i$) do
18         $cnt(i) \leftarrow$ count of $i$ in $path$
19         Insert $path$, excluding $i$, into FP-tree $R_X$ with count $cnt(i)$
20     end foreach
21     if $R_X \neq \emptyset$ then
22         FP-GROWTH($R_X$, $X$, $\mathcal{F}$, $minsup$)
23     end if
24 end
```

6.4.4 Disjunctive Patterns

All of the algorithms discussed so far are designed to mine conjunctions of frequent itemsets. However, in some application domains, e.g., mining of biomedical datasets, those algorithms are unsuitable to identify patterns owing to the limited expressiveness of conjunctions of frequent itemsets. For completeness, in this section, we briefly highlight some recent algorithms designed for mining disjunctive patterns. In particular, these algorithms deal with disjunctive minimal generators, disjunctive closed patterns and quasi-CNF emerging patterns. Details of the algorithms presented here can be found in the corresponding references.

6.4.4.1 TitanicOR

TitanicOR has been designed to identify patterns in datasets containing physiological, biochemical or medical record information [46]. The algorithm is an extension of the traditional conjunctive Titanic, and can mine disjunctive patterns in traditional transactional-like datasets.

Zhao et al. [55] were the first to investigate mining disjunctive minimal generators. This task is the counterpart of the traditional conjunctive approach, with the aim of augmenting the expressiveness of minimal generators. The main difference between disjunctive minimal generators and their conjunctive counterpart lies in the way the patterns are interpreted. Indeed, conjunctive minimal generators occur in one sample (i.e., a transaction) only if the sample contains all of the features (i.e., the items) in the set, while a disjunctive minimal generator occurs in a sample if the sample contains at least one feature of the set.

Disjunctive patterns are interesting in those situations where conjunctive interpretation may not be suitable. For instance, it might be of interest for a supermarket manager knowing that the purchase of at least one of the items in an itemset often implies the purchase of another set of items. As another example, a biologist might be interested to know that, even though there exist some proteins whose individual occurrence may not be associated with the development of a disease, their union may totally describe that condition.

TitanicOR [46] deals with minimal generators, a kind of pattern that was introduced as a concise representation of frequent itemsets. Minimal generators are the most general description for a set of samples: the set of all of the possible combinations of samples partitions the set of itemsets in equivalence classes. All of the itemsets falling in a same class occur in the same set of samples. Minimal generators are the smallest elements of such equivalence class.

TitanicOR has been derived from the original conjunctive version of *Titanic* proposed by Stumme et al. [42], and works in a breadth-first manner: it first determines all of the itemsets of size k before determining itemsets of size $k + 1$. The algorithm applies several pruning strategies to greatly reduce the search space.

6.4.4.2 Disclosed

The problem of mining frequent disjunctive closed itemsets was first introduced by Zhao et al. [55] in their algorithm for mining Boolean expressions, *BLOSOM*.

Disclosed is suitable for the characteristics of microarray gene expression datasets, which usually contains many features, but only few samples [47]. Note that this kind of situation is unusual in consumer behaviour datasets, in which there are usually a lot of samples and a few variables; nonetheless, there are instances in which data can be structured this way, and these are hard to deal with (e.g., click-through data creates lots of variables for just a few people).

Disclosed can be thought of as a complementary approach for *TitanicOR*, since it can mine disjunctive closed patterns tackling high dimensional datasets. Disclosed is the first algorithm to mine disjunctive closed patterns by enumerating samples rather than features. Therefore, *Disclosed* is the first algorithm that was designed for mining disjunctive closed patterns in biomedical datasets.

Disclosed is a sample-based enumeration algorithm. It traverses the itemset lattice in a depth-first, top-down manner, starting with the largest possible set of samples, and then exploring its children. The algorithm inserts the set of all of the features into the resulting set, owing to the fact that this is the most trivial closed itemset—by definition of disjunctive closed itemset, the set of all of the attributes always describes the set of all of the samples and vice-versa, and therefore it is closed. Then, the algorithm traverses the search space in order to determine other closed frequent itemsets, pruning candidates violating the constraints.

6.4.4.3 QCEP

The problem of mining disjunctive emerging patterns was introduced by Loekito and Bailey [27]. Disjunctive emerging patterns, or quasi-CNF emerging patterns (QCEP), are sets of features (attributes) that are more frequent among samples from one class to another. Attributes are assumed to be categorical, so that each one belongs to a domain with at least one value. QCEP patterns involve disjunctions among values within an attribute and conjunctions among attributes.

QCEP [48] is the only algorithm that takes into account datasets with class label information to mine quasi-CNF emerging patterns, i.e., those patterns that highlight the differences between the samples in one class and another. As an application, in this book the interested reader will find a chapter dealing with customer churn prediction, in which it is shown how to predict two classes, i.e., how to discriminate between churners versus non-churners.

Quasi-CNF emerging patterns are conjunctions of features; however, the algorithm also allows disjunctions of values within features, but not negation. One can see these patterns as Boolean formulas in the conjunctive normal form, except that negations are not permitted, hence the name quasi-CNF emerging patterns.

The problem of mining QCEP can be broken into smaller subproblems and then mapped onto the classical hypergraph problem of enumerating minimal transversals, also known as hypergraph dualization.

6.5 Parallel Algorithms

The main reason leading to the design and implementation of parallel algorithms is the need to solve compute-intensive problems faster. However, scientists are often interested in a slightly different use of parallel computing: they want to solve a larger problem in the same amount of time or the same problem in the same amount of time but with greater accuracy and numerical precision, perhaps using a much more complex scientific model at the hearth of the problem.

Parallelism is here to stay, owing to the availability of inexpensive parallel computers such as commodity desktop multiprocessors and even multicore laptops. This, and the continuous development of standards for portable parallel programming are driving a substantial growth and demand for parallel algorithms and applications.

Many different parallel algorithms have been designed in order to solve the FIM problem. In this section, we provide a survey of various solutions, both for classic shared-memory and shared-nothing architectures (message-passing based) and for more recent distributed and parallel computing platforms, such as Map–Reduce and GPU (graphical processor unit) accelerators.

6.5.1 Shared-Memory Algorithms

Regarding shared-memory architectures, all of the processors can access a global memory space, since the primary memory is shared among the processors. A global address space is easier to program, with global data structures used efficiently with little modifications. Sharing data among processors/tasks is fast and simple.

However, intrinsic hardware limits prevent scalability of SMP (symmetric multiprocessing) nodes to higher processor counts: adding processors increases the traffic on the shared memory—CPUs bus. Cache coherent systems suffer this problem even more. Indeed, current SMP nodes are equipped with at most a few dozens of processors. An additional problem is the need for proper synchronization: the programmer is responsible for correct synchronization when accessing shared data.

6.5.1.1 Parallel Programming Techniques

Jin and Agrawal [21] present a programming interface based on a reduction object which allows to specify a parallel algorithm based on a common framework. The programming interface allows the implementation of different techniques, including *full replication*, *buffering*, *full locking*, *fixed locking*, *group locking* and *optimized full locking*. This approach reflects the fact that many algorithms, including *Apriori*, require a parallel reduction in which many threads process different transactions. The main issue regarding correctness is the need to avoid race conditions and to properly handle critical sections ensuring mutual exclusion.

As an example, in *Apriori* a data item read must be matched against all of the candidates in order to determine the set of candidates whose count will be incremented. It follows that no partitioning strategy can assign items to threads so that the threads are guaranteed not to update the same element in the reduction object.

Full replication avoids locking, but requires one copy of the reduction object per thread. Therefore, each thread can locally reduce its set of items, and a final *merge* operation is then needed to correctly update the partial results.

Since full replication may incur significant space overhead, the authors also propose a buffering technique, in which each thread stores the updates to the reduction object in a fixed size buffer. This requires locking, but a single lock is enough. Two schemes are possible, a naive scheme in which threads busy wait to acquire the lock and another one that tries to avoid busy waiting by attempting to update the reduction object every few invocations of the corresponding reduction function.

A straightforward solution is full locking, i.e., the use of one lock per element in the reduction object. However, this incurs high overhead. In order to mitigate this problem, the authors propose three variants.

Fixed locking is based on the use of a limited and predefined number of locks l, so that the item i in the reduction object is assigned to the lock $i \mod l$. Since multiple elements are now associated with the same lock, parallelism is limited and performances suffer. In group locking, a lock is assigned to a group of elements instead of a single element. This variant, like fixed locking, reduces locking overhead but allows limited parallelism as well.

Finally, optimized full locking tries to improve the cache hits. Indeed, in order to overcame the cache misses commonly arising when the number of locks is large, the authors propose the use of data structures storing both an element of the reduction object and the corresponding lock associated with that element.

6.5.1.2 *Apriori*-Based Algorithms

For *Apriori*, the best technique [21] as shown by the experimental results on 4 threads is full replication. Buffering works better than the locking schemes, and, finally, among the locking alternatives, optimized full locking had the best performance.

In [54], three different data representations are tested: vertical tidset, binary representation (bitvector) of vertical tidset and diffset. In practice, a tidset stores all of the transaction IDs containing an item, while a diffset, as its name implies, stores the difference in transaction IDs between an itemset and its prefix. By using a vertical data representation, each thread can compute an independent support in parallel, since there are no data dependencies among the threads. It follows that no complex parallel algorithm is required. However, parallelizing support counting may incur load imbalance.

Regarding the experimental results, the authors report that parallel versions of *Apriori* (implemented using OpenMP) based on vertical tidset or bitvector representations do not scale well beyond 16 cores. The parallel version based on diffset achieves much better scalability, even though the maximum reported speedup is equal to 52 on 256 threads for the mushroom dataset [1].

6.5.1.3 The *CCPD* and *PCCD* Algorithms

Parthasarathy et al. [34] investigated the degree of parallelism, synchronization and data locality issues in shared-memory algorithms for mining association rules. In particular, they found that algorithms for mining association rules exploit complex pointer-based data structures which, typically, suffer from sub-optimal data locality. Moreover, on symmetric multiprocessors shared access to these data structures may result as well in false sharing. Indeed, recursive data structures are built once at the beginning, but are accessed several times in each iteration; the problem is exacerbated by the fact that the access patterns after the build phase are highly ordered.

The authors presented various optimizations for fast frequency computation, based on locality and false sharing sensitive memory placement of these data structures which enhance performance significantly. Two algorithms, common candidate partitioned database (*CCPD*) and partitioned candidate common database (*PCCD*) are compared.

CCPD makes use of a common candidate hash tree across all of the processors, while the database is logically split among them. The hash tree is built in parallel, then each processor traverses its local database and counts the support for each itemset. Finally, the master process selects the frequent itemsets.

PCCD is based on a partitioned candidate hash tree, but uses a common database. Initially, a local candidate hash tree is built per processor. Then, each processor traverses the entire database and counts support for itemsets only in its local tree. Finally the master process performs the reduction and selects the frequent itemsets for the next iteration.

6.5.1.4 Parallel Algorithms Derived by *FP-Growth*

A shared-memory parallel version of *FP-growth*, presented in [12], is based on the idea of building just one FP-Tree which is then partitioned into several independent parts processed by different threads. In order to balance the workload, the authors propose a heuristic, called content-based tree partition. The aim is to alleviate both the locking overhead incurred during the tree building phase and the overhead incurred during the mining phase.

Another approach to the parallelization of *FP-growth* [26] aims at achieving better performances by minimizing the cache misses augmenting data locality through a so-called FP-Array (frequent pattern array) data structure, and maximizing the opportunities for parallelism by exploiting a lock-free dataset tiling parallelization mechanism when building the FP-Tree.

The FP-Array, as discussed by the authors, also benefits from hardware/software data prefetching, a common technique for hiding the access latency of data patterns that defeat caching strategies. In practice, since a cache miss leads to a memory fetch, it is usually best to prefetch the data in advance, without waiting for the actual issue of the corresponding memory reference. This way, by overlapping computation and memory accesses, it is possible to improve the overall running time.

Data tiling is a dataset restructuring operation that reorganizes data in order to improve the existing temporal locality and consequently provides a better usage of the cache. Moreover, it also allows to avoid the use of locks, since all of the tiles are essentially independent from each other. Tiles can then be processed non-exclusively by different threads.

6.5.1.5 A Parallel Algorithm Derived by LCM

Negrevergne et al. [33] present *PLCM*, a parallel version of the sequential *LCM* algorithm (Linear time Closed itemset Miner) [45], which is widely regarded as the most efficient sequential algorithm for mining closed frequent itemsets (see Definition 6.4). In particular, the implementation is based on the Tuple Space concept which was originally proposed by Gelernter in the Linda model. The authors implement their model, *Melinda*, in shared memory, so that work units to be processed by threads are dropped in a global space called the Tuple Space. An idle thread can then extract work units as Tuples from the Tuple Space and process them.

There are two basic primitives, respectively, for insertion (`Put`) and extraction (`Get`). A `Put` adds a tuple in the Tuple Space, and, conversely, a `Get` extracts a tuple from the Tuple Space. In particular, the `Get` primitive has blocking semantics when the Tuple Space is empty. A parallel application terminates execution when the Tuple Space is empty and all of the threads are therefore blocked trying to get a Tuple.

Besides the parallelization of *LCM* in the Melinda environment, the authors also thoroughly discuss several algorithmic optimizations to reduce the overall memory footprint and bandwidth pressure.

6.5.1.6 A Parallel Algorithm Derived by Tree Projection

Teodoro et al. [43] present a shared memory parallel implementation of the Tree Projection sequential algorithm. The authors build, in parallel, the lexicographic tree data structure that will be used to perform the frequent itemsets counting using a breadth-first strategy. This allows to exploit two different opportunities to increase the parallelism. The former is to process transactions in parallel to the available threads, while the latter is the node level parallelism, in which each node being expanded is assigned to a different thread, responsible for processing the corresponding transactions updating its node matrix data structure.

The authors implemented the first strategy based on parallel processing of transactions, since the second strategy may lead to load imbalances and requires much more memory owing to the need to load in memory all of the node matrices concurrently. A problem strictly related to processing the transactions in parallel is that tree nodes that are being expanded are shared among the threads. As a consequence, since different threads may try to update concurrently objects in shared memory, accesses must be synchronized.

To this end, the authors exploit a work queue of transactions shared among the threads, and blocks of transactions are assigned to be processed instead of single transactions in order to minimize the overhead. Locking is also avoided through a lock-free implementation based on hardware provided atomic instructions. The use of these wait-free instructions eliminates the need for critical sections at the tree (one lock per tree level) or node level (one lock per node).

6.5.1.7 A Parallel Algorithm Derived by *Eclat*

Besides *Apriori*, the authors of [54] also investigated a shared memory parallel algorithm for *Eclat*, and again evaluated the tidset, bitvector and diffset representations. They show that a straightforward parallel approach is possible, since the threads can store and compute support independently of each other. Therefore, no locking is required. Unlike *Apriori*, once the transaction data is read, each thread can generate its own transaction data using the most appropriate data representation and start in parallel support counting. It is worth noting here that the generated data can be reused as needed. Moreover, there is no need to share these data among the threads. Since *Eclat* loop iterations are data independent, a parallel implementation can minimize the memory exchange improving the algorithm's scalability.

Regarding experimental results, the authors report a much better scalability with regard to *Apriori* for all of the three data representations. In particular, the maximum speedup equal to 171 is achieved on 256 threads using the tidset implementation for mining the pumsb dataset [1].

6.5.2 Message-Passing Algorithms

In a shared-nothing, or message-passing parallel architecture, processing nodes have private memory and storage and can communicate with each other through exchange of messages on a network infrastructure connecting all of the processing units.

Parallel algorithms are often organized so that the owner processor sends the data to the requestor processor before being asked for, in order to try to minimize the associated latency.

Advantages of this architecture include: much better and easier scalability, memory also scales linearly with the number of processors, processors can access their own memories rapidly, and, finally, the architecture is both cost-effective and easy to build: commodity components can be used (while in shared-memory architectures, there may be the need for dedicated hardware components, especially to handle cache memories).

However, the lack of a shared address space makes this architecture difficult to program since the programmer is now responsible for all of the communication's details. Moreover, data structures can be difficult to distribute; in some cases there is a need to revise them adding additional data (e.g., ghost cells, etc.) before distribution.

We begin this overview with three parallel algorithms developed by Agrawal and Shafer [5]: the *Count Distribution* algorithm, the *Data Distribution* algorithm and the *Candidate Distribution* algorithm. They are a good start to evaluate the challenges that come up in trying to solve the FIM problem in parallel. Furthermore, most of the parallel algorithms that have been developed later are variations and optimizations of these ones.

The three algorithms assume a message-passing architecture and that the input transactions database is evenly distributed among all of the nodes employed in the computation.

6.5.2.1 The *Count Distribution* Algorithm

The *Count Distribution* algorithm tries to minimize the communication among the processors by replicating part of the computation. It is a simple way to parallelize the *Apriori* algorithm, nonetheless an efficient one.

Apart from the first pass which is treated slightly differently, as we shall see later, the algorithm proceeds as follows: during the k-th pass each of the N processors among which the input has been evenly distributed, independently calculates the set C_k of candidate frequent k-itemsets from the set F_{k-1} of frequent $(k-1)$-itemsets.

This step is processed in the same way as in *Apriori*, by constructing a local hash tree and each processor comes up with the same C_k, since the set F_{k-1} is the same for all of the processors. At this point the processors calculate the support of the candidate k-itemsets related to their local portion of the transaction database, and then participate in a reduce-scatter communication through which the global supports of the candidate itemsets are calculated and distributed among all of the processors.

At last each processor can derive F_k from its local copy of C_k by comparing the global supports against the user defined support threshold and, if $F_k \neq \emptyset$, the computation keeps on going with the pass $k + 1$.

It should be noted that processors compute the same hash tree of candidate itemsets and can retrieve them in the same order. In this way, the identity of a candidate itemset can stay implicit and does not have to be communicated, only the counts of the candidate itemsets have to be used in the reduction, saving communication bandwidth and computation.

The first pass, which consists of calculating the set C_1, is performed in a different way: each processor counts the support of the items present in the transactions it holds and then communicates to the other processors the local part of C_1 it has computed. At the end of this communication phase, all of the processors gather the data needed to compute the global C_1.

The good part of the *Count Distribution* algorithm is that the communications are kept at a minimum, with no exchange of transactions' data between processors. Therefore, each processor can proceed independently and only synchronizes and communicates at the end of a pass. However, this comes at a prize as each processor duplicates the data structures needed to store the candidate itemsets and, if the local memory is not sufficient to hold this data, the algorithm becomes easily inefficient.

Furthermore, the algorithm does not make good use of the aggregate memory of the system: even when more processors are employed in the computation, the local memory of each processor rests the same and the dimension of the set of candidate itemsets that could be efficiently processed is limited.

6.5.2.2 The *Data Distribution* Algorithm

The *Data Distribution* algorithm tries to make better use of the aggregate memory of the system, though this comes at the expense of an increase in the need of communications among processors. Indeed, in Data Distribution, also the candidate itemsets are distributed among processors, and each processor only counts the candidate itemsets assigned to it, after gathering the dataset portion allocated to the other processors.

In order to compute C_1, *Data Distribution* performs the first pass as *Count Distribution* does.

On the other hand, during the k-th pass, each processor generates C_k from F_{k-1}, but discards all of the candidate itemsets which are not assigned to it. The local sets of candidate k-itemsets allocated to each processor are disjoint and their union forms the global set of candidate k-itemsets.

Each processor computes the support of the candidate itemsets for which it is responsible using both the local portion of the input dataset and the parts sent by the other processors, then it determines which candidate itemsets actually are frequent itemsets.

At last, processors exchange the local set of frequent itemsets, so that each processor obtains the whole set F_k and, if $F_k \neq \emptyset$, the computation can proceed to the next pass.

6.5.2.3 The *Candidate Distribution* Algorithm

The *Candidate distribution* algorithm tries to reduce the dependency between processors, which have to synchronize at each step in the two algorithms above. It reaches this major independence by partitioning the set of frequent itemsets between the processors starting at a heuristically determined pass l and selectively re-partitioning the input tuples so that each processor can independently proceed in the computation from that pass on.

The first $l - 1$ passes performed by the algorithm are handled as in the *Count distribution* or *Data Distribution* algorithms.

During the l-th pass the set F_{l-1} of frequent $(l-1)$-itemsets is partitioned among the processors in a way that assures a good load-balancing. Then, each processor generates the set of candidate l-itemsets related to its local portion of F_{l-1}. Since the processors hold the whole set F_{l-1}, no problems arise during the pruning step in this pass.

Thereafter, the global supports of the local sets of candidate itemsets are computed and a redistribution of the input dataset is performed, so that each processor will hold all of the tuples it needs to independently compute the global supports in the subsequent passes.

After the global counts are computed, each processor determines the frequent l-itemsets it is responsible for and asynchronously sends it to the other processors. This communication is important for the pruning step in the candidate generation phase of later passes; it is performed asynchronously so that processors do not need to synchronize. In each subsequent pass every processor will perform the pruning step with the information at its disposal at the moment, without attending to receive all of the frequent itemsets from the other processors: the frequent itemsets not used are employed in the pruning step of the successive passes.

During the k-th pass with $k > l$, each node independently performs the local candidate k-itemsets generation from local frequent $(k-1)$-itemsets, discarding the itemsets which are not frequent. This pruning is accomplished by also referring to the frequent $(k-1)$-itemsets sent by the other processors and that have been already received.

At last, the processors compute the local portion of frequent k-itemsets by using only local stored tuples, without further communications, and exchange them with the other processors for the subsequent pass.

6.5.2.4 The *PEAR* and *PPAR* Algorithms

The parallel *PEAR* and *PPAR* algorithms are derived from the sequential counterparts, the *SEAR* (Sequential Efficient Association Rules) and *SPEAR* (partitioned version of *SEAR*) algorithms designed by Mueller [32].

The *SEAR* algorithm is similar to *Apriori*, except for the usage of a more efficient data structure to store the frequent and candidate itemsets and to count their support. It generates candidate itemsets as in *Apriori*, but, instead of storing candidate itemsets in the tree-like structure of *Apriori*, it uses a prefix-tree structure, where each edge represents an item and each node stores the count of the itemset whose items are the ones represented by the edges along the path from the root to that node.

This data structure allows a more efficient generation of candidate itemsets and a quicker counting of their supports. Furthermore, it stores not just the candidate itemsets of a pass but also the frequent itemsets of the previous pass.

The *PEAR* algorithm follows the same steps as the *Count Distribution* algorithm but it uses the data structures and optimization of *SEAR* in candidate generation and local support counting.

On the other hand, the *PPAR* algorithm proceeds in four phases. The transaction database is partitioned among all of the processors and, in the first phase, each processor computes the local frequent itemsets, i.e., the frequent itemsets with reference to only the local partition of the transaction dataset. This phase can be performed by each processor independently from the others, since no communications are required.

In the second phase, the local frequent itemsets are joined to form the set of all of the global candidate frequent itemsets, which is then broadcasted to all of the processors. In phase 3, each processor computes the local support for all of the candidate global frequent sets with the exclusion of the ones already computed in phase one.

In phase 4 the local counts are combined with a reduce-scatter operation in order to compute the global support of each global candidate. At last, all of the globally infrequent sets are discarded and the final result produced.

PPAR requires the processors to communicate and synchronize only in phase 2 and 4, and to exchange only the counts for the locally frequent itemsets. Despite its minor communication cost, though, the algorithm is outperformed by *PEAR*.

6.5.2.5 *Eclat* Derived Algorithms

Zaki et al. present in [50] four parallel algorithms based on the observation that maximal frequent itemsets (see Definition 6.3) allow to detect the boundary in the lattice between the set of frequent itemsets and the set of infrequent ones. Looking at Fig. 6.3, the boundary is determined by the sub-lattices induced by the maximal frequent itemsets: all of the frequent itemsets are below that border.

All of the four algorithms first search for a set of potential maximal frequent itemsets and then use this set to determine the other frequent itemsets. As in *Eclat*, these algorithms make use of the vertical database format and differ in the techniques used to cluster and traverse the lattice.

The itemset clustering allows to group itemsets in order to obtain a superset of the maximal frequent itemsets—the set of potential maximal frequent itemsets. Two clustering schemes have been designed, differing in precision and computational cost. The first one is based on equivalence classes, the other one uses the uniform hypergraph cliques, which is more precise in the determination of the maximal itemsets, but heavier in terms of computational cost.

Once the set of potential maximal frequent itemsets has been determined, for each of the potential maximal itemsets the sub-lattice induced must be traversed in order to find the true frequent itemsets. Two different traversal schemes are employed, a purely bottom-up approach and a hybrid top-down/bottom-up scheme.

In the bottom-up traversal, frequent itemsets are generated in a breadth-first fashion: itemsets of level k are generated before itemsets of level $k+1$. This approach is the one used in *Apriori* and derived algorithms. On the other hand, the hybrid approach is designed to find the true maximal frequent itemsets faster, making better use of the clustering information and overcoming the issues of a pure top-down approach.

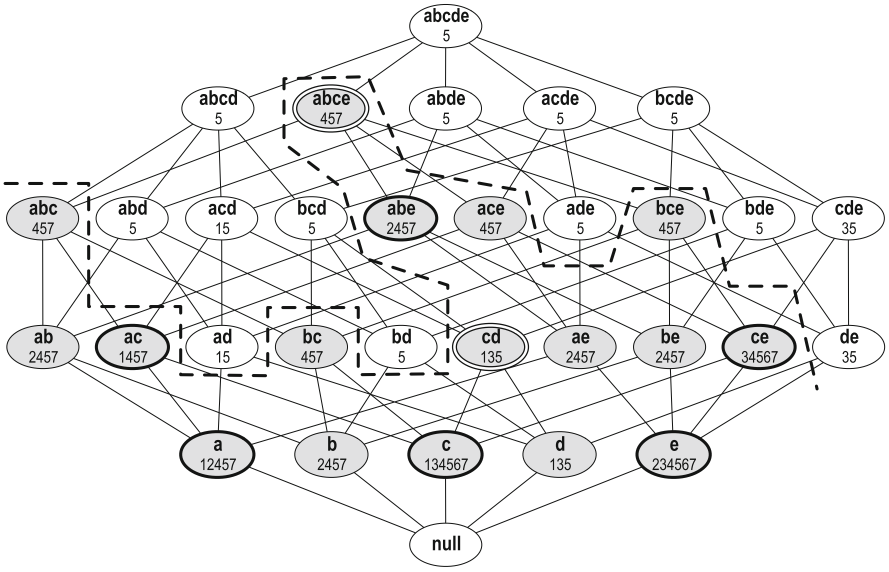


Fig. 6.3 Boundary between frequent and infrequent itemsets

The four algorithms, along with their names and the clustering and traversal techniques used, are reported in Table 6.4.

Table 6.4 Clustering and traversal schemes in Zaki et al. algorithms

Algorithm	Clustering technique	Traversal scheme
Par-Eclat	Equivalence class	Bottom-up
Par-MaxEclat	Equivalence class	Hybrid
Par-Clique	Maximal uniform hypergraph clique	Bottom-up
Par-MaxClique	Maximal uniform hypergraph clique	Hybrid

Each algorithm begins with an initialization phase, where the set of all of the frequent 2-itemsets is computed and one of the clustering schemes is applied. The clusters are then partitioned among all of the processors and the database is also repartitioned so that each processor also owns the tid-lists of the items assigned to it.

After the initialization step, each processor can proceed independently, computing the frequent itemsets of the assigned clusters with either a bottom-up or hybrid traversal. At last, a reduction phase aggregates the results and produces the output.

In contrast to Count and Candidate Distribution, the algorithms proposed by Zaki et al. make use of the aggregate memory of the parallel system and make the processors independent in the initial phases of the computation. Furthermore, they make use of the vertical database layout, which allows to limit the number of scans of the local database to just two, and computing the frequent itemsets without the overhead of complicated data structures.

6.5.2.6 Par-FP

Par-FP has been proposed by Cong et al. [13]. This algorithm takes advantage of the distinguished features of the *FP-growth* sequential algorithm and the fact that it attacks the frequent itemsets problem with a divide and conquer strategy.

The algorithm proceeds into three steps: in the first step the frequent items in the input dataset are found, together with the associated projected databases. The frequent items are determined in parallel with a final reduction operation. In the second step, the frequent items are partitioned among the available processors. Furthermore, each processor obtains the projected databases corresponding to the frequent items it is in charge of. In the third step, each processor can proceed with the mining of the frequent itemsets containing its frequent items independently, without further communications.

The main difficulty concerning the implementation of the algorithm consists in finding a good load balancing strategy. Indeed, the projected databases usually have different sizes and a simple fair distribution of the frequent items among the processors could produce a highly unbalanced workload and consequently worst performance.

To overcome this problem, an initial sampling of the input database is performed in order to estimate the mining time of the projected database associated with each frequent item. The projected databases whose mining time exceed a threshold, which is a function of the average mining time, are divided further into subtasks before task scheduling.

Cong et al. show that a simple random sampling does not work well as sampling strategy. Therefore, they propose an ad-hoc selective sampling strategy which analyses all of the transactions in the input dataset but, before the mining time is computed, removes from each transaction the non-frequent items and a part of the most frequent ones. The mining time for each selected frequent item is computed by a single processor by running the sequential *FP-growth* algorithm over the sampled transactions and recording the time needed to mine each projected database.

The algorithm has been implemented in a message-passing model and experimental results with 64 processors show good performances with regard to parallel speedup.

6.5.3 Map-Reduce Algorithms

The design of a parallel algorithm presents various challenges such as proper input data partitioning, good workload balancing and minimization of communication costs. Furthermore, in a distributed or grid environment, the possibility of node failures and task crashes increases the design difficulties and the errors in computations as the number of nodes increases, limiting the algorithm scalability.

The map-reduce framework was introduced to overcome these issues and simplify the task of parallel algorithms design. In a map-reduce cluster, storage is managed by a distributed filesystem such as HDFS (Hadoop Distributed File System) or GFS (Google File System) and data is partitioned into disjoint chunks, blocks of equal size with a fixed number of replicas, which are then distributed among the cluster nodes. Map-reduce deals with node failure by re-executing a crashed task and not the whole computation, and it handles workload by reassigning un-finished tasks of slower nodes to idle nodes which have already completed the processing of their tasks.

A computation in map-reduce proceeds in two steps. In the first step a Map function is executed on input data. In the second step, a reduce function processes the intermediate results produced by mapper tasks. A Map function processes a chunk of input data and outputs a set of <key, value> pairs which are then sorted and compressed by combining pairs with the same key. The Reduce function takes the sequence of <key, value> pairs produced by the map function and outputs the results of its computations to a file.

The design of a map-reduce algorithm consists in the definition of Map and Reduce functions that compute the desired results. A node (the master) is designated to schedule the execution of map and reduce jobs on the other nodes (the workers). Map and Reduce functions can be allocated to the same node. Each worker node involved in the computation can then independently execute both the Map function on its chunks of data and the Reduce function on the results of the map task.

6.5.3.1 *Apriori*-Based Map-Reduce Algorithms

In [24] Ming-Yen Lin et al. have proposed three algorithms based on *Apriori* and implemented taking advantage of the Map-Reduce framework. All of the three algorithms assign the two computation phases of the *Apriori* algorithm, that is, candidate generation and support counting, respectively, to the Map and Reduce functions of the map-reduce framework.

The mappers—i.e., the nodes executing the Map function—generate the candidate sets and count the occurrences of each candidate in their dataset partition. They output <itemset, count> pairs. Reducers—nodes that execute the Reduce function—collect and sum itemset counts and output to file the itemsets whose count exceeds the support threshold.

The first algorithm, the Single Pass Counting (*SPC*) algorithm, proceeds in a level-wise fashion. During each map-reduce pass the set of all of the itemsets of same length are produced, so that k passes of map-reduce and k scans of the transaction dataset are required to obtain all of the frequent itemsets of length up to k. This design is not optimal, because the number of candidates tends to decrease in later passes and the workers remain underused. Furthermore, the scheduling of work in these cases becomes an overhead and the cost of database loading too high in comparison to the workload of workers.

To overcome these issues a second algorithm was presented by Lin et al., the Fixed Passes Combined-counting (*FPC*) algorithm which combines a number of successive passes into one pass. In map phases after the first two, three sets of candidates are generated for the reduce phase, that is, starting from $k = 3$ the mappers read the file containing the frequent $k - 1$-itemsets and generate the set of candidates of length k, $k + 1$ and $k + 2$, where the candidates of length $k + 1$ and $k + 2$ are produced, respectively, from the set of k and $k + 1$ candidates. Then, the reduce phase proceeds as in the *SPC* algorithm, but processing the union of the candidates of length k to $k + 2$ and outputting three set of frequent itemsets, those of length k, $k + 1$ and $k + 2$.

Even though the number of database scans is reduced and the workers are better utilized in later phases of the computation, this design turns out to be even worse. Indeed, the number of candidates in earlier combined passes can be too high and overload the workers; furthermore, the number of false positives increases due to the lack of a pruning step in the combined passes.

The Dynamic Passes Counting (*DPC*) algorithm tries to improve on *FPC* by adopting a dynamic approach. While *FPC* combines a fixed number of passes, in *DPC*, after the first two phases, mappers generate candidates of longer length, combining a varying number of passes, until a threshold on the total number of candidates is reached. The threshold is computed and dynamically updated after each map-reduce phase taking into account the execution time of completed phases and the number of frequent itemsets to be used for candidate generation in the subsequent phase. Experimental results show that *DPC* has good scalability and achieves a much better trade-off between the need of reducing the number of database scans and that of increasing the number of pruned candidates and not overloading the workers.

6.5.3.2 *Eclat* Derived Map-Reduce Algorithms

The breadth-first approach of *Apriori* based Map-Reduce algorithms is effective on computing short frequent itemsets, but their level-wise nature can cause high scheduling overhead because of the several Map-Reduce phases needed, especially in long frequent itemsets search. To overcome these issues Moen et al. [31] propose two algorithms, one based on pure *Eclat*, *Dist-Eclat*, and one, *BigFIM*, presenting a hybrid approach which makes use of both breadth-first search and *Eclat* to better handle very large datasets (Big Data).

Dist-Eclat is based on the *dEclat* variant of *Eclat*, because of its better memory requirements. It makes use of databases in vertical format, and proceeds in three steps, each of which consists of a map-reduce phase. During the first step, the input database is distributed among the mappers in equal sized blocks, and each mapper computes the frequent items related to the dataset partition assigned to it. Then, reducers collect all of the frequent items without the need for further processing.

In the second step, the frequent items are distributed round-robin to the available mappers, and each mapper proceeds to the computation of the frequent k-itemsets which are supersets of the items assigned to it, running locally the algorithm dEclat until the level k. Then, during the reduce phase, the set of all of the frequent k-itemsets is distributed among a new set of mappers, using again a round-robin strategy.

In the last step, each mapper runs *Eclat* on the prefix subtrees starting from the assigned frequent k-itemsets until all of the levels of frequent itemsets are mined. This step does not require any further communication. When all of the mappers are done, the reducers collect all of the frequent itemsets found and output the result.

BigFIM tries to overcome the problems of *Dist-Eclat*, especially with very large datasets. Indeed, to mine the frequent 2-itemsets in the second step, many mappers could need to access the whole dataset. As a consequence, the cost of communications could be too high or the computation infeasible in case of Big Data, where even a single item tid-list could be too long to fit in memory.

BigFIM adopts a hybrid approach with a breadth-first search similar to *Apriori* for mining the first k levels of frequent itemsets and employing *Eclat* in a subsequent step when the projected databases could fit in memory.

In the first step of the algorithm the database is evenly distributed among the mappers which proceed to mine the local blocks of data for frequent itemsets in a level-wise manner. In each level, the locally frequent itemsets mined by the mappers are combined by the reducers in order to determine the globally frequent itemsets. Therefore, k map-reduce phases are needed to obtain the set of frequent k-itemsets serving as prefixes in the next step of the algorithm.

In the second step the prefixes are distributed among the mappers, and each one independently mines the conditional database associated with the assigned prefix group using dEclat. Eventually, the reducers collect all of the mined frequent itemsets and output the result to a file.

6.5.4 FIM on GPUs

At last, we present some algorithms that try to exploit the potential of massive parallel computing on GPUs. General Purpose computing on Graphical Processor Units (GPGPU) presents many opportunities for massive parallelization through thousands of threads, but also peculiar challenges because of the particular features and constraints of GPU architectures. Owing to significant cost reductions in recent years, GPGPU is increasingly accessible and used in small and medium-sized enterprises (SMEs).

A GPU contains several multiprocessors that share a common device (global) memory, with each multiprocessor consisting of a number of cores and a shared local memory. Threads are organized in blocks, and multiple blocks can be scheduled for execution on the GPU. Each thread has a small local memory and all of the threads in a block can access the multiprocessor memory, through which they can cooperate and synchronize. On the other hand, the order of execution of blocks is not specified and, even though blocks can access the global memory, they cannot rely on it for inter-block communications and synchronizations. Furthermore, access to the memory hierarchy must be carefully organized in order to maximize throughput and to avoid access serialization.

An algorithm designed to be executed on a GPU must take into account these peculiarities and constraints. The computation is usually organized into subproblems, each one solved cooperatively by the threads of a block, while data movements from shared memory to global memory and vice versa are explicitly handled.

The first two algorithms we discuss are *GPApriori*, presented by Zhang et al. [53] and based on *Apriori*, and *gpuDCI*, proposed by Silvestri and Orlando [41] and derived from a sequential optimization of *Apriori*, the *DIC* algorithm.

Both algorithms delegate to the GPU the support counting of candidates, whose generation is assigned to the CPU, and make use of a binary dataset representation called bitset, a vertical database representation in which a bitmap is associated with each item. Each bit in the bitmap represents a transaction and is set to 1 when the item is contained in that transaction or 0 otherwise.

The choice of a bitset representation is justified, though it usually requires more memory, because it is more suitable to exploit the parallelization opportunities of GPUs. Support counting of itemsets is accomplished by intersecting the bitmaps of constituent items, through a bitwise AND, and counting the bits set to 1 in the computed bitmap. The bitmaps to be intersected are moved to the GPU device memory and distributed among the threads of each block so that consecutive threads reads consecutive memory allocations and global memory accesses are reduced (a technique called "coalescing memory accesses").

After the bitwise AND, that is executed independently by each thread, a two step parallel reduction is executed. The first step reduces the results of threads inside each block, the second step collects and merges the reduction partial results of the first step. The support count is then transferred to CPU main memory. Indeed, this implementation technique has general application and was first used in histogram computations [37].

The *gpuDCI* algorithm adds to this design some optimizations derived from the sequential *DCI*. In particular, it makes use of a cache to store intermediate results of bitmap intersections. In fact, processing two candidates that share a common prefix would lead to redundant operations that can be avoided if intermediate results are cached and reused.

Furthermore, *gpuDCI* introduces two different approaches in parallelizing the computation of candidate support counts. The first strategy is called *transaction-wise* and requires that a single candidate support count is computed by all of the thread blocks at a time as previously described. The second strategy, called *candidate-wise*, consists in computing more than one candidate support count at a time. A set of consecutive lexicographic-ordered candidates are assigned to each block of threads and processed one at a time by that block. This second approach requires more memory due to the need of storing a cache of intermediate results for each block, but leads to better performances as assessed by experiments conducted by the authors.

Two further solutions also based on *Apriori* are presented by Fang et al. in [15]. Both implementations make use of bitmaps and perform support counting in parallel to the GPU with an approach already seen in the previous algorithms, the difference being the use of a lookup table which associates the integer represented by an item bitmap to the number of 1s in that bitmap, an expedient which allows faster itemset support computation. The first implementation shown is *TBI* (Trie-Based Implementation); it relies on a trie data structure for frequent itemsets storage and candidate generation which is done on the CPU as in other solutions. However, the second implementation proposed, *PBI* (Pure Bitmap Implementation), adopts bitmaps also to store frequent itemsets and this enables the candidate generation on the GPU, so that *PBI* performs entirely on the GPU.

A different approach is followed by Arour and Belkahla [6]. They present a solution based on *FP-growth*, named *gpuFPgrowth+*. In their algorithms, they make use of a different representation of the FP-tree, a representation which was first introduced by Li and Liu [26]. This structure, named FP-Array, allows improving the cache performance and is better suited to multi-core and GPU-based implementations of *FP-growth*. Indeed, the authors show two implementations, the first one targeted to multi-core architectures and the other to GPUs. Regarding the latter implementation, the computation starts on the CPU where frequent 1-itemsets are determined and an FP-Array is built. Then, processing continues on the GPU and, once completed, results are transferred from device memory to main memory.

All of the algorithms reviewed and the speedup they exhibit over the corresponding sequential solutions, as stated by the respective authors, show that GPUs are valid platforms for FIM, despite their limits and the constraints they impose. Therefore, it is worth investigating the design of suitable parallel algorithms for this architecture.

6.6 Implementation Repositories

We provide here, for the reader's convenience, information regarding selected, publicly available implementation repositories (Table 6.5).

Table 6.5 Implementation repositories

Repository	URL
FIMI	http://fimi.ua.ac.be
SPMF	http://www.philippe-fournier-viger.com/spmf/

The Frequent Itemset Mining Implementations (FIMI) repository is a web site that collects the result of the workshops on Frequent Itemset Mining Implementations, FIMI'03 and FIMI'04. Besides the source codes of all of the implementations that were accepted at the FIMI workshops, interested readers will also find several publicly available datasets.

Although several authors show experimental results measuring the performance of their algorithms in the corresponding papers, an independent and more focused analysis on performances could be found on the proceedings of FIMI'03 [17] and FIMI'04 [39], along with a comparison among different algorithms and implementations.

SPMF is an open-source data mining library written in Java by Philippe Fournier-Viger, specialized in pattern mining. It is distributed under the GPL v3 licence and provides implementations of many different data mining algorithms. The source code of each algorithm can be easily integrated in other Java software. Moreover, SPMF can be used as a standalone program with a simple user interface or from the command line. SPMF is fast and lightweight (no dependencies to other libraries). The web site provide datasets as well.

6.7 Conclusions

We presented a survey of the most important parallel algorithms that have been proposed in the context of the FIM problem. We started reviewing basic sequential algorithms on which most of the parallel algorithm are based, and then discussed and compared different parallel approaches based on shared-memory, message-passing, map-reduce and the use of GPU accelerators. Even though our survey certainly is not exhaustive, it covers essential reference material, since we believe that an attempt to cover everything will instead fail to convey any useful information to the interested readers. Our hope is that it will help interested researchers and practitioners, in particular those coming from a business-oriented background, quickly enabling them to develop their understanding of an area likely to play an ever more significant role in coming years.

References

1. Frequent itemset mining dataset repository. http://fimi.ua.ac.be/data/.
2. Charu C Aggarwal. *Data mining: The textbook*. Springer, 2015.
3. Charu C Aggarwal and Jiawei Han. *Frequent pattern mining*. Springer, 2014.
4. Rakesh Agrawal, Tomasz Imieliński, and Arun Swami. Mining association rules between sets of items in large databases. *ACM SIGMOD Record*, 22(2):207–216, 1993.
5. Rakesh Agrawal and John C. Shafer. Parallel mining of association rules. *IEEE Trans. on Knowl. and Data Eng.*, 8(6):962–969, December 1996.
6. Khedija Arour and Amani Belkahla. Frequent pattern-growth algorithm on multi-core CPU and GPU processors. *CIT. Journal of Computing and Information Technology*, 22(3):159–169, 2014.
7. Gowtham Atluri, Rohit Gupta, Gang Fang, Gaurav Pandey, Michael Steinbach, and Vipin Kumar. Association analysis techniques for bioinformatics problems. In *Bioinformatics and Computational Biology*, pages 1–13. Springer, 2009.
8. Roberto J Bayardo Jr. Efficiently mining long patterns from databases. *ACM Sigmod Record*, 27(2):85–93, 1998.
9. Florian Beil, Martin Ester, and Xiaowei Xu. Frequent term-based text clustering. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 436–442. ACM, 2002.
10. Daniel Boley, Maria Gini, Robert Gross, Eui-Hong Sam Han, Kyle Hastings, George Karypis, Vipin Kumar, Bamshad Mobasher, and Jerome Moore. Partitioning-based clustering for web document categorization. *Decision Support Systems*, 27(3):329–341, 1999.
11. Doug Burdick, Manuel Calimlim, and Johannes Gehrke. Mafia: A maximal frequent itemset algorithm for transactional databases. In *Data Engineering, 2001. Proceedings. 17th International Conference on*, pages 443–452. IEEE, 2001.
12. Dehao Chen, Chunrong Lai, Wei Hu, Wenguang Chen, Yimin Zhang, and Weimin Zheng. Tree partition based parallel frequent pattern mining on shared memory systems. In *Parallel and Distributed Processing Symposium, 2006. IPDPS 2006. 20th International*, 2006.
13. Shengnan Cong, Jiawei Han, Jay Hoeflinger, and David Padua. A sampling-based framework for parallel data mining. In *Proceedings of the Tenth ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming*, PPoPP '05, pages 255–265. ACM, 2005.
14. Guozhu Dong and Jinyan Li. Efficient mining of emerging patterns: Discovering trends and differences. In *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 43–52. ACM, 1999.
15. Wenbin Fang, Mian Lu, Xiangye Xiao, Bingsheng He, and Qiong Luo. Frequent itemset mining on graphics processors. In *Proceedings of the fifth international workshop on data management on new hardware*, pages 34–42. ACM, 2009.
16. Benjamin CM Fung, Ke Wang, and Martin Ester. Hierarchical document clustering using frequent itemsets. In *SDM*, volume 3, pages 59–70. SIAM, 2003.
17. Bart Goethals and Mohammed J. Zaki, editors. *Proceedings of the IEEE ICDM Workshop on Frequent Itemset Mining Implementations*, 2003. http://www.ceur-ws.org/Vol-90/.
18. Karam Gouda and Mohammed Zaki. Efficiently mining maximal frequent itemsets. In *Data Mining, 2001. ICDM 2001, Proceedings IEEE International Conference on*, pages 163–170. IEEE, 2001.
19. Jiawei Han, Jian Pei, and Yiwen Yin. Mining frequent patterns without candidate generation. *ACM SIGMOD Record*, 29(2):1–12, 2000.
20. Renáta Iváncsy and István Vajk. Frequent pattern mining in web log data. *Acta Polytechnica Hungarica*, 3(1):77–90, 2006.
21. Ruoming Jin, Ge Yang, and G. Agrawal. Shared memory parallelization of data mining algorithms: techniques, programming interface, and performance. *Knowledge and Data Engineering, IEEE Transactions on*, 17(1):71–89, 2005.

22. Wenke Lee, Salvatore J Stolfo, and Kui W Mok. Mining audit data to build intrusion detection models. In *KDD*, pages 66–72, 1998.
23. Kingsly Leung and Christopher Leckie. Unsupervised anomaly detection in network intrusion detection using clusters. In *Proceedings of the Twenty-eighth Australasian conference on Computer Science-Volume 38*, pages 333–342. Australian Computer Society, Inc., 2005.
24. Ming-Yen Lin, Pei-Yu Lee, and Sue-Chen Hsueh. Apriori-based frequent itemset mining algorithms on MapReduce. In *Proceedings of the 6th International Conference on Ubiquitous Information Management and Communication - ICUIMC '12*, page 1, New York, New York, USA, Feb 2012. ACM Press.
25. Weiyang Lin, Sergio A Alvarez, and Carolina Ruiz. Efficient adaptive-support association rule mining for recommender systems. *Data mining and knowledge discovery*, 6(1):83–105, 2002.
26. Li Liu, Eric Li, Yimin Zhang, and Zhizhong Tang. Optimization of frequent itemset mining on multiple-core processor. In *Proceedings of the 33rd international conference on Very large data bases*, pages 1275–1285. VLDB Endowment, 2007.
27. Elsa Loekito and James Bailey. Fast mining of high dimensional expressive contrast patterns using zero-suppressed binary decision diagrams. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 307–316. ACM, 2006.
28. Stéphane Lopes, Jean-Marc Petit, and Lotfi Lakhal. Efficient discovery of functional dependencies and Armstrong relations. In *EDBT*, volume 1777, pages 350–364. Springer, 2000.
29. Bamshad Mobasher, Honghua Dai, Tao Luo, and Miki Nakagawa. Effective personalization based on association rule discovery from web usage data. In *Proceedings of the 3rd international workshop on Web information and data management*, pages 9–15. ACM, 2001.
30. Bamshad Mobasher, Namit Jain, Eui-Hong Han, and Jaideep Srivastava. Web mining: Pattern discovery from world wide web transactions. Technical report, Technical Report TR96-050, Department of Computer Science, University of Minnesota, 1996.
31. Sandy Moens, Emin Aksehirli, and Bart Goethals. Frequent Itemset Mining for Big Data. In *2013 IEEE International Conference on Big Data*, pages 111–118. IEEE, Oct 2013.
32. Andreas Mueller. Fast sequential and parallel algorithms for association rule mining: A comparison. Technical report, 1995.
33. B. Negrevergne, A. Termier, J. Mehaut, and T. Uno. Discovering closed frequent itemsets on multicore: Parallelizing computations and optimizing memory accesses. In *High Performance Computing and Simulation (HPCS), 2010 International Conference on*, pages 521–528, 2010.
34. Srinivasan Parthasarathy, Mohammed Javeed Zaki, Mitsunori Ogihara, and Wei Li. Parallel data mining for association rules on shared-memory systems. *Knowledge and Information Systems*, 3(1):1–29, 2001.
35. Nicolas Pasquier, Yves Bastide, Rafik Taouil, and Lotfi Lakhal. Discovering frequent closed itemsets for association rules. *Database Theory – ICDT'99*, pages 398–416, 1999.
36. Jian Pei, Jiawei Han, and Runying Mao. Closet: An efficient algorithm for mining frequent closed itemsets. In *ACM SIGMOD workshop on research issues in data mining and knowledge discovery*, volume 4, pages 21–30, 2000.
37. Victor Podlozhnyuk. Histogram calculation on CUDA. http://developer.download.nvidia.com/compute/cuda/1.1-Beta/x86_website/projects/histogram64/doc/histogram.pdf.
38. Anand Rajaraman, Jeffrey D Ullman, Jeffrey David Ullman, and Jeffrey David Ullman. *Mining of massive datasets*, volume 1. Cambridge University Press Cambridge, 2012.
39. Bart Goethals Roberto Bayardo and Mohammed J. Zaki, editors. *Proceedings of the IEEE ICDM Workshop on Frequent Itemset Mining Implementations*, 2004. http://www.ceur-ws.org/Vol-126/.
40. Ken Satoh and Takeaki Uno. Enumerating maximal frequent sets using irredundant dualization. In *Discovery Science*, pages 256–268. Springer, 2003.
41. C. Silvestri and S. Orlando. gpuDCI: Exploiting GPUs in frequent itemset mining. In *Parallel, Distributed and Network-Based Processing (PDP), 2012 20th Euromicro International Conference on*, pages 416–425, Feb 2012.

42. Gerd Stumme, Rafik Taouil, Yves Bastide, Nicolas Pasquier, and Lotfi Lakhal. Computing iceberg concept lattices with titanic. *Data & knowledge engineering*, 42(2):189–222, 2002.
43. G. Teodoro, N. Mariano, W. Meira, and R. Ferreira. Tree projection-based frequent itemset mining on multicore CPUs and GPUs. In *Computer Architecture and High Performance Computing (SBAC-PAD), 2010 22nd International Symposium on*, pages 47–54, 2010.
44. Pawel Terlecki and Krzysztof Walczak. Jumping emerging patterns with negation in transaction databases–classification and discovery. *Information Sciences*, 177(24):5675–5690, 2007.
45. Takeaki Uno, Masashi Kiyomi, and Hiroki Arimura. LCM ver. 2: Efficient Mining Algorithms for Frequent/Closed/Maximal Itemsets. In *Workshop on Frequent Itemset Mining Implementations*, 2004.
46. Renato Vimieiro and Pablo Moscato. Mining disjunctive minimal generators with TitanicOR. *Expert Systems with Applications*, 39(9):8228–8238, 2012.
47. Renato Vimieiro and Pablo Moscato. Disclosed: An efficient depth-first, top-down algorithm for mining disjunctive closed itemsets in high-dimensional data. *Information Sciences*, 280:171–187, 2014.
48. Renato Vimieiro and Pablo Moscato. A new method for mining disjunctive emerging patterns in high-dimensional datasets using hypergraphs. *Information Systems*, 40:1–10, 2014.
49. Mohammed J Zaki. Scalable algorithms for association mining. *Knowledge and Data Engineering, IEEE Transactions on*, 12(3):372–390, 2000.
50. Mohammed J. Zaki, Srinivasan Parthasarathy, Mitsunori Ogihara, and Wei Li. Parallel algorithms for discovery of association rules. *Data Min. Knowl. Discov.*, 1(4):343–373, December 1997.
51. Mohammed J. Zaki and Jr. Wagner Meira. *Data Mining and Analysis: Fundamental Concepts and Algorithms*. Cambridge University Press, May 2014.
52. Mohammed Javeed Zaki and Ching-Jiu Hsiao. Charm: An efficient algorithm for closed itemset mining. In *SDM*, volume 2, pages 457–473, 2002.
53. Fan Zhang, Yan Zhang, and J. Bakos. GPApriori: GPU-accelerated frequent itemset mining. In *Cluster Computing (CLUSTER), 2011 IEEE International Conference on*, pages 590–594, Sept 2011.
54. Yan Zhang, Fan Zhang, and Jason Bakos. Frequent itemset mining on large-scale shared memory machines. In *Cluster Computing (CLUSTER), 2011 IEEE International Conference on*, pages 585–589. IEEE, 2011.
55. Lizhuang Zhao, Mohammed J Zaki, and Naren Ramakrishnan. Blosom: a framework for mining arbitrary boolean expressions. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 827–832. ACM, 2006.

Part III
Networks and Community Detection

Chapter 7
Business Network Analytics: From Graphs to Supernetworks

Pablo Moscato

Abstract A large number of problems in business and consumer analytics have input graphs or networks. These mathematical entities have a long standing tradition in discrete applied mathematics and computer science. In many cases, they are the most natural means to represent some type of relationships in data. Consequently, a large number of solution methods based on heuristics and exact algorithms exist for problems that have graphs and/or networks as part of their input. While the number of possible applications of these techniques is not limited to problems in business and customer analytics, we have chosen to present some of them in a survey that would allow newcomers to the field of data science to create some familiarity with the key questions that motivate the area. We have also provided a survey on recent applications and new algorithmic approaches for data analytics. In addition we discuss issues related to the computational complexity of some problems associated with them. Other chapters of this section complement the discussion in this chapter with specific examples of interest or that could motivate new novel research direction and application.

Keywords (α,β)-k-feature set · k-Truss decomposition of a graph · Community detection · Data analytics · Cartel formation · Graph theory · Marketing · Music recommendation · Network · Network alignment · News recommendation · Operations research · Ranking topic · Supernetworks · Viral marketing

P. Moscato (✉)
School of Electrical Engineering and Computing, The University of Newcastle, Callaghan, NSW, Australia
e-mail: Pablo.Moscato@newcastle.edu.au

P. Moscato, N. J. de Vries (eds.), *Business and Consumer Analytics: New Ideas*,
https://doi.org/10.1007/978-3-030-06222-4_7

7.1 Network Science: The New Field That is Hundreds of Years Old

There is an unquestionable excitement around the world regarding the possibilities that the mathematical study of networks can bring to science, technology, and their applications to society. We do share that excitement. How can it be any different when the first line of the abstract of a scientific article in the field says [117]:

> It is estimated that Italian Mafias registered 135 billion euros in profits only in 2010.

This immediately strikes a chord with us. We see that a centuries-old branch of discrete mathematics, one that is based on connecting nodes with lines, the one that we research and develop, suddenly becomes an incredibly useful tool to fight organized crime or at least to understand its strengths and weaknesses.

Indeed, applications of *network science* are promised in the analysis of criminality [66, 215, 254, 293, 303], education [312], and even ecology [63, 199]. The list is immense.

For mathematicians, graphs and networks and the algorithms for the problems that are defined on them are part of their own treasured "cellar of good old wines", metaphorically speaking. A collection of well known problems and algorithmic results obtained over decades of work. It is for them, perhaps, not news that they are incredibly important. However, this field is very exciting for everybody now. Some techniques considered in the past to be too slow have now become practical due to the new types of hardware. Allied to the fact of the ever expanding computer science list of breakthroughs, solving problems defined on large-scale graphs and networks of interest has become first priority in business and consumer analytics.

To try to make some justice to our pioneers, we give some examples of early applications of graphs and networks, and we start from some interesting cases when computers were not available. We aim to inspire the reader with this historical journey down memory lane and develop an appreciation that graphs and networks were useful tools for analysis many years before hardware brought some extra magic in our lives.

7.1.1 The Bridges of Kaliningrad

The so-called *network science* is not that new at all and we cannot do it justice here reviewing it in a single chapter. Many accounts exist about the origins of this area of mathematics. While representing structures that help create abstractions of real-world data by using nodes connected by lines may have occurred before in other parts of the world and with other civilizations, the mathematical field of *Graph Theory* is currently declared as being born out of the ingenuity of Leonhard Euler in 1735. It happened in Kaliningrad, a city that is now part of Russia. At that time it was a Prussian city and it had a different name. The city extended on the margins of the Pregel river. The 2 islands of Kneiphof were in the middle of the river and, by that time, it was connected to the river, thanks to seven bridges. Interestingly, the field of graph theory started with a riddle.

The habitants of the city were very curious with a particular question: no person had been able to walk across the seven bridges without crossing any one of the bridges twice or more times. This naturally called a more basic question, "Is it feasible?", i.e., *"Is it possible to find a path in which you only cross each bridge a single time?"*

When presented with this problem, Euler, perhaps one of the world's most famous mathematicians in the recorded history of the field, solved the problem with a stroke of genius. He proposed a solution to the riddle by implicitly representing each land area with a symbol indicating a "node" (now also known as a "vertex" in graph theory terms) and each bridge with a line that connected the two dots representing the land areas the bridge connected (i.e., introducing the concept of an "edge"). He then observed the following. First, the path should have a start and it will end not necessarily in the same land area in which it started. Second, any land area visited, if it is not the start or the end of the path, should have an even number of connections in the path (because you need to visit it and leave it to continue in the path, and you can do this any number of times, there is no constraint there). Since the graph has all nodes having an odd number of edges incident to them, all of them can be either the start or the end of the path, but they cannot be any internal node to the path. In conclusion, no such path is possible. The city was known at that time as Königsberg, and the problem Euler solved for the "Königsberg instance", is now known, in general, as the *Eulerian Path problem in graphs*. In fact, Euler was not satisfied with "just" solving this particular case and gave a more general solution [244].

7.1.2 The Icosian Game

The previous problem was posed to Euler as a puzzle. It could have been interesting if this was also the beginning of the "Smart City" movement in a "pre-digital" era; for instance, the problem could have been motivated by a cleaning service of bridges that needed to be optimized. Instead, it came from a leisure activity of the city inhabitants. Another early graph theory reference, one that also shares a recreational perspective, came from William Rowan Hamilton, Astronomer Royal of Ireland, who invented the "Icosian Game" in 1857.

Hamilton intended that one person should pose a puzzle and the adversary should solve it. The game was facilitated via a wooden board with a projection of an icosaedron drawn on it and ivory or bone plugs labelled with integer numbers (from 1 to 20) that can be inserted on the holes representing the vertices of the icosahedron.[1] In the Icosian Game, the objective is to find a cycle that starts in one vertex of a dodecahedron and, by traversing using the edges, completes a cycle that

[1]http://puzzlemuseum.com/month/picm02/200207icosian.htm.

visits all the vertices only once and finally returns to the starting one. A sequence that starts in 1 and ends in 20 would then be a solution if any pair of numbered pegs that differ in only one are adjacent to each other, connected by an edge of the dodecahedron.

Hamilton sold the rights to the London company of John Jacques and Sons, a wholesale dealer in games, for 25 Pounds. Unfortunately, the buyers soon discovered that this game turned to be "a complete sales flop, mainly because it was too easy, even for children".[2] However, on different types of instances (i.e., in other graphs, not the ones suggested by Hamilton) we know that the problem is indeed computationally hard (and, consequently, also considered more challenging to humans). We know this problem as one instance of the *Hamiltonian Cycle* and we know that it is NP-Complete in an undirected graph [156]. It is then in the same computational complexity class as the k-FEATURE SET problem that we have studied in the first chapter of this collection of works. The original instructions that came with the board included 15 puzzles that could be proposed to a contrincante. If Hamilton or the company would have exploited the rich set of different instances of this problem, which they could have marketed using different boards, the story would have been completely different. Still, the Hamiltonian Cycle remains as an excellent introduction of discrete mathematics for primary school students and it is used in many classroom projects around the world to introduce them into the world of computer science and discrete applied mathematics.

7.1.3 The Efficient Design of Electricity Networks

A *spanning tree* of a graph is a subgraph that contains all the same vertices but it has no cycles. By this definition, a graph can contain many spanning trees. If there is a weight associated with each of the edges of the graph, among the many spanning trees it may have, one or more have the minimum total sum of weights of the edges. This subgraph is called the *minimum spanning tree* of a graph.

We have seen in Chap. 3 that the computation of a minimum spanning tree (MST) is the basis of a good clustering algorithm called MST-kNN. In Chap. 4 that, for some problems, building the MST is useful as it is a *proximity graph* and some properties are associated with it. Later in Chap. 5 we have seen that the MST-kNN algorithm is useful to cluster a group of people across their responses on a survey. The importance of the MST is immense; a query of "minimum spanning tree" in Google Search brings nearly 57,000 hits. We now provide some information about the history of some of algorithms for this problem and an account of when did it first originate.

In 1926, Otakar Borůvka published a paper motivated by a question posed by his friend Jindrich Saxel, who worked in an electrical utility during World War I, that of designing an efficient electricity network connecting a set of points if you have

[2]http://www.daviddarling.info/encyclopedia/I/Icosian_Game.html.

a matrix of distances between them. A translation of his pioneering works can be found in [229]. His clever solution, in the form of an algorithm, remains practical even today. Either implementations of his algorithm or other hybrid algorithms are still based on his ideas.

In the 1960s this problem regained interest. The work of two other researchers, Prim [257] and Kruskal [165], who cited Borůvka, sparked an interest for alternative algorithms which could be more useful when the structure of the graph would be different. Today, computer science students around the world studied their algorithms and hybrids of them as well as implementations that exploit the particular characteristics of new hardware platforms (e.g., GPUs). Historically, other researchers working on this problem had been less fortunate in obtaining their deserved recognition and [233] is very helpful at providing a survey that helps to close this knowledge gap.

7.1.4 Runaway Girls: One of the First Social Network Studies

Let us now fast-track from the Victorian Era and from the early 1920s to the time of The Great Depression, in upstate New York in search for an another pioneering story on the use of graphs and networks to understand social phenomena. In 1932, fourteen girls escaped from the New York Training School of Girls which was a reformatory institution for delinquent girls under the age of 16. Since four runaway incidents happened in a period of just 2 weeks (totalling 14 girls), this triggered a serious concern. The number seemed to be even 30 times above what could be expected on average. As a comparison, only three runaway incidents occurred in the whole of 1931 and none in 1933. Jacob L. Moreno (a psychiatrist) and Helen Jennings (a psychologist) studied the case. Their approach was named *Sociometry*. They reasoned that the individual girls' motivations and their personalities may have had a lesser role in the events in comparison with the actual girls positions in their network of social contacts.

Moreno and Jennings collected information and mapped this social network [82]. More than 500 girls were examined in terms of intelligence, social activities, and mutual feelings. They used arcs and edges to represent some interactions. Finally, Moreno presented his results and ideas in a form of a book [214] which is available online on the public domain from the Internet Archive.[3] A graph of his mapped social network of 435 individuals (Map III, page 242 of [214]) shows the meticulous work conducted by using the data collected with Jennings in their attempt to understand the events as "runaway chains", paths in the graphs of the social network that represents their inter-relationships.

Moreno's work is also important for another motive; it also anticipates the need of *graph layout* and *graph visualization* which are important areas of research and vital for business and customer analytics. It is really mesmerizing to see the meticulous approach to visualization of information that Moreno showed in his

[3]https://archive.org/details/whoshallsurviven00jlmo.

book. We invite the reader to check the book online. Some drawings (i.e., Map II, page 240 of [214]) show a skill for layout of networks and a quality which is only comparable to what we have on modern automated solutions of graph editors like *yEd*[4] or *Graphviz*.[5] Graph and Network visualization are an important aid in communicating findings. Another great example of that time was Henry Charles Beck's London Underground Tube map of 1931. A number of very interesting mathematical problems [65] originated from these needs and excellent surveys of the state of the art would help to catch up with the existing literature [104, 127, 202]. We also invite the reader to see the discussions of early approaches to use graphs as a representation of *social networks* in [93] written by one of the pioneers of the area.

7.1.5 The Long Lasting Legacy of the Pioneers

From Euler until now, scientists collectively developed a theory about graphs and networks that permeates everything in computing and applied mathematics. Including these three above, problems like the Lights Out Puzzle, the Chinese Postman Problem, the Knights Tour, and the Road Colouring Problem, and the Minimum Spanning Tree problem can be reviewed from [20]. Advances in computing and data science are giving "new bottles" to these "old wines". Each one of them has variants and generalizations that occur in many aspects of science and technology.

7.2 Hypergraphs, Multinetworks, and Supernetworks

In addition to graphs, we believe that other mathematical structures are going to be highly influential in the near future in the study of integrated systems in business intelligence and consumer analytics. Contrary to the case of simple graphs, of which we it would already be an impossible task to review its literature in a single chapter, we have been able to cover a fraction of the activity in the emerging area of *hypergraphs*, *multinetworks*, and *supernetworks* relatively well. In this section we provide the definitions and links to available works in the literature.

7.2.1 Supernetworks

While the use of the term *supernetwork* [225] is not new in Operations Research and Management Science [181, 188, 192, 226, 318, 322] it is not often discussed in other areas of business data analytics. In some computer science applications,

[4]https://www.yworks.com/products/yed.

[5]www.graphviz.org.

the concept of "supernetworks" also appeared as a result of studying problems in which it is required to provide a switching infrastructure for many shared-memory multiprocessor systems and telecommunication networks [118] and its study is gaining strong momentum [190]. We are employing it here following Prof. Nagurney's work and use of the term: the conceptual formalization in the study of decision-making problems in integrated networks [222]:

> Supernetworks may be thought of as networks that are above and beyond existing networks, which consist of nodes, links, and flows, with nodes corresponding to locations in space, links to connections in the form of roads, cables, etc., and flows to vehicles, data, etc. In addition, supernetworks integrate existing unimodal network systems by providing a structure above and beyond the component networks. Supernetworks are conceptual in scope, graphical in perspective, and, with the accompanying theory, predictive in nature.

It is then clear that problems involving supernetworks are going to be studied more as more data sources can be integrated. For instance, in [342], the authors propose that the studies of robustness of incubating firms' social networks are lacking the necessary heterogeneity that is required to model a real scenario. The authors then propose that a supernetwork-based approach, with a knowledge interaction model among incubating firms, can facilitate addressing this need. In this case, they considered that both the social network relationships and the knowledge network relationships have to be taken into consideration in the search of better models.

There is also a natural trend, given by the integration of services, to study new optimization problems involving supernetworks. With more data available, new personalized services can be developed using all the available data. For instance, the Australian company *Rome2Rio*[6] combines information on air travel, train, bus, ferry, and automobile to provide online queries for point-to-point travel. In the academic area, in [182], the authors studied how to construct personalized transportation networks that benefit a particular activity program of a single individual. They present several examples that aim to show that a transport network (public) and a vehicle network (private) can represent attributes of individuals which can in turn be applied in problems that analyse the synchronization of land-use and transportation systems. This initial work was later extended with the incorporation of uncertainties, and heuristic proposals for dealing with the associated problems in [180, 183, 188, 189].

Business problems in the area of *supply chains* also naturally lead to the consideration of "supernetworks" since we need to integrate networks of very different types and a joint analysis of the optimal combined requests of manufacturers (which may be even in different countries), retailers (which can be physical, virtual, or both), and consumers (who need to be purchasing with different currencies and conditions according to their country of location). All these aspects need to be considered at the same time [223, 224, 315, 321].

[6]https://www.rome2rio.com/.

7.2.2 Multilayer Networks

Apart from the standard approach to define supernetworks as discussed above, either directed or undirected, and either weighted or unweighted, specific kinds of networks exist with structures that present additional characteristics. Among them, multilayer networks and/or layered graphs stand out as the focus of current research in computer science [29, 162].

In multilayer networks, nodes and edges can be found arranged in layers, with intra-layer links connecting nodes in the same layer, and inter-layer links establishing connections between nodes in different layers. If the layers are defined due to the existence of nodes of several categories, what we obtain is a network of networks, also known as an *interdependent network* [7, 58, 106, 112]. For example, in Fig. 7.1, the Internet network depends on the power network since computers need electrical supply, and at the same time the control of the power grid is performed using computers. Here, there are at least two kinds of nodes, computers and power generation stations, but others can be added, such as transformers, circuit breakers, and switches.

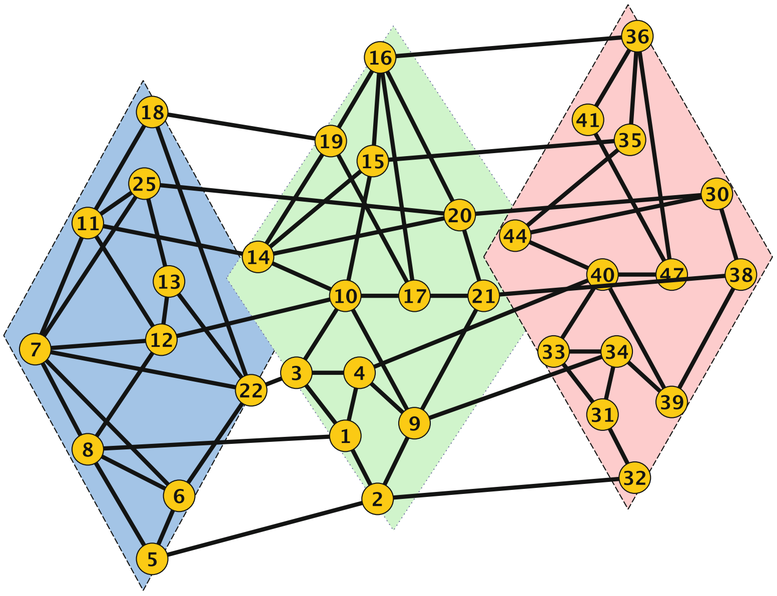


Fig. 7.1 Example of a layered graph. The nodes in the first layer are connected with nodes in the second layer, but not in any other layer (the same for nodes in the third layer, they are only connected to nodes in the second layer). The existence of structures like this one in a graph may allow that polynomial-time algorithms can be designed for NP-Complete problems. We then say that instances of this type are special cases of a problem (e.g., the problem restricted to instances defined on “layered graphs”) and, in some particular cases, they can be solved with efficient algorithms while the problem remains intractable (e.g., NP-Complete) for general graphs

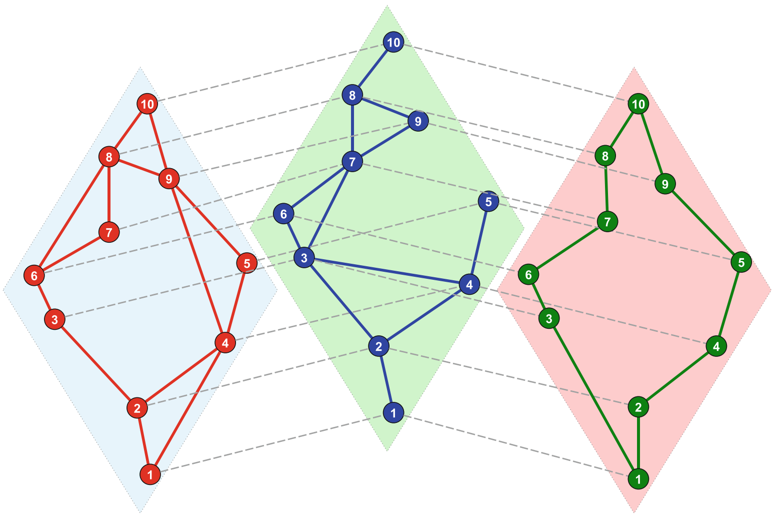


Fig. 7.2 In this case, the nodes in the different layers are representing the same objects. The red edges may be part of a transportation network (i.e., physical roads between the nodes), the ones in blue an electricity, and in green the shortest tour that can be defined on those set of nodes "as the crow flies", i.e., the Euclidean distance between the nodes

Another important class of multilayer networks is obtained when all the nodes are of the same type, but the links represent different kinds of interaction, thus being convenient to separate them in layers (such as the one in Fig. 7.2). In this case, it is common to find that the same node appears in different layers (the one in Fig. 7.2 all nodes appear in all layers). For example, in social networks, the same person connects to others through different types of relationship (family, work, friendship), or different online social platforms (Facebook, WeChat, Twitter, WhatsApp, etc.). Similarly, in transportation networks, the nodes could be the locations of the city in which there are bus stops, train stations, or taxi stands, and the edges are given by the lines of each of the transportation means. These kinds of multilayer networks receive different names, depending on the connectivity between layers [162]: *edge-coloured networks* (or *edge-coloured graphs* [83, 133, 134]), when there are no inter-layer links; *multiplex networks* [80, 139, 153, 290, 314], if inter-layer edges can connect only different instances of the same node in different layers; and *interconnected multilayer networks* [240, 289], in the general case in which inter-layer links can connect any pair of nodes. Figure 7.3 depicts an example of a three-coloured edge graph produced from the graph of Fig. 7.2.

There are several challenges put forward in the understanding of these types of networks [126] and some computational gains as well. Since many artificial neural network models exploit the benefits of basically adapting a layered graph structure (see the discussion on "piling things up" to build these systems in Chap. 1), so

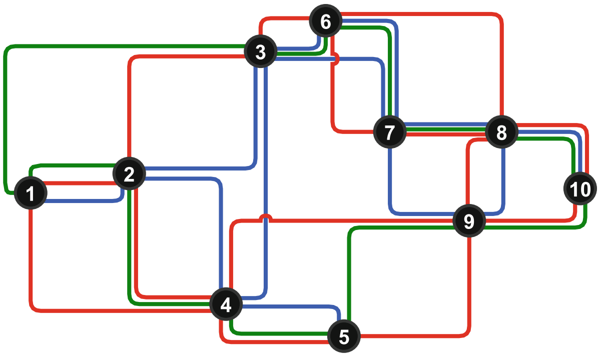


Fig. 7.3 The same structure of Fig. 7.2 but now on a single layer and employing different colours for the edges (according to the different layer they originated from). We are perceptually more accustomed to this representation due to its similarity with some metro maps which indicate different transport possibilities on a unique layer

we expect a surge of interest in problems defined on these types of mathematical structures and the design of new algorithms for their solution.

7.2.3 Hypergraphs

Some problems are naturally defined on a mathematical structure known as a *hypergraph* (we will see some examples in Sect. 7.7.4). In this sense, hypergraphs are both an "inevitable consequence" of a generalization of the concept of graphs and "natural need" in modelling some computational problems. The mathematical definition is straightforward. In an undirected graph $G(V, E)$ any edge of the set of edges of G connects a pair of vertices in V. In a hypergraph this is generalized, we can now join a larger number of vertices. Formally speaking, we say that a hypergraph is a pair of sets, written $H(V, X)$ where V is to be a set of elements called nodes or vertices and X now represents a set of non-empty subsets of V (the *hyperedges* of H).

This definition does not imply that the hyperedges of a given hypergraph should all connect the same number of vertices. Those that do are called k-uniform hypergraphs (e.g., a simple undirected graph is a 2-uniform hypergraph).

Analogously, there is a natural generalization of directed graphs. In a directed hypergraph $D(V, X)$ there is a set of hyperarcs (X); where in this case X is a set of ordered pairs (T, H) of disjoint subsets of a set of vertices V. For each hyperarc a in the set X the associated $T(a)$ is the "tail set" of the hyper arc a and the set $H(a)$ is called the "head set".

7.2.4 Hypergraphs in Business Analytics and Marketing

There are many applications of hypergraphs [5, 36, 39, 206, 213, 265, 302, 308] and directed hypergraphs [14, 46, 78, 100, 168, 205, 266, 287], as modelling data structures for problems in a large variety of fields. In Table 7.1 we give a brief survey of undirected hypergraphs and current applications in the business and Marketing area.

Table 7.1 Hypergraphs in business and marketing applications

Application area and paper	Method	Problem domain
Customer analytics [157]	Hypergraph kernel-based classification	Customer rating prediction
Customer analytics [67]	Optimal and heuristic solutions	Revenue-maximizing bundling configuration
Social networks [319]	Hypergraph indexing	Context-aware nearest-neighbour query
Social networks [172]	Hypergraph model	Link prediction
Social networks [172]	Influence maximization	Cost-aware target viral marketing
Social networks [152]	Hypergraph-based algorithm	Diffusion centrality model
Social-activity networks [9]	Hypergraph modelling	Multimedia social networks
Multimedia networks [344]	Influence centrality via random walks	Influential set identification
Multimedia networks [8]	Hypergraph data modelling	Topic ranking
Computational advertising [125]	Lagrangian decomposition	Allocating marketing channels
Recommending systems [304]	Hypergraphs and group sparsity	Music recommendation
Recommending systems [173]	Hypergraph learning	News recommendation
Business analytics [228]	Graph transformation rules	Modelling business processes variants
Business interconnectivity [84]	Hypergraph modelling	Business intelligence architectures interconnection
Business configurations [219]	Symbolic graphs	Semantics of service modelling languages
Business processes [256]	Hypergraph-based modelling	Semantics of flexible processes
Business networks [317]	Models and algorithms	Enterprise relationship network
Financial engineering [336]	Equity analyst hypergraph partitioning	Crowdsourced stock clustering
Data analytics [177]	Hypergraph-based spectral clustering	Clustering categorical data
Data analytics [284]	Hypergraph query system	Higher-order conjunctive relationships in multidimensional data

(continued)

Table 7.1 (continued)

Application area and paper	Method	Problem domain
Data analytics [298]	Hypergraph partitioning	Combining multiple partitions in a cluster ensemble
Data processing [174]	Hypergraph partitioning	Image summarization
Data processing [191]	Hypergraph spectral hashing	Image retrieval with heterogeneous social contexts
Data security [179]	Hypergraph-based data publishing	Identity disclosure control
Data security [296]	Minimal transversal of hypegraphs/integer linear programming	Frequent itemset hiding in transactional databases
Cybersecurity [137]	Hypergraph decomposition	Colluder detection
Cybersecurity [178]	Hypergraph anonymization	Private service customer data publishing
Knowledge engineering [278]	Hypergraph-based search	Wikipedia search with semantics
Knowledge engineering [297]	Formal concept analysis	Automated reasoning with data
Knowledge engineering [4]	Fuzzy partition	Intuitionistic fuzzy logic
Software engineering [175]	Hypergraph grammars	Software architectures reliability
Software engineering [97]	Identification of functional dependencies through refutations and duality of hypergraphs	Relational model from legacy systems
Software-as-a-service [97]	Extended dependency-aware hierarchical service model based on directed hypergraphs	Service dependency resolving
Cloud management [329]	Service clustering based on hypergraph partitioning	Cloud manufacturing environment
Modelling and verification [331]	Hypergraph grammars	Dynamic evolution of software architectures
Scheduling [286]	Modelling and approximation algorithms	Minimum length link scheduling in multiuser MIMO networks

7.3 Identifying Structures: Why is This Important?

There are several reasons why identifying the "structure" of a graph would be relevant. However, before looking at the different perspectives of what this means let us stop for a second and reconsider what we understand by the word "structure". It derives from Latin's *"struere"* (to build). In the English language, it is often understood as "an arrangement of, and the relations between the parts of elements of something that it is considered more complex". Two important messages can be derived. One is that some structures are indeed "naturally occurring" so our task is to understand how they come to be what they are. Decomposing them into "simpler" elements is then part of the reductionist approach that can help us explain how they come to existence. The other message is that structure can help us to understand their influence in a phenomenon or take advantage of it for a particular purpose. For instance, given two social networks of face-to-face interactions, having the same number of people, we expect an epidemic to propagate faster in the more dense network (the one having a larger number of people-to-people contacts). However, when both networks have the same density, other parameters of the network may be more important and it is our task as scientists to identify them.

An example of a study that involved social networks and how they can influence behaviour, the Framingham Heart Study gave a spin-off that would have made Moreno and Jennings open a big smile. The Framingham Heart Study is a famous public health study that started in 1948. It was designed to be a long-term study on residents of a town (Framingham, Massachusetts, USA). It is now in its third generation of participants and more than 1000 research papers in aspects of human health have been published based on it. Originally, it was designed to study cardiovascular function. More recently, participants gave contact information over the last 30 years which has helped research to map their social networks. Interestingly, using information of 12,067 individuals assessed repeatedly from 1971 to 2003, two researchers have been able to put some assumptions regarding the relationship between obesity and the participants' social networks to the test [51]. Moreno and Jennings may have predicted the results. Friendship relationships seem to have a higher influence in the observed obesity trend between pairs of individuals than familiar or marital ties. Also, "pairs of friends and siblings of the same sex appeared to have more influence on the weight gain of each other than did pairs of friends and siblings of the opposite sex." Christakis and Fowler also conclude that: "The spread of obesity in social networks appears to be a factor in the obesity epidemic. Yet the relevance of social influence also suggests that *it may be possible to harness this same force to slow the spread of obesity*. Network phenomena might be exploited to spread positive health behaviours, in part because people's perceptions of their own risk of illness may depend on the people around them" [51].

It is then clear, as perhaps it was already hypothesized by Moreno and Jennings, the friendship social network has a strong relevance if we aim to understand some behaviours. Marketing is now trying to adapt to this new world and benefit from these studies. They are providing well-controlled studies of large magnitude which are pioneering new ways to understand population behaviour.

7.3.1 Viral Marketing is Based on Networks

Understanding the structure of a network helps to produce mathematical models that can explain some phenomena [16, 241]. An example on the positive side would be to see obesity as an "epidemic process" on a network of friendships and relationships which can lead to some sort of public health interesting new government-guided health policies for intervention and control.

There are other areas in which the increased engagement of individuals can be powered by marketing and the existence of a network on which ideas and behaviours can propagate. Welcome to the new area of 'viral marketing' [16, 128, 176, 270, 299]. It is essential to understand these mechanisms in a quest to deliver for the common good. Ideas from dynamical systems [130, 169, 241, 337] and mathematical methods of epidemiology may turn suddenly very useful [109, 170]. A number of researchers are thus dedicating their efforts to using social networks to promote "smart city systems" and to discover future needs of residents in order to manage the city resources more efficiently [154], as well as selling digital goods with near zero marginal cost [262].

As expected, a number of phenomena in the field of "viral marketing" have similarities and differences with those present in the study of epidemics on social networks [109, 273, 274, 328]. It is the purpose of the marketer to propagate the messages as widely as possible, constrained to budget and time requirements. Identification of "influencers" [154, 234], the role of "celebrities" [195], is then linked to studies on propagation [345] and how to "seed" information in these networks [136]. "Influence Maximization" is the key factor for advertising professionals in this area [49, 154, 193, 195, 210, 211, 227, 255, 262, 299, 345].

7.3.2 Identification of Structures via NP-Hard Problems

There are other types of structures sometimes discussed in the context of business analytics. Due to the domain, these structures are known (e.g., by its own definition, we can assume the presence of layers if somebody gives us a multilayer network). Alternatively, the graph may have some known topology (e.g., bipartite graphs) or we may know that subgraphs with expected characteristics do not exist on them [107]. Sometimes, these known facts may allow us to have a problem that is NP-hard defined on a general graph become solvable in polynomial-time on particular architecture. This said, if an efficient algorithm exists for it, that should be a plus. There are circumstances, however, in which we do not have control of our input and we have to assume it is of a general type. As discussed in the first chapter, we can devise a number of techniques for those problems including, of course, exact algorithms, but also approximation algorithms, heuristics, and metaheuristics. Since there are so many NP-hard problems of interest defined on graphs, we review in this section some of the basic concepts and some approaches.

There is a folkloric saying among computer scientists: "most interesting problems are NP-hard". This means they consider the problems which are NP-Complete "or harder" than the ones where the real challenges are. This is well established in business analytics, but why should it be any different in business or marketing? We have seen in the first chapter of this collection, problems which were NP-Complete (like the k-FEATURE SET problem, which is a basic problem in artificial intelligence and machine learning). We have also seen the l-PATTERN IDENTIFICATION problem and the k-VERTEX SET Problem. The reader may then now be aware of the importance of classifying the complexity of the computational problems.

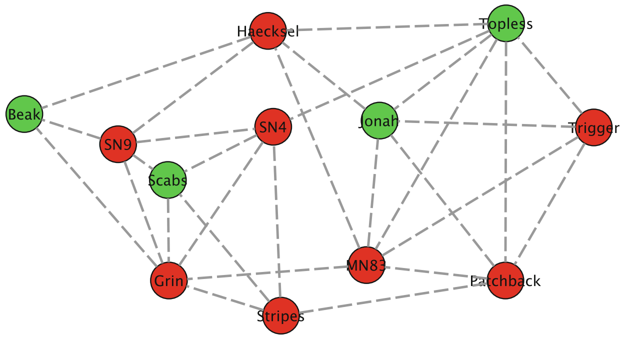
In fact, the marketing and customer analytics practitioners are starting to recognize that NP-hard problems are closer, as a mathematical model, to the real-world scenarios they need to address. Consequently, NP-Completeness and NP-hardness problems are reigning rampant in marketing and the new challenges presented by them have started to attract attention of computer scientists. For instance, researchers are starting to address some of these NP-hard problems in the booming area of viral marketing [12, 159, 193, 195, 234, 255], just to cite one example closely related to the core subject of this chapter.

There are cases, however, in which there is a question of interest that can be solved with an efficient algorithm (i.e., those that in the worst-case still run in a time which is a polynomial in the size of the instance). The quest for computer scientists is then to find the most efficient algorithm for that problem. Those problems are also very interesting. In other cases, an efficient algorithm can answer whether a problem, which is NP-Complete for a generic type of instances/inputs, can be solved in polynomial time for a particular instance given (or for a class of instances that may share a property). The set of all these instances solvable in polynomial-time becomes “a special case” of the more general problem and searching for those cases is also rewarding. For networks having billions or trillions of objects, sheer size is one clear obstacle and mathematical modelling should be guided by the principle that efficient algorithms and/or heuristics would be required for any practical purpose.

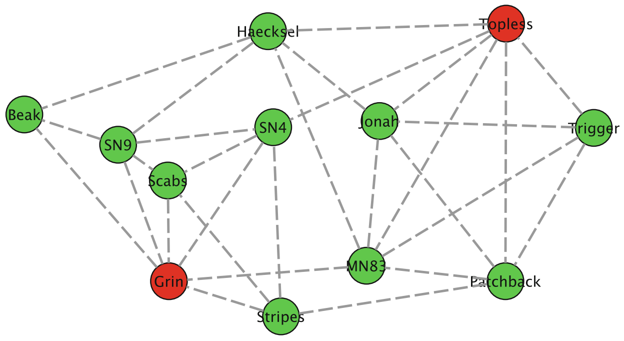
7.3.3 Three NP-Hard Problems on a Dolphins’ Social Network

The study of graph structure may be related to the solution of some problem that has a decision version that is NP-Complete. We will discuss three graph problems in which the optimal solution found uncovers some structure. We will use an unusual example of an interaction network. To illustrate the main characteristics of these three problems we will use data from a subpopulation of dolphins from a community totalling 62 individuals that were living off Doubtful Sound, New Zealand. The data was compiled by Lusseau et al. in [196] and it is generally used in the literature of another interesting approach; the identification of community structure of networks. We will see this concept later in the chapter. In Fig. 7.4, keeping the same layout for the nodes and edges helps to illustrate on feasible solutions for the three problems. In all of them a node represents a dolphin and an edge represents a frequently observed association that exists between a pair of animals [196]. Figure 7.4 shows, in red, the results of the identification of a *vertex cover* (Fig. 7.4a) and an *independent dominating set* (Fig. 7.4b) in this interaction graph. Finally, a valid solution for the *clique cover* problem is shown in Fig. 7.4c.

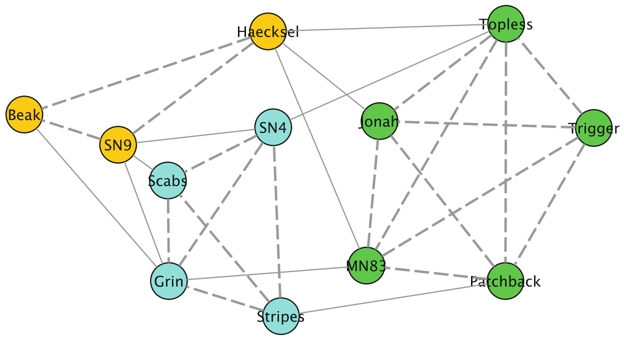
In the minimum dominating set problem, we seek to find a set of nodes of minimum cardinality, such that those that have not been selected have at least one



(a)



(b)



(c)

Fig. 7.4 Feasible solutions of three graph optimization problems give different insights on the structure of a network of mapped interactions in a community of dolphins. (**a**) A vertex cover (red vertices). (**b**) In the same graph, an independent dominating set with only two vertices. (**c**) A clique cover. Each node belongs to a clique, i.e., a completely connected subgraph. The edges belonging to the three cliques are shown in dotted lines. Clique membership is highlighted using different colours

edge in the graph that connects to a node that has been selected. In social network parlance, if we want to be "one friendship away" from all individuals in the network, propagating an interesting advertisement to the nodes in the dominating set would give some "kickstart" to our campaign. From a quick inspection of Fig. 7.4b it is clear that there is no 1-dominating set thus the shown solution is optimal (all nodes are connected to either the dolphins "Topless" or "Grin"). In graph theory terms, this pair of nodes is also called an *independent set* (of nodes) since there is no edge that connects a pair of nodes in the set. We should mention that deciding if a graph has a set of vertices which are independent and has a cardinality equal or greater than a given number $k \geq 2$ is also one of those "six essential" NP-Complete problems we were mentioning before. This is a problem that had attracted the attention of many researchers over several decades as it seems to be central to solve other problems related to it [43, 186].

Variants of the dominating set and the independent dominating set problem can be useful to establish dynamical seeding strategies for viral marketing [136] trying to maximize both spread and revenue [159, 234, 301].

In the first chapter of this collection of works the *vertex cover* problem was described; we revisit its definition here. A vertex cover of a graph $G(V, E)$ is a subset of the vertices of G ($S \subseteq V$) such that every edge in E connecting vertices a and b has either $a \in S$ or $b \in S$ (or both). A vertex cover S is said to be *minimal* if by deletion of any single vertex of S the result is a set that is not a vertex cover. Figure 7.4a shows a vertex cover in this social network. It is, however, left to the reader (as a challenge) to check if this solution is minimal and if it is the one with minimum cardinality or not. This problem has a great number of applications and is also consider one of the *"six essential"* NP-Complete problems [208].

The input of the k-CLIQUE COVER is a graph $G(V, E)$ and the question is to decide if a set of k cliques exist in the graph such that they are a partition of the set of vertices of the graph. In Fig. 7.4c we see that the graph vertices can be partitioned into three sets, and the vertices in each set are all connected between them (constituting a clique). Then the answer for the decision problem 3-CLIQUE COVER problem for this graph is "Yes". The k-CLIQUE COVER is one way to reveal the "community structure" of a network, a concept that will be discussed again in Sects. 7.4 and 7.3.6. The k-CLIQUE COVER is also NP-Complete [156].

7.3.4 Seeding a Viral Marketing Strategy via Covering, Dominating, and/or Independent Sets

Using a vertex cover to identify users of the network on which we could "seed" a message (for our viral marketing strategy) could be interesting. However, we should expect that this number should be really big, and we should really expect that no company can afford the advertising cost. Propagation on the network and word-of-mouth is required to diminish the costs. In fact, if we do not have any isolated node in our social network (an unlikely poor soul with no connections), then every vertex cover is also a dominating set. This means that the smallest dominating set cannot be bigger than the smallest vertex cover. Thus the former is probably closer to what you would like to compute if you want to propagate a message through a network with a "one-step-guarantee" that everybody who did not get the message directly from you at least has one connection who received it.

Very recently, Lamm et al. presented an approach that has all the characteristics of a memetic algorithm for the maximum independent set (this collection of works has a whole section dedicated to memetic algorithms as well as other chapters in this section which use the technique). The scalability of the approach proposed by Lamm et al. indicates that it may be possible, at least algorithmically, to compute them for instances like the ones you may need to deal within viral marketing in real-world social networks [166].

For realistic social networks, even a dominating set solution would not be feasible in terms of advertising costs (unless, of course, you own the social network). Another interesting mathematical model for seeding a message comes from the study of another structural parameter of graph, the *total k-domination number* ($\gamma_k^t(G)$). Given a graph $G(V, E)$, a subset $V' \subseteq V$ is called a *total k-dominating set* if every vertex in V is within distance k from some vertex of V' other than itself [119]. The distance between a pair of vertices $\{a, b\}$ in a graph is defined as the minimum number of vertices that need to be visited to reach from the other in the shortest path between them. Consequently, given a graph G, finding the values of $\gamma_k^t(G)$ for $k \geq 1$ can give important information on its structure as well as potential individuals for seeding a marketing campaign that might increase its chances of wide propagation.

7.3.5 *Sometimes You Can Guarantee Something: The Approximability of* MIN VERTEX COVER

Given a graph we can construct a cover and a minimal cover in polynomial-time, but how hard is it to find the cover of *minimum* cardinality? Since the MIN VERTEX COVER problem is NP-hard, our hopes are far remote on solving the problem in time bounded by a polynomial-time function of the size of the instance. However, sometimes it is possible to have a polynomial-time algorithm which has a worst-case analysis on its performance. This would allow us to identify "how far" we are from the optimum result (again, in the worst case situation). In the first chapter *approximation algorithms* were introduced as an alternative to *heuristics* and *metaheuristics*. The following result is an approximation algorithm due to Fanica Gavril who proved this result in 1974 (and who reportedly shared it in a private communication to David S. Johnson). This result is now a textbook example all over the world. It is also said to be independently discovered by Mihalis Yannakakis. We discuss it here because it is a constructive method that delivers the essence of what it is called a 2-*approximation algorithm* for this problem.

The algorithm is said to be *randomized* since it starts with a randomly chosen edge, then includes the endpoints vertices of the edge in S. It helps to think that we are actually selecting the endpoints of M, a set of edges, and at each iteration an edge is selected and removed, and that all adjacent edges are also removed (as they are covered by one of the two vertices added to S). Iterating this basic randomized procedure until no edge remains, we finally end up with a set S, which is a vertex cover, and its cardinality is exactly twice the number of edges that have been chosen. Note, however, that none of the chosen edges has a vertex in common (by the way they were chosen), so this set constitutes what is called a *maximal matching* in the graph G. We now note that *any* vertex cover *including the optimum one* must contain at least one vertex from each of these chosen edges (since otherwise that particular edge in the set M would not be covered). That being the case, if $|S_{opt}|$ is the cardinality of the optimum vertex cover then if $|M| \leq |S_{opt}|$. The randomized algorithm returns a vertex cover of size $|S| = 2|M|$. Since $2|M| \leq 2|S_{opt}|$, then we can write

Algorithm 1: 2-approx-algo-for-MIN-VERTEX-COVER

```
  Input: $G(V, E)$
  Output: $S$
1 $S = \emptyset$
  /* the vertex cover is initially empty */
2 while there is an edge left in $G$ do
3     choose any edge $(u, v)$
4     add $u$ and $v$ to $S$, and delete them from $G$
5 end while
6 return $S$
```

$$\frac{|S|}{|S_{opt}|} \leq 2$$

meaning that we have here one algorithm that *always* returns a solution which is feasible (i.e., a guaranteed vertex cover) and such that its cardinality is at most twice the one of the optimal one. That is why it is called a 2-approximation algorithm for this problem. Because the number of individual steps that the algorithm requires is bound by a polynomial function in the size of the graph, since the 1970s, this type of efficient algorithms, with performance guarantees, has attracted the attention of many researchers.

Several different approximability measures have been proposed and the field has been an active area of computer science since the 1970s. Viggo Kann, since the early 1990s, maintains an online compendium of NP-hard optimization problems and their approximability status.[7] It contains other results for the three problems of Sect. 7.3.3 and for others arising in business and network analytics.

[7] https://www.nada.kth.se/~viggo/problemlist/compendium.html.

Results like this one for vertex cover, although the fruit of much more elaborated proofs, indicate that it is possible for many problems in graphs and networks to provide such mathematical bounds. Gonzalez has put together in [113] an important collection of works in which most of the techniques employed are introduced (and now is preparing a second edition). Vazirani in [310] and Williamson in [327] also present approximation algorithms for many graph optimization problems and their associated techniques.

7.3.6 Community Structures

The identification of communities in graphs has many applications, for instance, in consumer/market segmentation [221, 341]. In [151] a community structure study of the itineraries of 16,363 cargo ships during the year of 2007 helped to understand the patterns of global trade and bioinvasion. Applications in social networks [31] and business intelligence [184, 264] are obviously common. Less commonly known are applications in *immunization of networks* [338], *malware detection* [77], and *prehistory* studies [263]. This short list can already be a witness of the wide range of applications.

In general, authors refer to some approaches that identify subgroups of vertices of the graph that are highly connected relative to the sparse connectivity shown to other groups of vertices. There are several approaches in the literature and we cannot properly review here all the literature that exists on the subject. We refer to the surveys of Santo Fortunato [91], the one in directed graphs of Malliaros and Vazirgiannis [198], and the more recent surveys of Fortunato [92], and Khan and Niazi [160].

We briefly describe some of the main strategies/methodologies at the core of the different proposals to identify community structure.

7.3.6.1 Clique-Based Methodologies

We have mentioned that the k-CLIQUE COVER problem is one key approach to reveal groups of densely connected vertices. We refer to [45] for a discussion of such a method on real-world instances.

The Bron-Kerbosh algorithm is a recursive method for identifying all maximal cliques in a given graph $G(V, E)$. It has several versions according to some of its characteristics (i.e., pivoting strategy and vertex ordering). It has a worst-case behaviour of $O(3^{|V|/3})$ [307]. We will see later in the chapter other relaxations (e.g., avoiding computing maximum or maximal cliques) that may lead to problems that can be solved in polynomial-time in the number of vertices, turning them as perhaps a more useful alternative for large networks.

7.3.7 Modularity

Given a graph $G(V, E)$, and a partition of V, the *modularity* is the fraction of the edges in E, that connect vertices that are in different sets of the partition, minus the expected fraction if the edges were to be distributed at random [231]. Modularity has already been defined in this chapter by Eq. (7.3). The problem of deciding if a graph has a partition of its vertices that has a value of modularity above a given value is NP-Complete [34]. Consequently, a number of heuristics have been proposed [6, 41, 110, 111, 220] as well as approximation algorithms [158]. An example of the application of modularity as a score to uncover leads to cartel identification is later discussed in Sect. 7.4.8.

7.3.8 Overlapping Communities

Some mathematical models also allow for *overlapping* communities as we have seen in Sect. 7.4.6. This means that a particular vertex can belong to more than one community. This new approach to define community structure of a graph has proved useful to study Amazon's co-purchasing networks [98]. In that contribution the authors have a "brand-centric" approach, trying to elicitate which are the co-purchasing behaviours that defines "the limits of the brand", as it starts to compete with products which the clients prefer to obtain from other brands [98]. Chapter 9 presents an application of a model based on overlapping communities for the analysis of co-purchasing and social networks.

7.3.9 Identifying Communities via Relaxations

Finding dense subgraphs has been in the work agenda of many researchers over the years, so notably several variants of the classical "maximum clique" approach have been proposed. Let S be a set of vertices. If S is a clique, all vertices should be at distance of 1 of each other and the degree of them should be equal to $|S| - 1$. The following relaxations have been proposed in the past:

- For obtaining a k-*plex* [53, 207, 283, 320, 330], the restriction that the degree of each vertex in the clique should be one $|S| - 1$ is relaxed to be $|S| - k$,
- For obtaining an n-*clique*[212], the restriction that all vertices should be at a distance 1 to each other is relaxed to n.
- For an n-*clan* we have the same as for an n-clique but a restriction is added to the diameter [212],
- The "edge density" of a clique is the maximum possible (all the vertices in S are connected by an edge), this is relaxed in the definition of *quasi-cliques* [2, 37, 50,

187, 243, 247, 294, 309, 311, 343] by allowing to be at least a value $1 \leq \gamma < 0$ with γ being the ratio between the number of edges connecting vertices in S and the possible maximum (i.e., $|S| \times |S - 1|/2$).

Unfortunately, many of these alternatives give rise to NP-hard optimization problems [249, 316], again indicating that heuristics and metaheuristics may be needed for large graphs.

7.3.10 Killing Two Problems with One Stone: Graph Colouring Problem

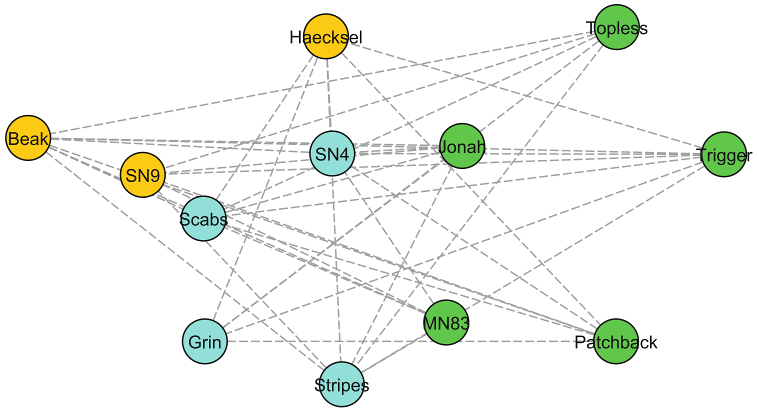
Another interesting structural property of a graph $G(V, E)$ is its *chromatic number*, i.e., the minimum number of colours needed so that each node of the graph is coloured and no two adjacent vertices of G are assigned the same colour [332]. Feasible solutions of these problems are called *proper* colourings. Finding good algorithms for this problem is of great importance in many different fields since it is a core problem in many applications [69, 102, 155, 161, 194, 277, 333, 334, 340]. The problem is the subject of constant investigation, more than 800 papers have been published having the phrase "graph colouring" in its title since 2010 (data from Google Scholar).

We mentioned before that, given a graph, studying structures in the complement graph can potentially be very useful. In fact, a graph has a k-clique cover if and only if its complement can be coloured with k-colours (i.e., no two vertices connected by an edge can have the same colour). In Fig. 7.5 we give an illustrative example. Algorithms for k-COLOURING can be applied for solving the k-CLIQUE COVER by running them on the complement of the graph given as input. This strategy of "fighting the battle elsewhere" can be useful in practice if we know that we have good algorithms and/or heuristics for sparse graph problems.

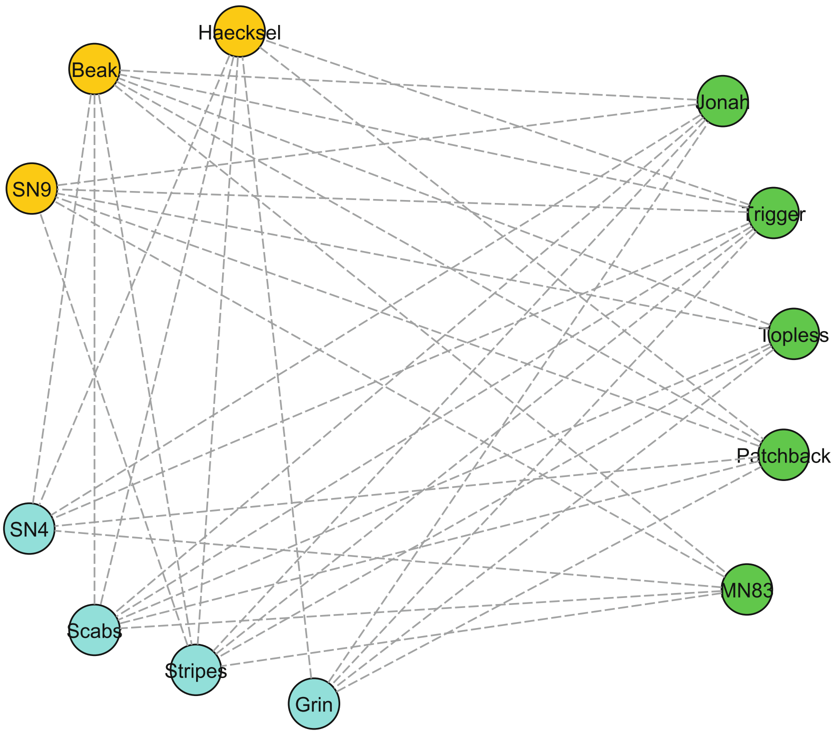
7.3.11 Polynomial-Time Computable Parameters of Graphs

If we think of a large social network (e.g., Facebook) or the co-purchasing network from Amazon, it is clear that understanding something about the structure of the graphs by solving NP-optimization problems and analysing the results of the solutions found present great challenges. Consequently, it is not surprising that some researchers seek other alternatives to identify parameters of interest.

Researchers also try to characterize graphs and networks using polynomial-time algorithms and link them with observed properties of certain phenomena such as the possible type of process that generated the network [163, 164, 242]. In the next subsections we discuss some of the alternatives most commonly used.



(a)



(b)

Fig. 7.5 Solving the Exact Clique Covering problem via graph colouring algorithms. (**a**) The complement graph of the one presented in Fig. 7.4c. The vertices are coloured according to the solution of the Clique Covering problem. (**b**) The same graph as the one of (**a**) but now with a different layout. A solution of the graph colouring problem in the complement graph is a solution of the Clique Cover problem in the original graph

7.3.12 Degree Distributions

The *degree distribution* $P(k)$ of a graph is defined to be a fraction of the vertices of the graph that have degree k. This means that if n is the total number of vertices and n_k is the number of nodes that have degree k then $P(k) = n_k/n$.

This is clearly a very simple measure but it is interesting to note that a number of real-world graphs (such as those from social networks or web connections) have degree distributions that approximately follow a power law behaviour (i.e., $log\ (P\ (k)) = -c_1 log\ (k) + c_2$ where c_1 and c_2 are constants). These graphs are said to be "scale-free". Financial networks, such as the ones originating from interbank payments are said also to be of this type [62, 292]. We refer to [103, 123, 146, 203] for some interesting new results that connect some of the properties of being scale-free with structural parameters and other properties.

7.3.13 Transitivity or Global Clustering Coefficient

The *global clustering coefficient* of a graph, also known as the *graph transitivity*, is defined as the ratio of three times the number of triangles to the number of pairs of adjacent edges in the graph [258].

7.3.14 Average Local Clustering Coefficient

The *average local clustering coefficient* of a graph with n vertices is the average value (over all vertices) of the ratio of $T(v_i)/P_2(v_i)$, where $T(v_i)$ is the number of edges between the neighbours of vertex v_i and $P_2(v_i)$ the number of pairs of neighbours [259]. For a complete graph, both the average and the local clustering coefficients are equal to one.

7.3.15 Assortativity

The *assortativity coefficient* of a graph, as introduced by Newman, is the Pearson correlation of the degree of all pairs of vertices that are connected by an edge in the graph [164]. Another alternative way of assessing the "assortment" of a graph is to compute the average degree of a neighbour of a vertex of degree k. A graph is then assortive if it is an increasing function of k (and alternatively, dissortive if it is a decreasing function).

7.3.16 Girvan-Newman Algorithm for Community Structure

Given a graph $G(V, E)$, this algorithms runs in $O(|E|^2|V|)$ time and helps to identify edges that somewhat "lie between" sets of more densely connected sets of vertices (e.g., the communities). By removing these edges we then produce a partition of V [105]. The method is based on the concept of *edge betweenness*, a centrality measure for edges. This section of the book contains a detailed explanation of the concept of centralities in networks and we refer to it and the original publication [105] for specific details on this popular method.

7.4 Interesting Problems in Networks Originating from Transactional Data

We now turn our attention to a problem that presents an alternative new definition of what a concept of a "community" in a graph represents.

In 2011, Friggeri and Fleury presented a reduction from the classical NP-Complete decision problem in graphs called CLIQUE to a problem they denominated "CONNECTED-COHESIVE". As the latter is in class NP, their reduction implies that CONNECTED-COHESIVE also is NP-Complete [95]. Informally speaking, the aim of these authors is to model the problem of finding a dense subgraph (e.g., a community in a social network) as the problem of finding a subset of the nodes of an undirected weighted graph such that it maximizes a certain objective function that takes into account the number of "inward" and "outward" triangles (i.e., cliques of size 3) in the subgraph induced by this subset of nodes. We refer to [95] for a formal definition of the objective function and [94] for an illustrative example.

While we can be investigating and presenting to our readers other clique relaxation models (e.g., [250, 251]) we are drawn to highlight this new measure since it is based on counting "triangles" which in turn is related to the notions of triadic closure and weak ties introduced in [267] and [101].

Our business and marketing colleagues may ask themselves, why are these triangles so important for computer scientists? While the perspective of researchers studying networks originating from temporal processes should be considered, triangles can be regarded as the most simple building block for the creation of higher-order feedback loops and other more complex structures involving more nodes which are essential for the control of and the dynamical stability of these networks [197].

Due to the NP-hardness of the CONNECTED-COHESIVE problem, in [96] the authors propose a heuristic to optimize the cohesion and apply to the graph of voting agreements between US Senators. We have recently proposed a memetic algorithm approach to deal with this problem in [120]. We will use one of the solutions produced by our method to give some intuitive understanding of what this means in a particular case. Figure 7.6 shows the result of identifying a cohesive-connected group in the *Dolphins Social Network*. This is an undirected graph that represents

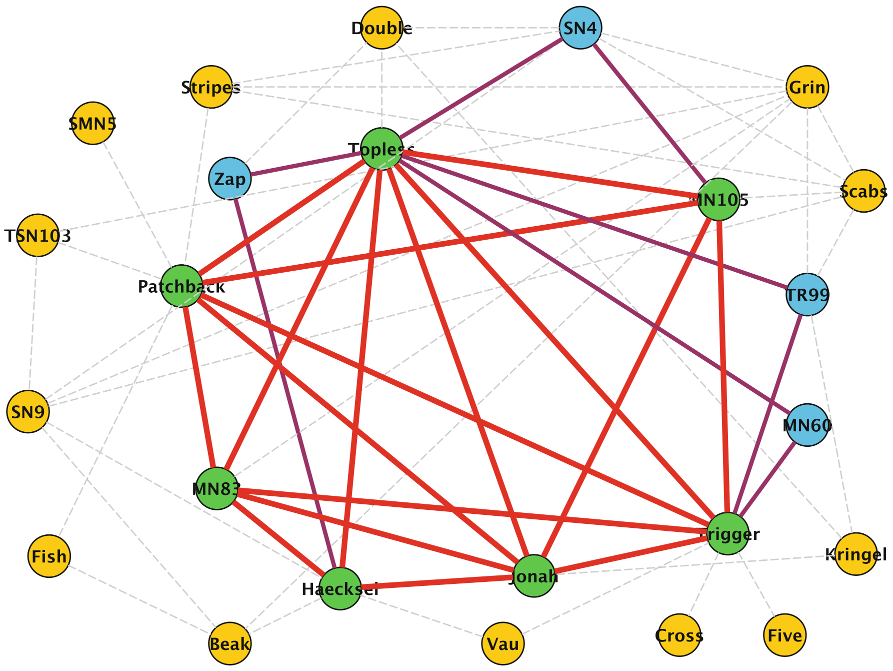


Fig. 7.6 A connected-cohesive group of nodes in the Dolphin's network dataset [1]. We also include the neighbours of other nodes connected to the group. There are 19 inbound triangles (i.e., nineteen 3-cliques) and only 4 outbound triangles: {Topless, Trigger, MN60}, {Topless, Trigger, TR99}, {Topless, Haecksel, Zap} and {Topless, MN105, SN4}. Edges of inbound triangles are coloured in red and we used violet to colour the four outbounding ones. The vertices in the connected-cohesive group are coloured in green, in light blue those belonging to outbound triangles and in yellow the other neighbouring vertices of the connected-cohesive group

the frequent associations observed between 62 dolphins in a community living off Doubtful Sound, New Zealand, as compiled by Lusseau et al. [196] (the whole network will later be displayed in Fig. 7.17). Figure 7.6 shows the largest cohesive-connected community we found using our memetic algorithm. It has seven dolphins (i.e., the group of green vertices in Fig. 7.6). The connected-cohesive group consists of the dolphins named Patchback, Trigger, MN83, MN105, Haecksel, Jonah, and Topless. This analysis reveals a community structure which is not a clique (there are only 17 edges connecting the 7 nodes out of a possible maximum of 21) (Fig. 7.7).

7.4.1 Formal Definition of the Connected-Cohesive Problem

We now give a more formal definition of the problem. We need first to define four mathematical entities.

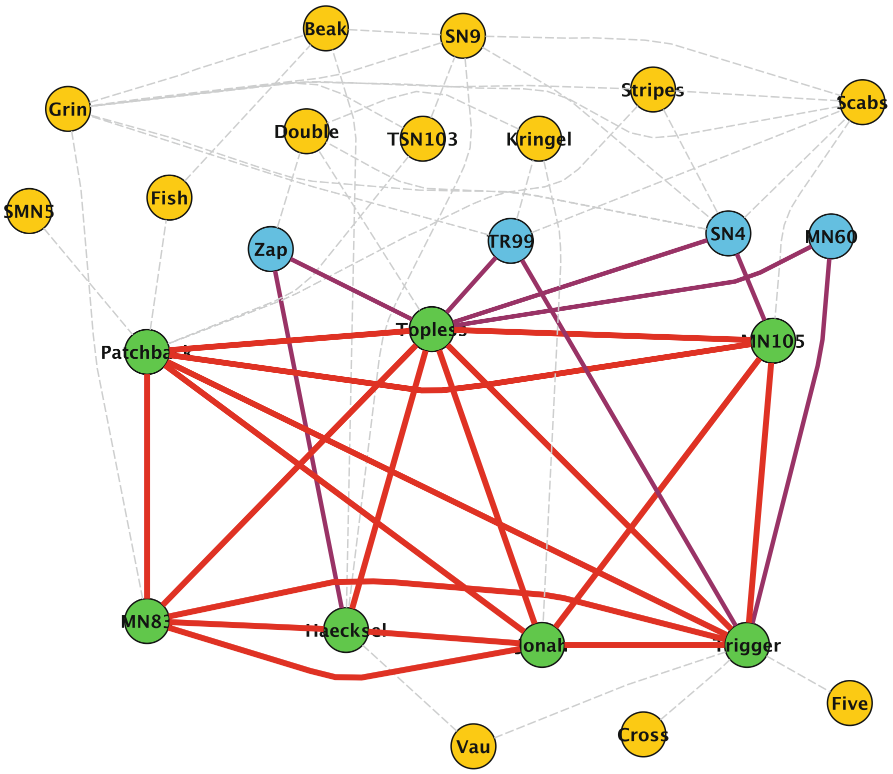


Fig. 7.7 Another layout of the graph previously shown in Fig. 7.6 helps to highlight other aspects of the connected-cohesive group of nodes found in the Dolphin's network dataset [1]. Combinatorial optimization algorithms for identifying structures together with other optimization algorithms used for visualization can reveal important structures. We will discuss this matter later in Sect. 7.4.4

Definition 7.1 (Triangle) Given an undirected graph $G(V, E)$ a *Triangle* (Δ) in G is a triplet $(a, b, c) \in V^3$ of pairwise connected vertices, such that $(ab, bc, ac) \in E^3$ (i.e., a clique of size three).

Definition 7.2 (Triangle Neighbours) The *ΔNeighbours(a)* are the neighbouring vertices $\{v_1, v_2, \cdots, v_m\} \subset V$ of a vertex a, such that for each i, there exists a j where a, v_i, and v_j form a triangle in G.

Definition 7.3 (Inbound Triangle Count) The *Inbound Triangle Count* for $G[S]$ is denoted as $\Delta_i(S) = |\{(a, b, c) \in S^3 : (ab, bc, ac) \in {E_s}^3\}|$; this means that it is the number of triangles in G whose vertices are all in S.

Definition 7.4 (Outbound Triangle Count) The *Outbound Triangle Count* for $G[S]$ is denoted as $\Delta_o(S) = \{(a, b, c), (a, b) \in S^2, c \in V \setminus S : (ab, bc, ac) \in E^3|\}$, that is the count of those triangles in G, which have *exactly two* vertices in S.

Following the definitions, if S denotes a set of vertices in a graph, let $\Delta_i(S)$ be the number of triangles with all vertices in S and let $\Delta_o(S)$ be the number of triangles with exactly two vertices in S, then the *cohesion score* of the set of vertices S, written $C(S)$, is defined as:

$$C(S) = \frac{\Delta_i(S)^2}{\binom{|S|}{3} \times (\Delta_i(S) + \Delta_o(S))} \tag{7.1}$$

With these definitions, we can now state the decision version of the λ-CONNECTED-COHESIVE problem as:

λ-CONNECTED-COHESIVE Problem
Instance: A graph $G = (V, E)$, $\lambda \in \mathbb{Q}$ with $0 \leq \lambda \leq 1$.
Question: Is there a subset $S \subseteq V$ with $C(S) \geq \lambda$?

Algorithms that explicitly use the *cohesion score* (defined in Eq. (7.1)) as a guiding function produce sets $S \subset V$ which were of very small cardinality (sometimes even reducing to a single triangle in some preliminary experiments).

It is known that slight variations of the score/objective function of an original problem (such as the one defined by Friggeri and Fleury) may give practical benefits for metaheuristics such as memetic algorithms [323]. Formally speaking, it is actually a different optimization/decision problem once you propose a change on the objective function, but it seems proper to discuss a variant score that merits further investigation. In practice, this new objective function, the $F(S)$ score first introduced in [347], seems more useful for the quest of finding larger cohesive groups. It is defined as:

$$F(S) = \frac{|S|}{|V|} \times C(S), \tag{7.2}$$

where $S \subseteq V$ in G. Consequently, for the set S in green in Fig. 7.6, the *Cohesion Score* is given by:

$$\begin{aligned} C(S) &= \frac{\Delta_i(S)^2}{\binom{|S|}{3} \times (\Delta_i(S) + \Delta_o(S))} \\ &= \frac{19^2}{\binom{7}{3} \times (19 + 4)} \\ &= 0.44845 \end{aligned}$$

and in this case the $F(S)$ score for this set of vertices is:

$$\begin{aligned} F(S) &= \frac{|S|}{|V|} \times C(S) \\ &= \frac{7}{62} \times 0.44845 \\ &= 0.050631, \end{aligned}$$

noting that the original graph has 62 vertices and the one in Fig. 7.6 just contains the group we have identified and the ones that are connected to them.

7.4.2 *The* CONNECTED-COHESIVE *Problem in the Analysis of Online Shopping Co-Purchasing Data*

We now show how this problem can give new insights in the area of online shopping. Recommending systems benefit from identifying groups of objects that are likely to be co-purchased. There are, however, limits to the number of objects that can be shown to the user as “suggestions” (generally limited to another four or five). Identifying groups objects in which many triads of objects exist that have been co-purchased in pairs may help produce generating these lists of four or five objects to present to the user. An adaptive system may also “remember” previous presentations and avoid repeating certain objects already presented to the users.

With this mind framework, the results of an algorithm for the CONNECTED-COHESIVE problem were obtained to identify groups of highly co-purchased items in an online shopping dataset provided by Amazon. The relevance of this study is quite intuitive. For instance, a company may be interested in analysing the items, from any brand, that are purchased together with other items of a particular brand. A network is then created having a node for each product and an edge if those two products have been purchased together (i.e., in an e-commerce website, these products were “in the virtual shopping cart” and purchased in a single operation, perhaps even with a discount deal for the package). Such a network can then be mined for the existence of these Connected Cohesive structures. The e-commerce company then tends to find products of which there is a high probability that they will be purchased together and could suggest additional products to online shoppers for their virtual cart.

While other probabilistic methods exist, the Connected Cohesive problem, seen as a mathematical model, gives us a combinatorial view in which a large object of interest is “mined from the network”, with the characteristic that a few outbound triangles exist, thus maximizing the chances of recommending products which may “call each other”, expanding the opportunities for selling more than two products.

7.4.3 *Fighting Organized Crime with Nodes, Lines and Finding Triangles?*

Our discussion on the importance of triangles in networks takes us to analyse another interesting possible application Connected Cohesive problem as a mathematical problem: the infiltration of organized crime in the legitimate economy from transactional data.

Although there are nearly two hundred definitions of what organized crime is,[8] the United Nations Convention against Organized Crime (Palermo, Italy, 2000) adopted the following accepted definition:

> a structured group of three or more persons existing for a prolonged period of time and having the aim of committing serious crimes through concerted action by using intimidation, violence, corruption or other means in order to obtain, directly or indirectly, a financial or other material benefit.

Gurciullo argues that this definition is not entirely rigorous, leaving out three important aspects: infiltration into the political system, threat to the stability of the state, and *entrepreneurial planning* [117]. His proposed definition maintains, however, that there should be at least *a group of three or more* implicated in the action. This means that, if there are attempts for organized crime to infiltrate in the legitimate economy, it is natural to assume that at least three individuals should be operating in concert in at least three different commercial firms.

Triangles now come again at the centre of the scene. Three is the minimum number of nodes that could be required to have a necessary feedback loop which may return benefits to one of the nodes of an interacting network (and the number two, while still possible, would be perhaps too obvious, as a pair of firms would look as constantly requiring and providing services to each other). The individual firms may develop other sorts of legal activities and sporadically engage in the activation of one feedback loop, returning financial benefit to a specific "core", while most of the time they would be engaging in other commercial pursuits. This system would then require little centralized coordination and would be preferred to feedback loops that have a larger number of commercial actors and may be harder to control and coordinate.

Gurciullo presents a case study on the city of Porto Empedocle, Italy and uses methods of network analytics to derive some conclusions [117]. We will use one of the networks he has built and which we have obtained from one of the figures of his manuscript. In this case, the graph of our interest has 49 nodes and 93 edges. This is a sparse graph of commercial activities between a set of companies. It is logical to be sparse since each company is in the construction sector of Porto Empedocle and each edge represents one economic transaction that existed between the pair

[8] Klaus von Lampe, Definitions of Organized Crime, Sep. 7, 2017, Available from www.organized-crime.de/organizedcrimedefinitions.htm.

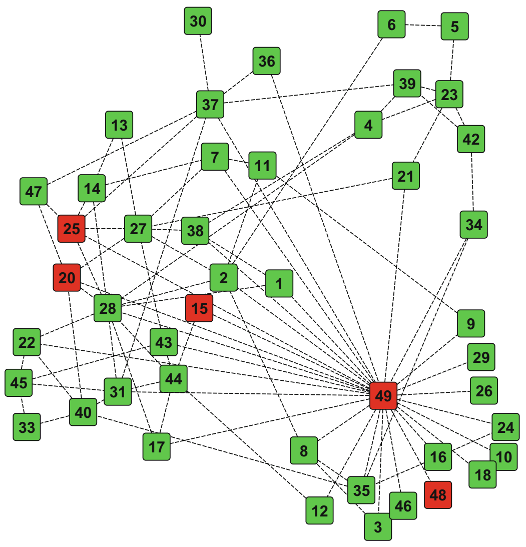
of companies it connects in that year (2002). Giuciullo questions the need of some companies, particularly if they are in the same economy sector, to have so many transactions with others in the same sector. Obviously, some construction firms may specialize in providing services for other construction companies, so having some sort of interaction is, after all, expected, thus explaining both the sparseness and connectedness of this economic sector.

7.4.4 The Role of Visualization and Layout Algorithms to Uncover Structure in Networks

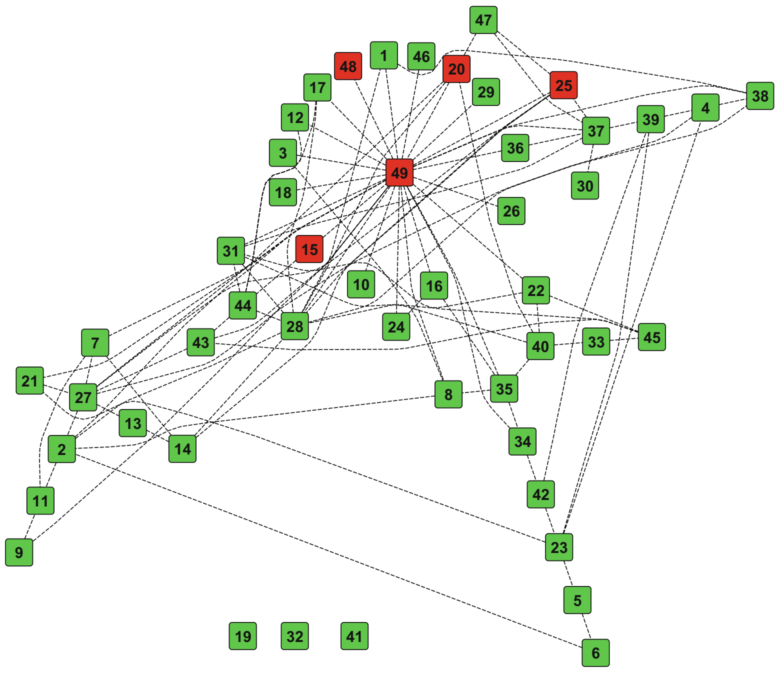
In the area of business analytics, there are many adopters of the use of visualization and layout algorithms since it is believed that they help to extract characteristics of interest. A discussion of the myriad of approaches that can be used is out of the scope of this chapter, and perhaps even of this collection, requiring a detailed survey we cannot do here. We do, however, want to illustrate on a few approaches available via some of the public domain packages and comment on the biases that such an analysis can bring if it is only guided by human perception of the final outcome. We note, however, that any specific mathematical model will certainly bias a search procedure to produce outcomes that reflect "some aspect" of the structure present in the data.

To simplify the discussions and allow readers to experiment on their datasets of interest after reading this chapter, we discuss a few layout algorithms given by the *yEd* software package.[9] In Fig. 7.8a we see the result of the "Organic Layout" option of *yEd*. This layout is based on a general strategy used by many algorithms. The vertices of the graph are considered to be "physical objects" that have mutually repulsive forces. Edges, on the other side, are considered to be like "springs", attracting pairs of vertices. A numerical simulation is then employed to identify which 2-dimensional position of the nodes reached a configuration in which all forces reach an equilibrium. As it happens with the simulation of a multi-body physical system, given an initial configuration for the set of nodes the final configuration reached is a certain local minimum of an energy function; there is no guarantee that this layout is optimal in any sense. Since the simulation stops when the forces on the nodes are negligible, this is likely to be a local minimum of that energy function. Such a layout helps to uncover highly connected subgraphs connected with more peripheral sets of vertices. Symmetries are also generally easy to spot with this approach and, although it is not directed to avoid the presence of edge crossings, the layout generally has a few of them. In the case of Fig. 7.8a, after completion of the automatic layout, we have to do some manual fine tuning of the

[9]https://www.yworks.com/yed.



(a)



(b)

Fig. 7.8 Two visualizations of the interaction graph of the Construction Firms Network of the Italian town of Porto Empedocle. The firms have been anonymized for confidentiality reasons and the ones in red are assumed to have been infiltrated by organized crime. (**a**) Result obtained of displaying the network with the "Organic layout" option of the software *yEd* followed by a fine-tuning manual optimization to increase visibility and reduce edge crossing. (**b**) Result obtained using the "Tree and Baloon Layout" options of the software *yEd*

final solution to allow some connections to be clearly distinguishable and to avoid ambiguities in the final layout of edges. We note that one very connected red node (corresponding to node 49) is clearly distinguishable by this layout but the other four nodes coloured in red are inconspicuous (we will later explain what these red nodes are; see also the description of Fig. 7.8).

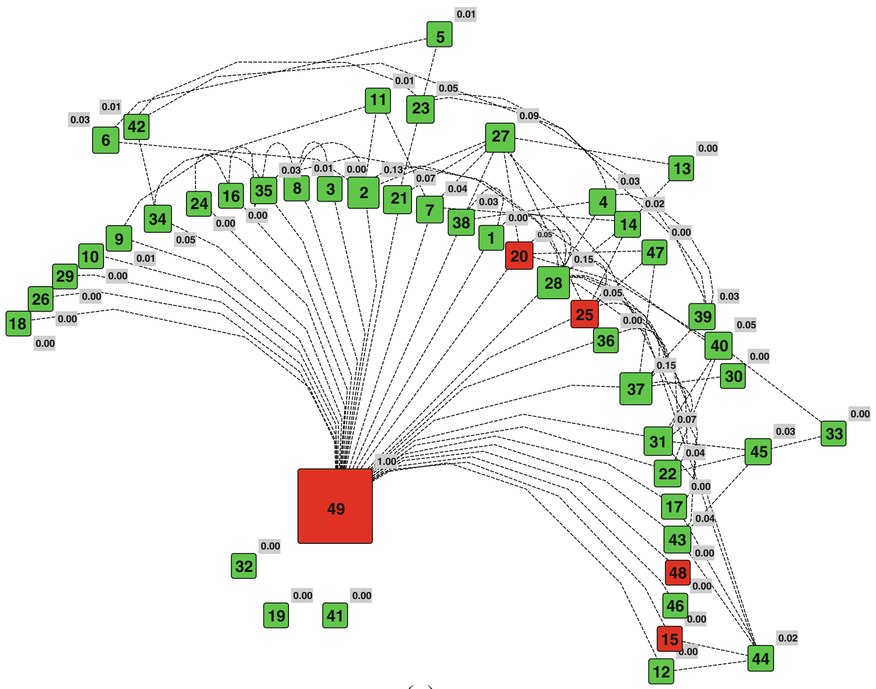
The second layout is based on the “Tree Layout” (Balloon style) option of *yEd* shown in Fig. 7.8b. This layout method is originally designed for trees (i.e., graphs that have no cycles), so we are really stretching this layout algorithm a bit far away from the original design specification since our graph is not a tree. We have, however, a relatively sparse graph and the layout method actually does a good job on it. Node 49, which is indeed very central, remains in a relatively central position and the layout of the nodes follows three major axes. We note that this characteristic structure is clearly not visible with the previous layout and indicates that the red nodes are within a certain “core” part of the graphs and that some companies that are quite far in terms of the distance (in terms of shortest paths in the graph) to one of the red nodes. Again, apart from node 49, the other nodes look relatively inconspicuous with this layout.

We present the results of another layout algorithm in Fig. 7.9a. In this case, it is a “Radial Layout” (again of *yEd*) and the algorithm places vertices in concentric circles around the common centre. A “centre node” is defined, so that the minimum number of circles can also be defined. Again, perhaps to no surprise now, node 49 has an even more predominant role. Also, in this case, the other red vertices are not particularly different from a large number of vertices that had at least one economic transaction with the company represented by vertex 49. We have selected the “size” of the vertex to be proportional to its “node betweenness” (a centrality score for each node in a graph). This is one measure of centrality and we dedicate Chap. 8 of this collection to discuss some of the most commonly used centrality scores.

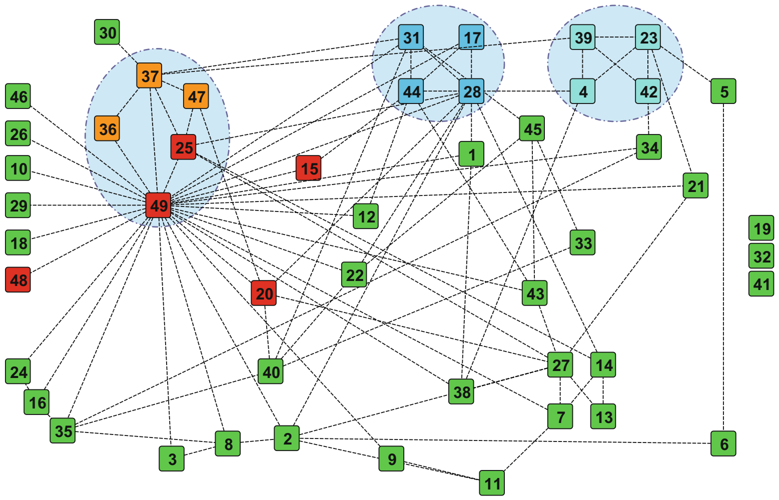
Obviously, other alternatives exist here (see, for instance, the results presented in Figs. 8.1 and 8.2) of Chap. 8. Different layout algorithms and different centrality scores can give, together, some interesting new insights for large networks.

One of the chapters of our collection of works is dedicated to a layout algorithm for visualizing graphs which is based on a combinatorial optimization approach (see Chap. 16). In [143] this approach was used to visualize a dataset of the world’s best 500 universities as well as in other two different datasets. The approach had been previously tested on a time series dataset from the Standard & Poor’s 100, which follows the stocks of the 100 largest US Companies [142]. Such an approach has also been used in the analysis of several thousands of time series from biological experiments in [52]. An online interactive resource was specially built[10] for this large-scale time-series analysis. The method has also been used to study the technical efficiency of hospitals and visualize the results [313].

[10]http://stability.matticklab.com.



(a)



(b)

Fig. 7.9 The first figure (**a**) helps to clarify the impressive central role that firm "49" has in the construction network of firms of Porto Empedocle. As it is a provider of concrete to the other local firms it is a prime target for organized crime activity infiltration. (**b**) shows three communities which have significant inward triangles. The process is stopped when, finally, a community with five firms is found and no other triangles remain in the graph. (**a**) Results of using the "Radial Layout" option of the *yEd* software using the same graph as previously shown in Fig. 7.8a, b. In this case the size of the node is proportional to the "Node Betweenness" (also computed via *yEd*). (**b**) Results of the iterative application of a memetic algorithm to identify the most cohesive-connected community for the Construction Firms network of [117]

7.4.5 Infiltration in the Legitimate Economy: The Case for Triangles

Finally, we present in Fig. 7.9 the results of an iterative process that identifies a single cohesive connected community of vertices and, at each step, removes the one found. The resulting graph is reduced (as now all vertices of that community and the edges connecting to them are removed). The layout of Fig. 7.9 was created manually (as yet there is no automatic solution for generating the communities followed and a layout that enforces these nodes to be closely placed in the 2-dimensional plane). We first describe how the communities were found followed by the results of the whole procedure. In this section we will also answer the question of what these red vertices really are. The result makes, at least in this case, an anecdotal case for the quest of identifying triangles in these transaction networks.

In the first iteration our memetic algorithm for the CONNECTED-COHESIVE problem identifies a group of four firms (i.e., {23, 39, 4, 42}, upper right, in light blue in Fig. 7.9) with two inbound triangles and no outbound ones. After removal of these vertices and their connecting edges, a second cohesive connected community was found in the remaining graph ({17, 28, 31, 44}, in blue in Fig. 7.9), again with two inbound triangles but now with two outbound triangles as well. In the third and last iteration of the method we found five firms ({25, 36, 37, 47, 49}) with three inbound triangles and no outbound ones.

Two vertices are coloured in red in this last community of five, the by-now-famous vertex 49 and vertex 25. We can unveil the mystery now. The red vertices indicate the firms that were already considered "subject to Mafioso infiltration" according to the documents discussed in Section 2.2 of [117]; the other firms in this third cohesive connected community are coloured in orange in Fig. 7.9. The iterative procedure now stops here as there are no other triangles in the remaining graph after removal of these five. This process brings to the attention three firms, represented by vertices 37, 47, and 36, for possible further investigation as they are part of this group. This is an interesting finding. The most connected company (49) is understandably "suspicious", but the existence of three connected cohesive groups suggests that these "triangles" of financial transactions may be an initial lead for further investigations.

7.4.6 Identification of Overlapping Communities

Another chapter of this collection (Chap. 9) presents an algorithm for the detection of overlapping communities in graphs. This methodology was first presented in the late 2016 in a study that aimed at identifying *"the limits of a brand"* in co-purchasing networks. We base the results we are about to describe on the use of that method. We have been using a memetic algorithm that, although in this case has no guarantee of optimality, it has been shown that it delivers optimal results for networks of these sizes. We thus considered it interesting to see what the algorithm would reveal if we now allow communities to be defined even having some of its vertices belonging to one or more of them.

The results produced by this method are presented in Fig. 7.10. The memetic algorithm has identified five communities with overlaps. These communities are shown in Fig. 7.10 with different colours. In each of the five figures we show one of them. Interestingly, the company represented by vertex 25, which is known to be a victim of Mafioso infiltration, is present in two communities. In one it shares membership with vertices 27 and 14 (in yellow, Fig. 7.10d). There seem to be no triangles here, but other cycles in the graph exist involving four firms (i.e., (25,27,13,14),(25,14,7,27)). Vertex 22 is very conspicuous, it belongs to three different communities, but it only has a connection with one of the nodes already known to be infiltrated (Vertex 49, which is in a single community). However, it is unclear if any relationship exists between the number of shared communities that a vertex has and its role as a central agent in infiltrations in the legitimate economy. The existence of cycles with length greater than three (triangles) indicates that the definition of the CONNECTED-COHESIVE problem may eventually be generalized to another problem in order to also include cycles of four or more vertices (which would not be unusual in organized crimes activities).

7.4.7 The Weighted Feedback Vertex Set Problem

The discussion of the previous subsections on the dataset of Porto Empedocle and the previous discussions in which we highlighted that we can render the graph "triangle-free" by iterative deletion of connected cohesive communities, together with the fact that the graph has cycles of length greater than three, oblige us to introduce another important problem to analyse the structure of graphs.

In the k-FEEDBACK VERTEX SET problem we are given a graph $G(V, E)$ and the task is to identify if there is a set V' subset of V such that $|V'| \leq k$ vertices whose removal leaves us a graph $G'(V', E')$ that has no cycles. Such a graph is called a maximum induced forest (as the resulting graph can be a group of trees which are not necessarily connected). This problem is also NP-Complete.

In the weighted variant, we have a graph $G(V, E, W)$ in which W is the set of weights that are given to each of the vertices. The task is then to find a feedback vertex set V', and among all of them the one that minimizes the total sum of the weights of the vertices V'. A memetic algorithm for this problem was proposed in [44]. For instance, given the data of Porto Empedocle, we can assign an arbitrarily large weight of $w = |V| = 49$ to all the vertices that have been known to had a Mafioso infiltration. In that case, no such a node will be selected in any feature set (as they have a very large weight). We can give a weight of $w = 1$ to any node that is not known to be infiltrated. In that case, a minimum weight feedback vertex will be selecting only from those vertices and will select one of minimum

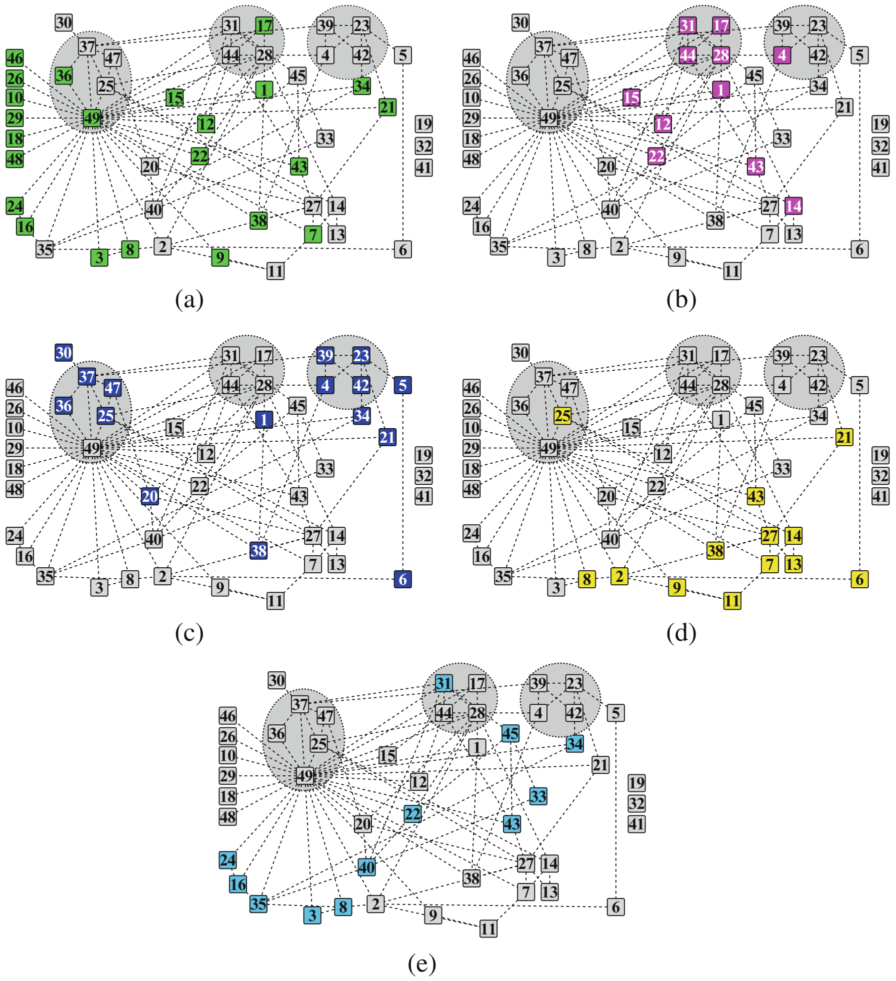


Fig. 7.10 Each of the figures show one of the five overlapping communities obtained via the memetic algorithm described in [98]. Interestingly, the most connected company in the network (49, in the connected cohesive group with five nodes) in the network is a member of only one community, while others connected to it (like 8, 22 and 38, for instance), are present in three different communities. Note that while 49 (a company that supplies concrete and thus has a high degree centrality in the network) is perhaps "less central" from the perspective of its lack of overlap with other communities. This new approach reveals new interesting subgraphs for consideration by the authorities. (**a**) C1. (**b**) C2. (**c**) C3. (**d**) C4. (**e**) C5

cardinality. However, the weight function may be used to "prefer" some of them, biasing the result to the search according to the intention of the modeller. We leave for the curiosity of the reader the identification of different feedback vertex sets in this dataset.

7.4.8 Are You Buying from a Fair Market?

Large corporations and governments frequently engage in several public bids for purchasing of items and services. Other groups respond to these bids, and subsequently, the best offer that satisfies the conditions of the buyer is finally accepted. However, has the price been manipulated somehow?

A *cartel* is a group of suppliers who set up agreements to increase the profits of the group by some practices. In some cases, this may involve restriction of the demand, limit the supply, or other forms of price fixing are orchestrated. There is an important issue to highlight. By its own definition a monopoly cannot be a cartel (e.g., there is no group of suppliers that can conspire, in a monopoly we just have a single group supplying services or products). Then the detection of cartels requires some sort of "network discovery" from existing longitudinal data.

One special way of setting up a cartel is by *bid rigging* [135, 269]. It is a form of *collusion* that is illegal in many countries and its identification can save, potentially, huge amounts of funds to both governments and large corporations. In a potential case of bid rigging, a contract is promised to one party (e.g., the State of Paraná in this case) even though to disguise the purpose, other parties also present a bid. This can help the whole group to fix prices.

It is generally very difficult to detect cartel formation in a single public bid. Using longitudinal data, however, some situations like the one of Fig. 7.11 can be observed. In this situation, we illustrate a potential scenario in which a group of companies is very successful in repeated bids while there is a group of other companies that are notoriously unsuccessful and they have participated, on average, on the same bids. This type of practice is called *cover* or *complementary bidding* and graph-based analysis of existing datasets can help to detect the practice and alert the purchaser of a possible unfairness in the market.

A few years ago, a team of two researchers looked at patterns in public bids for construction and engineering services in one of Brazil's largest economical and political regions, the State of Paraná [99]. To have an idea of the typical size of these problems, the State conducted 21,878 public bids, with a total of 41,385 participations by 15,955 distinct companies during a single year (2011). Brazil is a Federation of states and Paraná is one of them. It has 399 municipalities and is divided into 10 meso-regions. The capital is the city of Curitiba and it was the centre of the study being the largest meso-region.

The researchers built a weighted undirected graph in this following way. Each node in the graph represents a company. Two companies are connected by an edge if there exists at least one bid event during the year in which the two companies were competitors. The weight of an edge is the number of times the two companies were competitors in one or more bid events during the year.

Gabardo and Lopes conducted an analysis of the bids of Curitiba, Paraná's capital [99]. In their analysis of the year 2011 bids they identified a total of 544 companies involved and the presence of 3129 co-bidding events creating a graph that was not connected (each node represents a different company and an edge exists if there has been at least one event in which both companies bid at the same time). Instead of using a mathematical model that would detect subgraphs which would resemble those of Fig. 7.11, they have used the edge structure to partition the set of vertices of the graph.

Such a partition of the set of vertices is informed by the structure of its set of edges. Community structure identification is generally based on this principle, that groups of vertices highly connected constitute a community. This opens the mathematical modelling problem, whose objective function may guide the search process. One approach to find communities is to use "modularity" as an objective function (we presented this score in Sect. 7.3.7). For the purpose of the discussion we have here, and according to the definition of modularity, a *good* community should present a significantly higher number of connections between its nodes than found in a random graph [232]. Equation (7.3) shows how to compute the modularity Q given a network $G = (V, E)$ represented by an adjacency matrix $A = \{a_{ij}\}$,

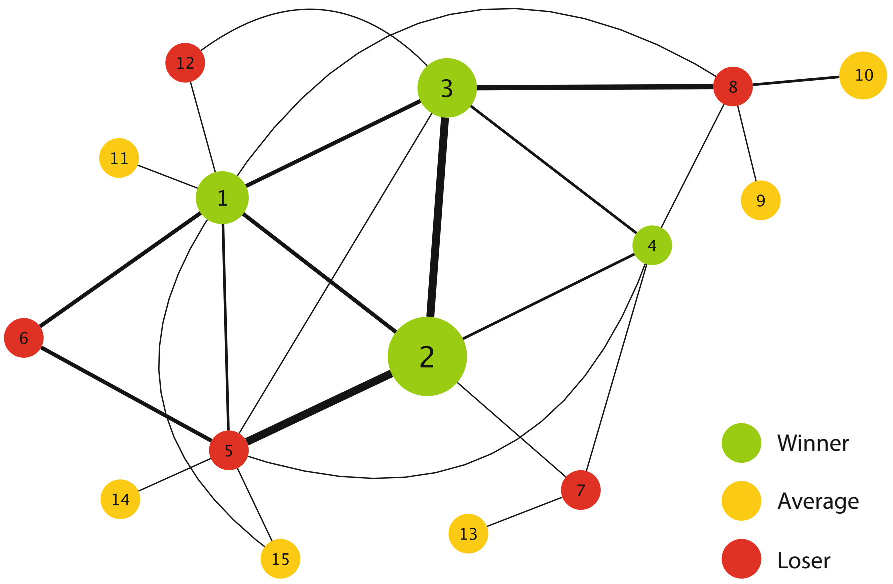


Fig. 7.11 A graph of 15 hypothetical companies that have engaged in public bids participation over a period of a year. In green, the companies that have been successful in more than 75% of the bids on which they have participated (the frequent "winners"). In red those that have lost more than 75% of the time in their bid events (the others in yellow, with performance in the two medium quartiles). The edges' thickness is proportional to the number of bids on which each pair has participated. A group of four "winners" (i.e., {1,2,3,4}) is connected to a number of frequent "losers" (e.g., red vertices {5,7,8,12}) which have been co-bidding. The identified subgraph may indicate the presence of *complementary bidding* also known as *cover bidding* or *courtesy bidding* which is an illegal practice in many countries

$$Q = \frac{1}{2|E|} \sum_{e_{ij} \in E} \left(a_{ij} - \frac{d(v_i)d(v_j)}{2|E|} \right) \delta_{ij}, \tag{7.3}$$

where Q is the modularity, $a_{ij} \in \{0, 1\}$ is a value of the adjacency matrix A for the nodes v_i, and v_j, $d(v_i)$, and $d(v_j)$ are the degrees of nodes $v_i, v_j \in V$, and the Kronecker delta δ_{ij} takes the value one (1) if v_i and v_j are in the same community or takes the value zero (0) otherwise [230].

From the 44 communities observed, four communities were too large to be considered of interest. They are unlikely, as a whole, to be involved in complementary bidding since they had 26.10%, 15.62%, 14.34%, and 12.32% of the companies in them. While this does not exclude other possible subgroups (within these large communities) to actually be engaged in illegal practices, this means that we may require a different approach (e.g., via another objective function, such of those given by the CONNECTED-COHESIVE problem and its variants) if we may further partition the set of companies in even smaller groups.

With the data they have obtained, Gabardo and Lopes then turned their attention to the largest of the 40 remaining partitions of the set of nodes. Surprisingly, some of the companies in this group have not won a single bid during the year (in some cases, after bidding five times). One company that did not have a successful bid in 2011 was further investigated over the period and it was observed that it actually had no successful bid from 2005 to 2012. While the rest of the study remains confidential, the methodology helped the government to identify other companies that may have been engaging in arranged bidding operations thus possibly distorting market practices.

There are other illegal tactics that may require new mathematical models for its detection. One is called *bid rotation*. In this case a group of conspirators decide to take turns so they plan which company will be successful in each of them. Consequently, they will be designated to be a loser of other contracts in the future. Cartels of companies, if unidentified by data analytic means would then manipulate business practices endangering what should be a fair competition processes of the economy. Networks based on longitudinal analyses of the processes are then essential for elicitation of possible cases.

7.4.9 Detecting Fan-Out-Fan-in (FOFI) Structures

For the study of financial transactions, which have "timestamp" information, researchers rely on *directed* graphs. Arcs in these graphs indicate some sort of time-stamped transaction between two entities. Paths in directed graphs can indicate the flow of funds which may pass between actors in the financial system. Like in the study of Karsten Weihe that will be discussed in Sect. 7.5.1, paths are our central subject here. They are time-respecting indicators of transactions in which funds may have been transferred from the source of the path to the end of it.

Fraudulent behaviours in financial networks of this type often exhibit particular structures. Michalak and Korczak [209] gave an impressive account of the scale of the problem and suggest that methods based on mining for subgraphs are necessary now to detect suspicious transactions.

In *smurfing*, a financial transaction is separated in multiple smaller transactions which, together, would have caused the regulating financial system to take notice. Individually, however, they can escape the identification since they are below a certain threshold. "Flying under the radar", they would escape identification but they leave a trace. This is called the "fan-out-fan-in" structure (FOFI). This said, a FOFI is basically a directed subgraph of time-respecting paths of transactions that stem from a source vertex (sender), then expands in different paths including a number of intermediate vertices (the internal hops or "intermediaries") to finally converge in a given vertex (the "receiver"). Holme and Saramäki [131, 132] present several examples of temporal networks. In [268], Redmond, Harrigan, and Cunninghan present results of mining for these structures in a network composed of nearly 90,000 actors that can be borrowers and lenders in a network of nearly 3.5 million edges. They mine for several of the graph structures discussed in this chapter including cliques, trusses, and the so-called FOFI structures that take into account of flows of money across multiple accounts [268].

It is interesting to note that there is a natural extension of this work now involving a variant of the network alignment problem (which is presented in Chap. 12) that would allow for some edges/arcs not to be part of the putative FOFI structure. The missing transactions may occur in another system, for instance, and the number of unaligned arcs may be present in a different (unknown) network. This is indeed an interesting problem waiting for a good mathematical modelling that characterizes the core legal requirements needed to be a proper tool for cyberfraud identification purposes.

7.5 The Power of Data Reduction

The problem areas so-far discussed in this chapter indicate that for large scale graphs there is an intrinsic difficulty in solving some core problems to optimality. We would say that, in part, the difficulty is also *practical*. We have defined the computational complexity of some of them, so it may be the case that for some *instances* of the problem (i.e., a particular graph given as input) it is very difficult for an exact algorithm to finish and thus prove that the solution found is indeed optimal. Could it then be possible that we can "transform" such a graph into a smaller graph, solve the problem exactly for this small graph, and then "blow up" the solution for the smaller graph, somehow, so that we can finally have a solution for the original large graph?

To achieve such a feat we will need some sort of *safe data reduction* techniques [81]. An area of Computer Science has been actively dedicated to this approach over the past three decades. We note, however, that this does not limit only to graphs. Safe data reduction would be useful in many scenarios. In the next section we will present this approach with an emphasis in graphs and starting from an important study that, quite independently of this line of research, led the author to reconsider his approach to algorithmic development and problem solving. His story then becomes quite universal and certainly applicable, in spirit, to other domains.

7.5.1 Covering Trains by Stations and Other Applications of Data Reduction

There are many situations in business which bring out our attention to several types of NP-Complete problems, or sometimes even computationally harder than those. One alternative is to see if the situations we are dealing with, in practice, have special subcases of interest that can be solved with an efficient algorithm. For very large instances of the problem of interest, if that is not the case, we can ask ourselves: is it possible to reduce the dimensionality of the data and yet still be able to solve it to optimality once a small "kernel graph" (i.e., a graph that cannot be further safely reduced) has been finally found [144].

One of the best stories we have on the power of data reduction was perhaps the one contributed by Karsten Weihe in his brilliantly titled paper: *"Covering Trains by Stations or The Power of Data Reduction"* [324]. This is an interesting case as the problem was indeed was defined on a network, but the solution method was quite indirect and very powerful.

The following story is then archetypical in nature. Its importance grows as many researchers become aware of the key message. It clearly illustrates how a large logistic problem can be efficiently reduced to a problem that becomes quite practical to solve. More formally, people say that it is "kernelized". In Sect. 7.6 we will discuss a field of theoretical computer science which aims to explore this way of solving computational problems and we will illustrate with problems in graphs and hypernetworks.

7.5.1.1 Weihe's Data Reduction Success

In the mid-1990s, a German academic, Karsten Weihe, was working as part of a cooperation agreement with a subsidiary of the central German train and railroad company (Deutsche Bahn AG). His task was to solve a problem which seemed very complicated at first hand and he first considered that, a priori, a sophisticated algorithmic solution may be required. He was given data from a network of European train schedules, information on stations run by the Deutsche Bahn AG and by other service providers, identification of trains classes (i.e., high-speed, international or long-distance, regional, local, and other types), etc. The European train schedule contained information of 140,000 trains, 25,000 stations, and 1,600,000 single train stops.

He modelled the problem as a graph problem. The problem was defined as the *"path cover by vertices"* (PCV) problem (in a graph $G(V, E)$) and he proved that it is NP-hard. A path p^l in a graph G is an ordered sequence $(v_1^l, \ldots, v_{n_l}^l)$ of vertices such that $\{v_i^l, v_{i+1}^l\} \in E$ for all $i = 1, \ldots (n_l - 1)$. Paths can share vertices and edges and every edge belongs to some path. The problem is then the following: given a set of paths $\{p_1, \ldots, p_k\}$ of G, and a partition of $V = V_1 \cup V_2 \cup \cdots \cup V_m$ into m disjoint classes, the task is to find a subset $V' \subseteq V$ such that: (a) every path of the given set contains at least one vertex in V', (b) from all those sets V' that satisfy that condition also bring the one of minimum cardinality, (c) V' must also maximize the vector $(|V' \cap V_1|, |V' \cap V_1|, \ldots, |V' \cap V_m|)$, lexicographically.

In this graph representation (i.e., $G(V, E)$), the set V is the union of all stations met during the path of at least one of the trains, but E is not necessarily the set of direct connections in an underlying railroad network. An edge $e = \{a, b\}$ is in E if and only if there is at least one train p_l such that $a = v_i^l$ and $b = v_{i+1}^l$ or vice versa for some $i \in \{1, \ldots, (n_l - 1)\}$. This means that an edge $e = \{a, b\}$ if there is a scheduled train that stops at a and b and may pass another station c which is in between, in the physical network, without the train stopping there.

The set of paths $\{p_1, \ldots, p_k\}$ represents the set of scheduled trains as found in their timetables, this means that the set includes the trains that are not operated every day. Also, if the same course of stations is served several times a day, there are an equal number of paths in the input of the PCV. Finally, the reason of the lexicographic maximization of the solution comes from the preference of the final solution to have stations that are operated by Deutsche Bahn AG.

Weihe observed that there were two types of *safe reductions* of data dimensionality that can be used. They are based on equivalence and dominance relations on paths and vertices (see [324]). The algorithmic scheme is then simple, a condition is computed, and if it is the case (that the condition is met) then an action (i.e., the reduction) is applied thus diminishing the dimensionality. After a *vertex reduction* is applied, three things have to be done: (a) a vertex v is removed from V and all its edges from E, (b) if $a, b \in V$ were incident to v and there is a path p_i which contains $a - v - b$ or $b - v - a$ as a subpath, an edge $\{a, b\}$ is added to E (it is not already there), (c) all occurrences of v in the paths are removed.

Analogously, if the condition for a *path reduction* for a path p_i is present in the data, we then do: (a) p_i is removed from the set of paths, (b) each edge $e \in E$ which is not part of any path afterwards it removed from E, (c) each vertex $v \in V$ whose path set is empty afterwards is removed from V.

These reductions in size are applied until there is a non-reducible instance, also known as a "kernel". This approach is sometimes known as "preprocessing" in Operations Research.

Given the size of the original data, it is really impressive that the application of these safe preprocessing rules can reduce the graph in such a dramatic way. Weihe himself was very impressed. He obtained "kernels" that were so small that they could be solved by brute-force alone and, sometimes, even by hand due to their small dimensionality. He concluded:

The power of rigorous data reduction in a preprocessing phase should not be underestimated. I thought a lot about appropriate algorithms approaches *before* I implemented the preprocessing. All of these considerations are now obsolete. The algorithmic problems presented in this paper might be an extreme case. However, it might generally be a good idea to implement the preprocessing first, to look at the results, and to invest time in the design of the core routine only afterwards, because only then are the characteristics of the input to the core routine definitely known.

The story is illustrative of the power of safe data reduction and also uncovers that the structure of $G(V, E)$, which may be highly different than the structure of the underlying railway network, can be effectively used to solve large scale optimization problems in this way. The question is: how can we recognize these structures?

7.5.2 Sources of Structure in Networks—First Act: Genus and Planarity

If the problem studied by Weihe is NP-hard, why has it been so relatively easy to solve it to optimality for such a large instance? Actually, this is hard to be an isolated case in business and data analytics. Well, as discussed in the first chapter of this collection, sheer size alone does not matter when we deal with issues of computational complexity. It may be possible that there is some particular characteristic of this problem that make it "easier in practice". In situations in which we are in control of the type of instances of problems we can face (i.e., by design) it may be possible that we can have efficient algorithms, something that would help our business.

It would be interesting to link the existence of an efficient algorithm with a certain property that the instance may have. Which examples of such properties are known? Weihe makes an important observation in [324] when he discussed the problem of covering train paths with stations. He said: "…the underlying railroad network has nice properties such as a small *genus*".

What is the genus of a graph?, the reader may rightly ask. The quick answer is: given a graph $G(V, E)$ then we can define *the genus of a graph* $(g(G))$ as the minimal integer $g \geq 0$ such that the graph can be drawn without crossing itself on a sphere with g handles (i.e., on an oriented surface of genus g). What this means requires a bit of explanation.

This gives the opportunity to introduce another very important property of graphs: *planarity*. A graph $G(V, E)$ is planar if and only if it can be drawn in the two-dimensional Euclidean plane in such a way that none of its edges cross. The importance of this is clear if we think of visualization of data. Given a graph, we first may ask if it is possible, not, to create such a drawing. In case that this is not possible, perhaps another strategy for layout algorithms for graphs may be needed. For instance, you may wish to minimize the number of edges that cross in the final layout. But the issue of planarity has important computational efficiency considerations, some NP-hard problems can become easier if the input is restricted to be a planar graph.

In the literature, a clique of four vertices is called a K_4. The reader can quickly check that there is a way of drawing the edges of a K_4 in such a way no two edges cross. (Hint: drawn edges do not necessarily need to be straight segments between nodes.) A 5-clique, called K_5, is not planar. Another graph that is not planar is called $K_{3,3}$, if you are not fluent with this mathematical notation see the grey box below.

Bipartite and Biclique Graphs

A graph is said to be *bipartite* if its set of vertices V can be divided into two disjoint (i.e., $V_1 \cap V_2 = \emptyset$) and independent sets and such that every edge connects a vertex in V_1 with a vertex in V_2. A bipartite graph does not contain any odd-length cycles. Interestingly, hypergraphs can be modelled by a bipartite graph (V_1, V_2, E) in which the *biadjacency matrix* of (V_1, V_2, E) is a (0,1)-matrix of size $|U| \times |V|$ that has a one for each pair of adjacent vertices and a zero for non-adjacent vertices. V_1 is then the set of vertices of the hypergraph, V_2 is the set of hyperedges, and E contains an edge from the hypergraph vertex $v \in V_1$ to a hypergraph edge $e \in E$ exactly when v is an endpoint of e.

A *biclique* is a special kind of bipartite graph in which each vertex of one of the two independent sets of vertices is connected to other vertices of the other independent set. They are denoted as $K_{n,n'}$ with n and n' being the number of vertices in the two independent sets. The reader may now try to prove that $K_{3,3}$ is not planar. (Hint: Rosen [275] in which part of the argumentation is given and it is proposed as an exercise.)

7.5.2.1 How Can We Prove if a Graph is Planar?

An important result is owed to Kazimierz Kuratowski who in 1930 proved the conditions for identifying if a graph is planar (independent discoveries are also discussed in [305]). Several proofs of Kuratowski's theorem lead to linear time algorithms for testing if a graph is planar [33], an area of research now obviously called "planarity testing" [248]. The above mentioned K_5 and $K_{3,3}$ have a central role in some of the proofs and algorithms.

Graphs, Networks, and Topology—Euler Again!
We were discussing the *genus* of a graph, and then we move on to discuss *planarity*. Let's see now how these are connected.

This brings us to expand a bit on the story of Euler and the puzzle of the Königsberg bridges tour. It is known that Euler was indeed surprised by his own solution method. He wrote to Carl Leonhard Ehler, mayor of Danzig, that the problem "bears little relationship to mathematics . . . for the solution is based on reason alone, and its discovery does not depend on any mathematical principle."[11] The title of Euler's paper was *Solutio problematis ad geometriam situs pertinentis* (or, in English, "The solution of a problem relating to the geometry of position"). He noticed that the solution seems to be in "a new type of geometry", and for that, this paper is also considered a pioneering one in the mathematical area known today as *Topology*. Now tightly embraced as being part of pure mathematics, topology is the formal study of relationships within space. In particular, its core aim is to study the properties of shapes which are preserved under deformation of objects caused by bending and stretching.

How this connects to our problems in graphs? If, for instance, we may have been given a graph or a network and we deform its physical nature by stretching or bending its edges, or changing the position of nodes by displacements, it may still have properties (i.e., topological ones) which are unaltered. That is exactly what happened when Euler moved from the physical problem (involving land masses and bridges) to a problem involving only nodes and edges, and his solution was indeed based on a topological principle. *Topological graph theory* is the mathematical field which studies the embedding of graphs in surfaces. The *genus* is one of the oldest known topological invariants of surfaces. For mathematicians, a "natural question" is to understand if a graph and be "embedded" into a surface of known genus. The plane has genus equal to zero and computer screens are planar, or deformable from a plane. Can we embed a graph in the plane?

A spherical surface also has *genus=0* and it can be proven that a graph is planar if and only if it can be drawn on such a surface without any pair of edges crossing. For any orientable surface that it is connected, the genus is an integer value that represents the maximum number of cuttings along non-intersecting closed simple curves without rendering the resultant manifold disconnected. A *torus* has a genus of 1. One topological way of thinking of a torus is to think of a sphere to which a

[11] https://www.sciencenews.org/article/eulers-bridges.

handle has been added. The *Klein bottle*[12] has genus equal to 2. The reader would then like to read again the definition of genus of a graph we have given in the third paragraph of this section.

This may be an interesting and important research direction. Given a problem that has one of these properties (i.e., a low genus of an associated graph) and for which we also know that the value of the genus is a small positive integer, perhaps we can find a way by which this problem is more "approachable" via some sort of preprocessing technique. Since planarity can be tested in polynomial-time (actually, in linear time!), many computer scientists naturally were drawn to study the importance of planarity for computational problems. Some NP-Complete problems were "easier to approximate" but several problems remain NP-Complete even if the graph is planar.

It is unlikely, though, that in any real-world scenario we will face a problem and the value of a genus of a graph will "be given" as part of the input, most likely we will have to compute it ourselves! The exception, perhaps, would be problems in which we could somehow constraint the type of instances to be solved to be graphs of low and/or known genus. What is then the cost of actually "discovering" this from a generic graph? This question has been answered. Unfortunately, the problem of determining the genus of a graph is also NP-Complete [306]. This said, we still need to pay some computational complexity "price" to recognize if the particular instance of a problem has "some characteristic" that makes it amenable for data reduction (noting, for instance, that we can detect if it's planar in linear-time, so there are some exceptions). In practice, though, it may well pay the effort if you can quickly prove/disprove that you may have a reasonable low genus on a graph (i.e., by resorting to a heuristic to have a good bound).

We will return to the discussion of genus and other types of characteristics of networks later in the chapter due to its importance to scalability [339]. The following subsection explains why perhaps the genus of a graph is important to understand the good performance of some evolutionary algorithms for graph optimization problems; in particular the so-called memetic algorithms, as they benefit from using information from different feasible solutions previously found. We give an example of a problem from the area of social networks.

7.5.3 Finding Another Influential Set in a Social Network When You Already Know Two

In the first chapter of this collection of works, we have already discussed a safe reduction rule for the *recombination* of two feasible solutions for the MIN VERTEX COVER. "Recombination" is a term borrowed from biology to identify the process by which new combinations of genetic material in offspring that are not present in the parents. The field of *evolutionary computation* frequently uses it to describe a

[12]https://en.wikipedia.org/wiki/Klein_bottle.

situation in which we have an instance of a problem and some feasible solutions for the problem which may not necessarily be optimal. Applying "recombination operators" we generate new feasible solutions (i.e., algorithms that receive as input the instance and two or more solutions and generate one or more new feasible solutions).

Data reduction rules can have an important role in recombination algorithms and in the creation of useful heuristics. These rules are highly dependent on the problem, but some of them can be used for other problems. For instance, for the problem of finding a dominating set in a graph we can prove that:

> If the input graph has a vertex u of degree 1, the other endpoint vertex v connected to u *must be* in the dominating set.

We can see in practice the effect of applying this reduction rule as part of a recombination algorithm. Suppose we are given the two dominating sets of Fig. 7.12a, b (called D_1 and D_2, respectively).

One possible recombination procedure that can be constructed works as follows (Fig. 7.13):

- **Step 1**: Identify all vertices in $D_1 \cap D_2$. Let V_D the set of vertices of G that are not in $(D_1 \cap D_2)$ but are already dominated by a vertex in $(D_1 \cap D_2)$.
- **Step 2**: Remove from G all edges u, v such that either u or v (or both) are in V_D.
- **Step 3**: Remove from G all edges u, v such that either u or v (or both) are in $D_1 \cap D_2$. Remove all vertices in $(D_1 \cap D_2)$.
- **Step 4**: Remove all the vertices, now isolated, in $V_D \cup (D_1 \cap D_2)$.
- **Step 5**: Find the optimal solution of the MIN DOMINATING SET problem of the remaining graph.

Now Fig. 7.14 shows the output of a recombination procedure that respects the vertices that are common to both dominating sets.

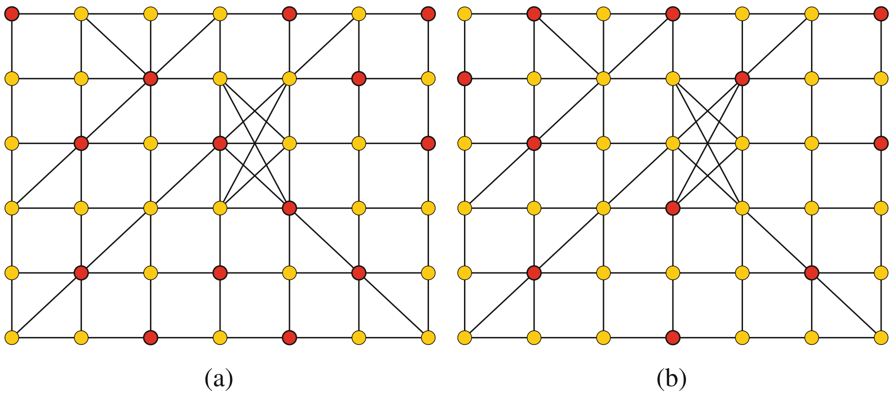


Fig. 7.12 Two "parent solutions" to be used in recombination operation. The "red" vertices are the vertices in two different dominating sets. (**a**) D_1, a 14-vertices dominating set. (**b**) D_2, a 11-vertices dominating set

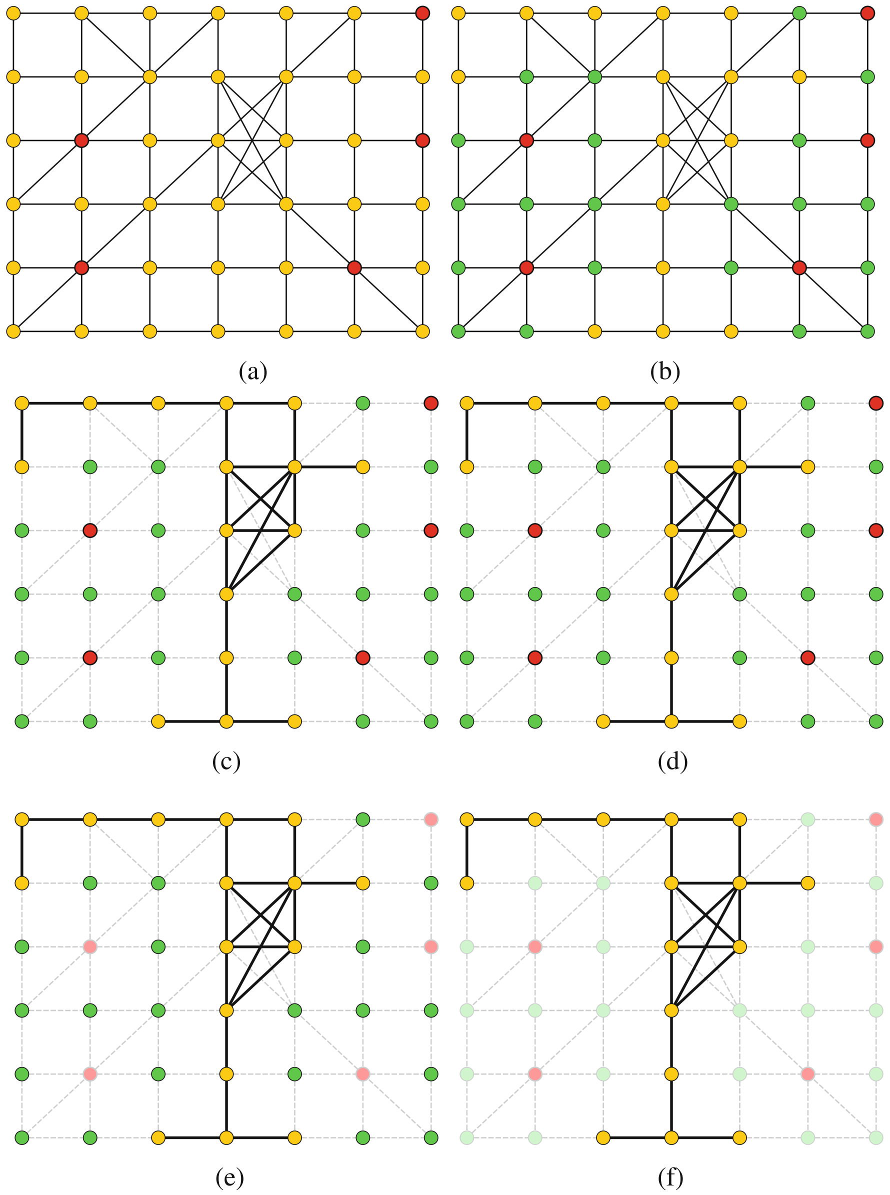


Fig. 7.13 Pictures depicting the recombination operation in detail. (**a**) Step 1-a : all vertices of $D_1 \cap D_2$ (in "red"). (**b**) Step 1-b : vertices in V_D are in "green". (**c**) Step 2 : remove edges incident to any vertices in V_D (now with light dashed lines). (**d**) Step 3-a : remove edges incident to any vertices in $D_1 \cap D_2$ (absent in this instance). (**e**) Step 3-b : remove all vertices in $(D_1 \cap D_2)$ (light red vertices). (**f**) Step 4 : remove all isolated vertices , in $V_D \cup (D_1 \cap D_2)$ (light green vertices)

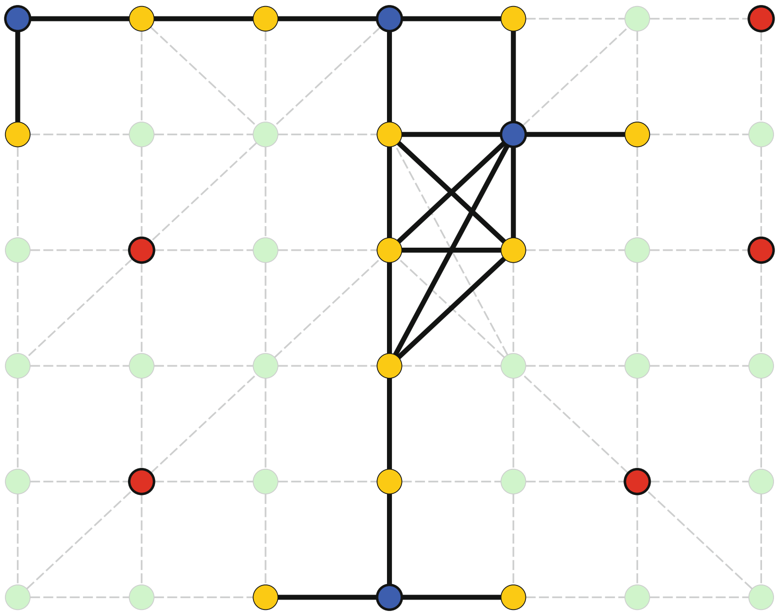


Fig. 7.14 Step 5 : of the recombination process. The dominating set is shown with highlighted vertices which are either "Red" or "Blue" Five "Red" vertices where common to both solutions (i.e., $D_1 \cap D_2$) and the vertices found after applying the reduction rule give other four "Blue" vertices that together make a feasible solution. Hence, the application of a reduction rules allows creating an offspring with only nine vertices

Note that **Step 5**, of the recombination procedure proposed above, may resort to a valid reduction rule. The "degree one rule" discussed for MIN VERTEX COVER also works for MIN DOMINATING SET. If you have a vertex of degree one, select the vertex linked to it (which may have degree one or higher). By repeated application of this safe reduction rule, we can find an optimal solution that only requires four vertices to dominate the remaining subgraph (with edges boldfaced in Fig. 7.14), thus we managed to build a new feasible solution using only nine vertices. On the other hand, it may be the case that this reduction rule may not be applicable, but other valid reduction rules would, so repeated application of known for MIN DOMINATING SET rules may lead to a feasible instance that it is known to be optimal.

7.5.3.1 The Many Forms of Recombination

Recombining high-quality solutions of combinatorial problems is one of the core issues of successful methods like memetic algorithms (see Chap. 13); they have earned a great reputation in graph optimization problems [10].

High-performance algorithms will optimize solutions and then they are recombined by procedures like this one. If, for instance, like in this case, the reduction procedure continues performing safe changes until no nodes and edges remain, the solution is indeed optimal for the remaining subgraph. However, is it the best we can achieve? We note that the recombination algorithm respects that common nodes in $D_1 \cap D_2$ are always present in the newly created solution? This said, among all the possible feasible solutions of the problem constrained to have $D_1 \cap D_2$ in the dominating set, this may not be optimal. In general, we may require a mathematical guarantee (i.e., a theorem that proves such a result).

For instance, consider the following alternative recombination algorithm.

- **Step A**: Identify all vertices in $D_1 \cap D_2$. Let V_D the set of vertices of G that are not in $(D_1 \cap D_2)$ but are already dominated by a vertex in $(D_1 \cap D_2)$.
- **Step B**: Remove all vertices in $(D_1 \cap D_2)$.
- **Step C**: Remove from G any vertex in V_D such that all its neighbours are also in V_D.
- **Step D**: Find the optimal solution of the MIN DOMINATING SET problem of the remaining graph.

We invite the reader to find a case (a graph, and two dominating sets) for which these two recombination algorithms dramatically differ in the newly created solution. Step C seems to be a clear difference between the methods, as vertices in V_D that are still connected to vertices that are not dominated may, in turn, contribute to obtain smaller cardinality solutions in Step D. By remaining in the subgraph, they may need to reduce the cardinality of the final dominating set. However, since they are already dominated, yet another recombination algorithm may be created that "knows" that these are needed to be dominated (so **Step D** may not require to solve MIN DOMINATING SET problem but a variant).

Note that the computational complexity of these recombination algorithms, in principle, is not bounded by a polynomial. We have not defined *how* Step D will be executed for both of them. If, for instance, we resort to a polynomial-time algorithm, the whole procedure is bounded by a polynomial in the number of vertices, however, we may need to give up on our claim on optimality. On the positive side, it may be the case that we can guarantee that the instance to be solved has some particular property. In that case, it may be possible that although the decision problem is NP-Complete, a polynomial-time algorithm may exist for graphs that have that property.

Since in modern techniques, like memetic algorithms, the recombination of solutions is done together with the use of other techniques for improving the solutions, there is a "synergy" between the basic components. There is an active search for the best combination of algorithms, both exact, approximated and heuristics.

In Sect. 7.5.1.1 we have seen that Weihe's recommendation is to use "preprocessing/data reduction first". However, our illustrations on possible use of these reduction techniques to develop recombination algorithms for MIN VERTEX COVER and the MIN DOMINATING SET indicate that they have a wider role in evolutionary algorithms, particularly of the memetic type. In practice, for many graphs and network problems (i.e., arising in a social network in which we want to find a minimum set of possible influencers) the genus is not necessarily low. However, it may be the case that some recombination algorithms may "divide" the problem in graph subproblems of low genus, thus being more amenable to obtain optimal solutions on them. Other type of parameterizations may exist, and data reduction techniques may become very useful for dealing with the arising subproblems when recombining feasible solutions.

7.5.4 Bipartite Graphs and Other Data Reduction Techniques

In many problems in the domain of business, it is easy to recognize that there is a natural representation in terms of graphs and, perhaps, nodes can be associated with two different types of objects in the real-world domain. For example, we can have problems in which we need to associate ships to harbours, doctors to hospitals, etc., there is a representation in which we still associate them to "nodes" or "vertices" and a relationship between them is represented with an edge. However, it may be clear that no edge between nodes of the same type would exist. These are bipartite graphs and we have given the definition in Sect. 7.5.2. For instance, a $K_{3,3}$ is a complete bipartite graph having three nodes, separated in two independent sets, in which each node of an independent set is connected to each node in the other independent set. Clearly a graph can be bipartite and not be planar (as we discussed before, the $K_{3,3}$ is non-planar).

Finally in Fig. 7.15 we see another example of a complete automatic layout that includes a complete bipartite graph between two layers. Placing these nodes relatively close to each other helps to understand where the densely connected regions are and identify the most central groups in the network of interactions. Put together, this other property of graphs can have important consequences also from an algorithmic point of view. A number of combinatorial optimization problems in graphs have, as special cases, those that are restricted to bipartite graphs. It is frequently the case that the computational complexity on these special cases varies, as well as results regarding their approximability could also significantly change.

7.5.4.1 The k-RED/BLUE DOMINATING SET and Influence Maximization on Social Networks

One particular interesting problem is the k-RED/BLUE DOMINATING SET defined on bipartite graphs. We first start discussing the more general case. We are given a graph $G(V, E)$ and a bipartition of the set of vertices (i.e., the set "Red" (R) and the set "Blue"(B), such that $V = R \cup B$). In the decision problem we are also given an integer $k > 0$ and the task is to decide if G has a set of size at most k of red vertices such that each vertex $v \in B$ is adjacent to at least one of the k selected red vertices.

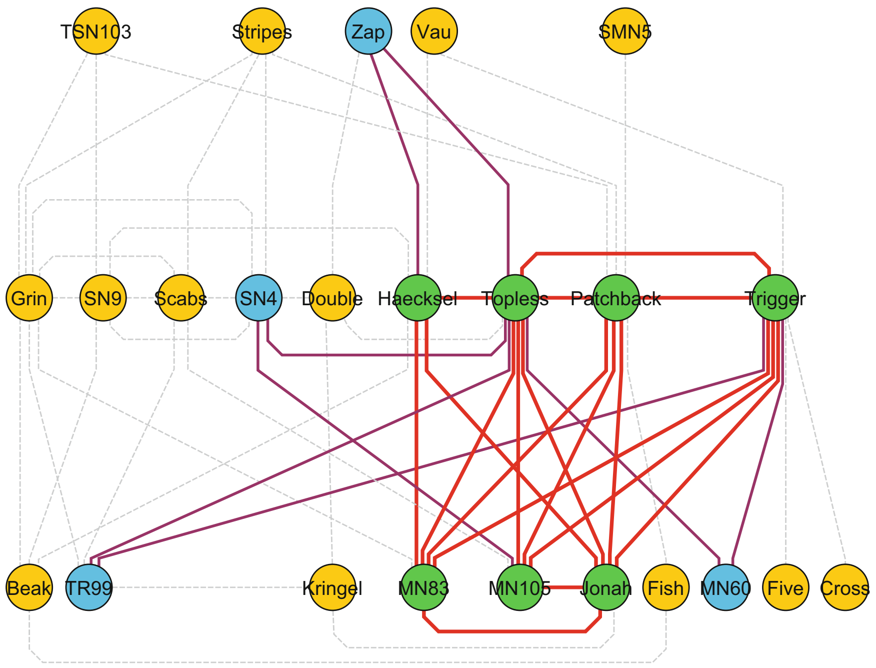


Fig. 7.15 Yet another layout of the graph first presented in Fig. 7.6. This result is interesting because it was automatically generated using the *yEd* software using the Hierarchical Layout (option BFS) and no fine-tuning was needed at all. The connected-cohesive community (coloured in green) is clearly identifiable as well as the four outgoing triangles. Nodes appear in layers. At the top, a 5-vertices independent set is placed (which could have been expanded to seven, since the nodes corresponding to the dolphins named "Five" and "Cross" could have been also included there). In cases where the graph is not planar, some good visualization algorithms may require the calculation of bipartite graphs, independent sets and cliques and somewhat "isolate" these "problematic" regions (with many edge crossings). Human perception then may help to "summarize" the main structures found

One direct possible application of the k-RED/BLUE DOMINATING SET problem would be again in the area of social networks and influence maximization. For instance, suppose that a set of members of a social network $G(V, E)$ have been identified, via a targeted marketing mechanism, as a set of individuals on which some action has previously been taken. They may already have received some advertising or communication. A previous iteration of a viral marketing campaign has used them as seeds to "diffuse" the message. On a second iteration of the campaign, it would be prudent not to target any member of this group again. Let's call this set $H \subset V$. Let $N(H)$ be the set of vertices of the social network $G(V, E)$ that is adjacent to at least one vertex in H. Let $N(N(H)) - H$ be the set of nodes that are adjacent to a node of $N(H)$ but they are not in H (i.e., in simpler terms, nodes that are at distance 2 of nodes in H in G). Now let's define $G'(V', E')$ with V' the set of nodes that are at distance at most two from H in G and $e' = (r, b) \in E'$ if and only if $r \in N(N(H)) - H$ and $b \in N(H) \cup H$ and $e' = (r, b) \in E$. If we call “red” all the nodes $N(N(H)) - H$ and “blue” all the nodes in $N(H) \cup H$, then we have a k-RED/BLUE DOMINATING SET defined on this subgraph of $G(V, E)$, the original social network.

Because we are searching for a subset of the red vertices, a solution of the k-RED/BLUE DOMINATING SET will then give us a set of members of the network that have not yet been previously targeted and, by targeting them now, we can indirectly also influence others targeted in the previous iteration of the marketing campaign. Let’s call such a set of vertices D (the ones found when we solved the k-RED/BLUE DOMINATING SET problem instance).

We can now delete from our original network G all the vertices H (and their edges) (since they have been previously directly targeted). We can then delete from G all the vertices in D (and their edges) but not their adjacent vertices (we spare these for a reason that will be clear later). Now we could proceed by finding another set of influential nodes in the remaining social network, (for instance, by solving a relaxation of the k-DOMINATING SET problem for the now reduced graph) and the process can be reiterated. Such an approach guarantees that, after successive iterations, individuals in the network are not subject to reiterated targeted marketing on a relatively frequent basis. Deleted sets could be incorporated to the graph after several iterations of the procedure (if the budget allows it, of course). Other mathematical models can also explicitly incorporate the costs of advertising as it may relate to the cardinality of the solutions sought, thus giving rise to new mathematical problems and models.

7.5.4.2 The k-RED/BLUE DOMINATING SET in Bipartite Graphs and its Role in Data Mining

It has been shown in [217] that any instance of k-FEATURE SET (the data mining problem described in the first chapter of this set of works) can be transformed into an instance of the k-RED/BLUE DOMINATING SET problem on a bipartite graph. This transformation can be achieved in polynomial-time. This result was discussed in the context of the (α, β)-k-FEATURE SET problem, which contains the k-FEATURE SET as a special-case since it corresponds to the case $\alpha = 1$ and $\beta = 0$ [217].

In this case, one set of vertices of the graph bipartition corresponds, one to one, to the set of features of the instance k-FEATURE SET so if k red vertices are found which are a solution, then they constitute a solution of the corresponding instance of the k-FEATURE SET. Moreover, [217] describes reduction rules for the problem that allows to reduce the dimensionality of the corresponding data mining problem. Three types of reductions exist: (a) reductions that identify a feature as essential, (b) reductions that eliminate features, (c) reductions that eliminate pairs of samples to be considered for further investigation. These last two reductions eliminate red and blue vertices and the first one establishes a feature to be essential, add it to the solution, and reduce the value of k by one.

Importantly, these data reduction rules can be generalized, and they also work for the (α, β)-k-FEATURE SET for arbitrary values of $\alpha = 1$ and $\beta = 0$. This approach of data reduction followed by computation of an exact solution for the problem has led to a number of significant advances in areas of bioinformatics and molecular biology in the interpretation of large datasets, see [22–24, 55, 56, 121, 147, 149, 200, 216, 217, 252, 260, 261, 271, 276].

7.5.5 Source of Structure in Networks: Second Act, Motifs, and the Network Alignment Problem

Earlier in this chapter (Sect. 7.1) we have seen two case studies of problems in networks of historical interest, the "Königsberg bridges puzzle" proposed to Leonhard Euler and the problem of understanding the frequent "runaway problem" that Moreno and Jennings dealt with in the 1930s. There is a common issue among them: in both cases the researchers intended to identify a *subgraph* of the graph given as input. In addition, although the requirements were different, in both cases it was a path. This said, it is not surprising that many problems involve not only the identification of sets of edges and/or vertices, but actually subgraphs of particular interest. These subgraphs are generally called *motifs* [325].

These subgraphs may give important clues about how the graphs or networks have originated from some underlying process that are responsible for the creation of these structures. Since these processes may "grow" the networks through extended periods of time [54], some investigators believe that the identification of very common subgraphs present on these networks can give us information about the nature of these processes and the putative functions involved (e.g., [21, 38, 68, 218, 326]). There are many researchers working in areas that involve algorithms for *motif identification* and/or *motif counting* [28, 88, 280, 281]. Applications involve problems in the area of images [279], query expansion [116], time series analysis [140], information diffusion [17], co-purchasing networks [295], purchase patterns, [141] pattern mining in relational graphs [64], manipulated stocks in trading networks [300], and in event networks available from crime databases [60]. The list of manuscripts on network and graph motif identification is very extensive and we can probably not honour all the many works currently available.

7.5.5.1 The Subgraph Isomorphism Problem

When we are interested in finding a particular subgraph within a larger subgraph, this problem is known as *the subgraph isomorphism problem* (SIP). In this case, we are given two graphs G and M and we are asked to determine if G contains Q as a subgraph (formally speaking, if G contains a subgraph that is isomorphic to Q). This is a very important problem with many applications. When Q is a clique, this is basically the *maximum clique problem*, so we say that SIP is a generalization (since Q could be any type of graph). For instance, Figs. 1.3 and 7.12a are actually the same graph (and even the dominating set is the same). The reader may easily verify that by inspection (hint: a rotation by 90° and a different layout will help to see that they are actually the same graph) but in order to guarantee this mathematically we need an exact algorithm.

Part of the increased research in "network motifs" is another manifestation of the interest of this very basic problem that is known for many years. For many researchers the graph of interest Q is just composed of a few vertices and edges. When we relax the isomorphism requirement, and we ask to maximize some metric, we have the more general *network alignment problem*. This problem is discussed in this collection of works, so we refer to Chap. 12 for more information about it and its application in a problem on the comparative analysis of results of surveys.

7.6 Parameterized Complexity and the Search of Useful Structures

The reader may now be convinced that the area of business and customer analytics is rich in graphs and network problems that are NP-hard. This is generally used as an "excuse" for attempting to use heuristics and metaheuristics. However, for some NP-hard problems, it is possible to identify a part of the input that is somehow bounded. For instance, in the previous problem of subgraph isomorphism problem, we could be bounded the size of the "query graph" Q. If Q is relatively "small" and bounded, would it be possible to solve the problem to optimality in a time polynomial in the size of G ?

It is now known that for some "parameterized" versions of these NP-hard problems it is possible to prove that a *fixed-parameter tractable* algorithm exists [75, 76, 236, 272] or you can prove that it is *most likely* that no algorithm of that type might exist (some examples coming from research linked these topics can be found in [55, 167, 272]). The computer science area that studies this particular topic is called *parameterized complexity*.

When one or several parameters in the input data are known, a computational problem can be solved in polynomial-time in the instance size while the running-time dependence in the parameter is an arbitrary function. We have "a good deal" then, the combinatorial running-time "explosion" (inherent to most exact algorithms for intractable problems) has now been "confined to the parameters", which in turn explains the denomination "fixed-parameter tractable" for this type of algorithms. This class of problems is called FPT.

The purpose of this section is then to give an introduction to this topic that, we think, may motivate some of our readers to consider this approach. It is, however, an advanced topic but it is taught in some undergraduate degrees in computer science around the world. We thus felt it was proper to have some words of advice about it since one of the chapters on network analytics also addresses a problem (i.e., network alignment) for which its hardness, in terms of parameterized complexity, has been established for the first time. This section then "closes the gap" with the necessary reading which may be required to grasp some of the main concepts in this area.

The other important reason to discuss this topic is that the development of fixed-parameter tractable algorithms involves, generally speaking, the combination of techniques for safe data reduction of the dimensionality of the problem followed by an exhaustive search for an optimal solution. In some sense this area formalizes these ideas in a coherent framework for addressing large instances of combinatorial problems. The new power we can have by the combination of techniques can be interpreted as a "deal" of an unusual type. The next subsection is then dedicated to it.

7.6.1 Dealing with "The Devil" of Intractability

Computer scientists like to talk about "The Devil". It is well known that "The Devil" has many faces. One of these faces is the one recognized by Vladimir Vapnik, a world pioneer in the theory of statistical learning. He had a warning about the recent successes of *Deep Learning* methods. In that context, Vapnik has been quoted as having put forward the notion that: *"ideas and intuitions come either from God or from The Devil"*. He suggested that *"The Devil appeared always in the form of brute force"* [185]. In essence, "The Devil" may tempt a researcher to use algorithmic approach with early successes which, in fact, may later lead to very complex, and potentially unsolvable research questions which would appear further on the track. Successes may occur for small problems, but the final outcome is that you have been given an idea which is intrinsically related to do some sort of brute force search, and it will not pan out well in the long run.

Like Vapnik, Rodney Downey, and Michael Fellows are also two computer science pioneers that considered that the research field they have championed, *parameterized complexity*, is actually *"a deal with The Devil of Intractability"*. Are they talking about the same demon? We recapitulate on a few assumed notions. Computer scientists are naturally lured to these *intractable problems* since they naturally found them very interesting. For some problems, it is known that no efficient algorithm exist, for others no efficient algorithm *might* exist. For both types, it is clear that searching for the optimal solution involves some sort of "combinatorial explosion". For some problems, some parameters can be assumed fixed. Think, for instance, on the *k-Feature Set* problem. We may ask ourselves, what if k is fixed? What if k is fixed to a *small* value? Would that give us some computational advantage? After all, when we are looking for motifs, for instance, we know that the motifs may be small subgraphs of very large graphs. Would that make the problem "easier"? Would efficient safe data reduction techniques with exhaustive search techniques on the reduced problem be useful?

As you may remember, the problems of interest are NP-Complete, so here is the deal with The Devil: we drop our quest of having an efficient algorithm for those problems…and we accept that we will need to do some brute force or an implicitly enumerative complete search. The Devil will allow us to have some safe data reduction techniques, but it is our job to find them. However in this deal The Devil has given us something; some of these intractable problems may have subproblems; sets of instances for which the "combinatorial explosion" is somewhat confined. In networks, for instance, we already know that they have some properties, like the *genus*. Perhaps the explosion is only a function of the genus and not of the instance size. We will then feel better with ourselves and ready to strike a deal.

Actually, for many problems it may make sense to strike such a deal. And it may not be unique. Several network properties are actually characterized by some small numbers. We have seen before that motifs of interests may be bounded in size, so perhaps there are problems that are characterized by the need of a bounded brute search algorithm which is only related to the size of the motifs [26, 27, 88, 145]. The field of parameterized complexity is now on a three-decades long quest on characterizing these types of deals with "The Devil of Intractability".

7.6.2 Why is it Relevant to Study FPT Problems in Graphs and Networks?

Understanding where the boundaries of our deal with the Devil of Intractability are is important. It is difficult to articulate a more complete defence on why research on fixed-parameter tractability is relevant, and parameterized complexity is in general, than the one available in some of the more recent articles of the topic. See, for instance, [73, 235], and particularly [72], the introduction of [74] and the new and updated work [75]. It is estimated that "about half" of the naturally parameterized problems catalogued as NP-Complete in the classic book by Garey and Johnson [208] are in FPT, including three of the six basic problems singled out for attention in Chapter 3 of [208]. Many graph problems have some parameterized variant that is in FPT.

In problems in the business and data analytics space, the integer numbers that are associated with the parameters are sometimes small (e.g., remember Weihe's comment about the low genus of a railway transportation network). Logic and database problems are frequently defined as having an input consisting of a formula (which may be small and relatively invariant), and some other structure (such as a database) which is typically quite large and changeable. Formula size, or other aspects of formula structure may be a relevant parameter [335]. In production planning the number of processors or machines to be scheduled may be bounded by a values such as $k \leq 20$. The same can be said for a fleet of trucks in charge of transport of some goods, the number may be bounded and relatively small in comparison with the number of products and clients to serve. The number of degrees of freedom in a robot motion-planning problem is commonly in the range $k \leq 10$. This said, we may say that in many real-world scenarios "parameterization" is the rule and not the exception.

7.6.3 The Basic and Formal Definitions of Fixed-Parameter Tractability

A parameterized problem can be generally viewed as a problem that has as input two components, i.e., a pair $\langle x, k\rangle$. The former is generally an instance (i.e., $x \in I_P$) of some other decision problem P and the latter is some numerical aspect of the former (generally a positive integer assumed $k \ll |x|$) that constitutes a *parameter*. For a maximization (resp. minimization) problem P, the induced language $L_P(param)$ is the parameterized language consisting of all pairs $\langle x, k\rangle$ where $x \in I_P$ and $opt_P(x) \geq k$ (resp. $opt_P(x) \leq k$). If there exists an algorithm solving $L_P(param)$ in time $O(f(k)|x|^\alpha)$, where $|x|$ is a measure of the instance size, $f(k)$ an *arbitrary* function depending on k only, and α a constant independent of k or n, the parameterized problem is said to be *fixed-parameter tractable* and the language recognition problem belongs to the computational complexity class FPT . Though it is surprising, although the argument is not hard, we remark that *FPT* is unchanged if the definition above is modified by replacing $f(k)|x|^\alpha$ by $f(k) + |x|^\alpha$ [42].

A Historical Note: The Successes on k-VERTEX COVER
Any new idea, if also supported by considerable progress in one area, leads to the immediate interest of researchers if it is a reasonable alternative. This happened with a problem we have covered in the first chapter of this collection of works, the k-VERTEX COVER. Here we give some historical notes on how it helped to attract the interest of researchers while, at the same time, there was no progress with approximation algorithms for this problem. It also gives some insights at how the theory developed in one particular problem and the references can give access to the interested readers to follow the particular techniques in many other problems.

The definition of the FPT class allows $f(k)$ to grow arbitrarily fast. For PLANAR k-DOMINATING SET algorithms that have $f(k) = 11^k$ or $f(k) = 2^{11.98\sqrt{k}}$ are known, but for other problems, like k-VERTEX COVER this situation is perhaps remarkably better. We can discuss this problem as it is one of the most emblematic of the FPT class.

k-VERTEX COVER
Instance: A graph $G(V, E)$ and a positive integer k.
Parameter: k.
Question: Is there a set $V' \subseteq V$, such that for every edge $(u, v) \in E$, at least u or v is a member of V' and $|V'| \leq k$?

(continued)

In general, efficient fixed-parameter tractable algorithms are based on the techniques of *reduction to a problem kernel* and *bounded search trees*. The reader interested in a shortcut to understand these methods may find a paper by Chen, Kanj, and Yia appropriate. They show how k-VERTEX COVER can be solved in time $O(1.271^k k^2 + kn)$ [47].

On the positive side, a new general method to improve FPT algorithms that run on time $O(q(k)\beta^k + p(n))$ (where p and q are polynomials and β is a small constant) has been developed by Niedermeier and Rossmanith [239]. Their method interleaves the reduction to a problem kernel and bounded search methods. If it is the case that β^k is the size of the bounded search tree and $q(k)$ is the size of the problem kernel, the new technique is able to get rid of the $q(k)$ factor allowing the algorithm to be $O(\beta^k + p(n))$. With this method, the algorithm presented by Chen et al. [47] can be improved leading to an algorithm that solves the parameterized version of VERTEX COVER in time $O(1.271^k + n)$.

In contrast with the lack of improvement on approximation algorithms, somewhat recently justified by the hardness of approximation result, there have been systematic advances in the development of fixed-parameter tractable algorithms for this problem. We summarize here the main achievements for the MIN VERTEX COVER problem. Let $n = |V|$, the number of vertices of the graph.

- Fellows and Langston, in 1986, first observed that the *graph minor's* theory developed by Robertson and Seymour can be used to conclude that for each fixed value of k, the k-VERTEX COVER problem can be solved in time $O(f(k)n^3)$ [87].
- D.S. Johnson observed in 1987 [148] that an $O(f(k)n^2)$ algorithm for k-VERTEX COVER could be based using a combination of *tree-decomposition* and finite-state dynamic programming techniques.
- In 1988, Fellows gave an algorithm for k-VERTEX COVER based on a search tree technique [86] with a time complexity $O(2^k n)$.
- In 1989, Buss developed an algorithm that run in $O(kn + 2^k k^{2k+2})$, see [40]. The method was based on what we know now as *reduction to problem kernel*. It has been further developed as a general method in [71].
- In 1992, Balasubramanian, Downey, Fellows, and Raman combined ideas from [86] and those developed by Buss. Then Balasubramanian, Fellows, and Raman in [15] reported these findings and a new algorithm with running time $O(kn + 2^k k^2)$.
- Using maximum matching as a subroutine, Papadimitriou and Yannakakis presented an algorithm of $O(3^k n)$ time complexity in 1993 [245]. They also noted that the fact that this yields a polynomial-time algorithm for

(continued)

VERTEX COVER in those cases where k is $O(logn)$ [246]. See also the work by Bonet et al. [32].

- In [15] Balasubramanian, Fellows, and Raman described an algorithm that has a running time of $O(kn + (4/3)^k k^2)$ based on combination and refinements of previous methods.
- In 1999, the $O(kn + 1.324718^k k^2)$ algorithm from [15] was improved to $O(kn + 1.31951^k k^2)$ in [73].
- Niedermeier and Rossmanith improved the result to $O(kn + 1.29175^k k^2)$ [238].
- Finally, the two results we have mentioned before, Chen, Kanj, and Yia improved the result even further to $O(kn + 1.271^k k^2)$ [47]. Combined with the general method introduced in [239] this would give an impressive $O(1.271^k + n)$, i.e., linear in n for fixed k, and polynomial in n for $k \in O(\log n)$.

7.6.4 When a Problem May be Not in Class FPT

There was good progress in our deal with the Devil, at least for the k-VERTEX COVER as the historical note above indicates. In comparison with other approaches, this area has produced notable progress over the two first decades since its establishment in the late 1980s. A natural question is then: could it be possible that for some parameterized problem we can prove that there is no fixed-parameter tractable algorithm?

In this section, we briefly give some explanation on the role that some transformation between problems has in defining that a problem may be belonging to computational complexity classes for which it is highly unlikely that a fixed-parameter tractable algorithm exists. We see in the chapter of network alignment on such a transformation. This section presents the basic framework.

7.6.4.1 Parameterized Transformations, the W-Hierarchy, and the Complexity of k-FEATURE SET

The key elements that "bind" or "create" these computational complexity classes are based on a membership test and some sort of transformation (a process that converts a generic instance of a problem into a particular instance of another one). A particular one is needed in this case.

Parametric transformations between problems/languages differ in some key characteristic from ordinary polynomial-time reductions between combinatorial problems to prove NP-hardness. An example of a polynomial-time reduction is the one used to prove that a problem amply discussed in the first chapter of this collection, k-Feature Set, is NP-hard [59]. The reduction is from k-Vertex Cover problem and it shows that any instance of k-Vertex Cover can be transformed in a subproblem of the k-Feature Set in which only one sample belongs to the other class. Since the k-Feature Set in class NP, the two results together prove that the problem is NP-Complete as shown by Davies and Russell [59]. A parameterized transformation from the k-Dominating Set problem was used to prove that k-Feature Set belongs to the class of problems called $W[2]$-hard, and together with a proof of membership to the class $W[2]$, Cotta and Moscato proved it is $W[2]$-complete [55]. Without entering into detail of what this class is, at this stage, what this means is that, under the current assumptions about the nature of these computational complexity classes, a problem like k-Feature Set may not have a fixed-parameter algorithm (i.e., it is not in class FPT).

This will require a bit of extra formality and some definitions are provided below, although the whole discussion is an advanced topic for which we suggest Refs. [74] and [75].

We aim now to give some insights on how parameterized problems can influence our decision of how to address them with algorithmic means. For problems of graphs and networks, the existence of parameterized complexity creates an interesting divide. Perhaps the following is the one that illustrates the situation in its simplest way. For instance, if $G = (V, E)$ is a graph on $|V| = n$ vertices, a set of vertices $V' \subseteq V$ is a k-clique in G if and only if $V - V'$ is a vertex cover in the complementary graph G' (in G' two vertices are adjacent if and only if they are not adjacent in G). This gives an easy polynomial-time reduction of the naturally parameterized k-Clique problem to the naturally parameterized k- k-Vertex Cover problem, transforming the instance (G, k) of k-Clique into the instance (G', k') of k'-Vertex Cover. However, this is not a parametric transformation, since $k' = n - k$ is not purely a function of k.

The current conjecture is that there is no parametric transformation in this direction thus the problems k-Clique and k-Vertex Cover would be in different classes (one is in class W[1] and the other is in class FPT, respectively). We now formally define.

Definition A *parametric transformation* from a parameterized language L to a parameterized language L' is an algorithm that computes from input consisting of a pair (x, k), a pair (x', k') such that:

1. $(x, k) \in L$ if and only if $(x', k') \in L'$,
2. $k' = g(k)$ is a function only of k, and
3. the computation is accomplished in time $f(k)n^{\alpha}$, where $n = |x|$, α is a constant independent of both n and k, and f is an arbitrary function.

There exists a fairly elaborate parametric transformation from the naturally parameterized Clique problem to the naturally parameterized Dominating Set problem, mapping (G, k) to (G', k') where $k' = 2k$ [70, 74]. Again, all the evidence apparently indicates that there is no such parametric transformation in the other direction, this indicates that DOMINATING SET would be in yet a different class. $W[1]$ is then the class to which CLIQUE belongs to, and DOMINATING SET belongs to $W[2]$.

The essential property of parametric transformations is that if a problem P transforms in this way to a problem P' and $P' \in FPT$, then $P \in FPT$.

This naturally leads to a completeness program based on a hierarchy of parameterized problem classes:

$$FPT \subseteq W[1] \subseteq W[2] \subseteq \cdots \subseteq W[SAT] \subseteq W[P] \subseteq AW[P] \subseteq XP.$$

It is not necessary at this stage to delve into the details of what these classes are. We just want to clarify that there is a deep theory here and that the analogue of the question if $P \neq NP$ is, for the parameterized complexity setting, to finally settle the question: *"Is $W[1] \neq FPT$?"*.

Currently a proof of $W[1]$-hardness is the basic evidence that suggests that a parameterized problem is likely not to be fixed-parameter tractable.

Consequently, parameterized complexity allows a certain kind of "dialogue" with a single problem. After proving that a problem is $W[1]$-hard, the next natural step is to find if there are versions of it, using some different parameterization, that can be in the FPT class. Note that it is always possible to parameterize a problem in various ways that are fixed-parameter tractable, so alternatively it is not surprising that many parameterized problems apparently *do not* belong to *FPT*.

The naturally parameterized DOMINATING SET problem defined above, in which the parameter is the cardinality of the dominating set, is one of these. However, PLANAR DOMINATING SET is in FPT, but PLANAR CAPACITATED DOMINATING SET is W[1]-hard [30], so planarity (i.e., $genus = 0$) alone is not a panacea to make a problem fixed-parameter tractable. If we need to find structures that may impact on how to build efficient algorithms, we need to consider other parameters as well.

7.7 Treewidth and Other Structural Parameters of Graphs

We have given some detail on the theory of parameterized complexity so that the readers can have an idea of the interest in linking the structure of the networks with a tight result that can predict the existence of a fixed-parameter algorithm or the "unlikeness" to find one (e.g., by proving that a problem is $W[1]$-hard).

One very important graph parameter is called *treewidth*, first called "dimension" by Umberto Bertelé and Francesco Briosch in 1972 [25]. Graphs with a treewidth at most k are also called a *partial k-tree*. This said, it is important then to introduce a new type of decomposition of graphs that reveal information about their structure.

7.7.1 Tree Decomposition

One way that helps to understand what treewidth means is by introducing the *tree decomposition* of a graph $G = (V, E)$. It is defined as being a new graph which is a tree $T(V', E')$, in which each vertex of V' represents as subset of the set of vertices of G (i.e., for all vertices $v' \in V'$ there is an assignment to a subset of vertices X' of V), satisfying:

- The union of all vertices in all subsets X' equals V.
- If two sets X_i and X_j contain a node $v \in V$, then all vertices of T in the (unique) path between X_i and X_j represent subsets of V that also contain v.
- For every edge $(v, w) \in E$, there is a subset X_i that contains both v and w.

The *width* of a tree decomposition is the size of its largest set X_i minus one and the *treewidth* of a graph G is the minimum width among all possible tree decompositions of G.

It is NP-Complete to decide if a given graph G has a treewidth which is at most a given value k [11]. The same can be said about another interesting parameter, *cliquewidth* [89].

7.7.2 Treewidth Links to Other Structural Parameters of Networks

Following Harvey and Wood [122] we introduce the concept of tiredness of graph parameters. A parameter, treewidth for instance, can be understood as a real valued function α defined on all graphs such that $\alpha(G_1) = \alpha(G_2)$ whenever G_1 and G_2 are isomorphic. Two graph parameters α and β are said to be *tied* if there exist a function f such that for every graph G $\alpha(G) \le f(\beta(G))$ and $\beta(G) \le f(\alpha(G))$. When f is a polynomial we say that α and β are *polynomially tied.*

A number of graph parameters are then polynomially tied to treewidth like the *branchwidth* [90, 288], *bramble number* [115, 291], *separation number* [122], *tangle number* [114, 129], etc. We refer to [122] for the complete list.

7.7.3 The k-Truss of a Graph and Word-of-Mouth Marketing Strategies

We have seen in Sect. 7.4 that a triangle-based definition of a combinatorial optimization problem in graphs leads to a mathematical model that is able to extract interesting "communities" of vertices (subgraphs that are densely connected but they are not necessarily cliques). The price we still have to pay to identify these communities is that the associated problem is NP-hard, this limits the usefulness of exact algorithms for large graphs such as those of real-life social networks. This means that we need to probably need to look for other alternatives for large graphs [138]. If we do not want to resort to heuristics, or if we want to accelerate local search algorithms which can be heuristics for the NP-hard problem, we could look at the possibility of finding other types of graph structures which are computable in polynomial-time.

One such approach is to identify either a *k-truss* [316], or perhaps even better, to generate the full *k-truss decomposition* of the graph. The k-truss decomposition is then likely to be a useful structure to consider when designing "word-of-mouth" marketing strategies for the promotion of products or services in social networks. Most of the approaches discussed in this chapter could benefit from computing this structure a priori and guide the selection of targets.

It has been recently proposed that vertices belonging to the maximal k-truss subgraph of a graph help to identify the most influential vertices in spreading information processes in them. In addition, algorithms for graphs that are too large to be stored have been recently proposed [346] and parallel algorithms also exist [48, 150, 150], pointing at a very new promising research direction. What are these structures?

Given a graph G, we will define the k-truss of G as the largest subgraph of G in which every edge is contained in at least $(k - 2)$ triangles within the subgraph. Accordingly, a *truss decomposition* of G is the set of all the (non-empty) k-trusses of G for all integers $k \geq 2$.

We have chosen to illustrate with two examples k-truss decompositions by referring again to the dataset of economic activities in the construction sector of Porto Empedocle and the complete social network of dolphin interactions. Both datasets have been previously discussed in the chapter and the truss decompositions are shown in Figs. 7.16 and 7.17. The *truss number* of an edge $e \in E$ in a graph $G(V, E)$ is the number of triangles in G that contain e. The *k-class* of G is then the set of edges that have a truss number equal to k. This means that the 2-truss of G is G itself. In fact, we have a hierarchical decomposition of the graph edges set since any edge is that is of a l-class is always an edge of an l'-class for all l' integer $2 \leq l' < l$.

It is also clear that maximal cliques are more likely to be in subgraphs that have the edges with the maximal class values, so these individuals do not all need to be targeted with advertising at the same time. In addition, the decomposition would then help to identify another objective for viral marketing strategies. Not only the degree centrality of a vertex is of interest, but the total sum of the maximum class value attained by its incident edges would indicate their relevance in "feedback loops" of word-of-mouth interactions.

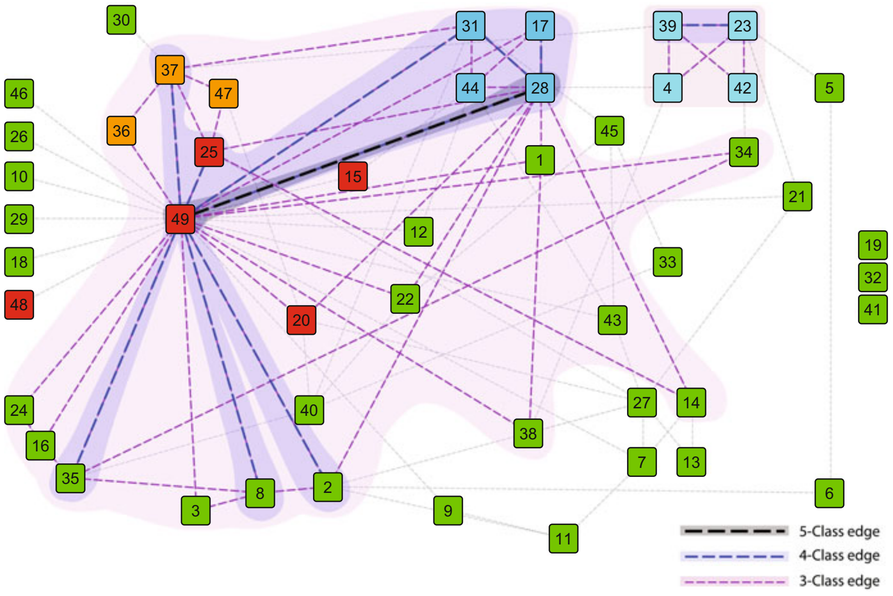


Fig. 7.16 Computation of the k-trusses of the construction firms network of the Italian town of Porto Empedocle. We have preserved the layout of the nodes of Fig. 7.9. The picture shows a "contour plot" like the one from Fig. 2 from [316]. In this case, we observe that there is only one group of vertices that belong to the k-truss with $k = 3$

7.7.4 *Problems that Are Actually Hypergraph Problems and Are Fixed-Parameter Tractable*

To discuss another problem that belongs to the FPT class, we will first present its associated optimization problem, the MINIMUM HITTING SET Problem. It was proved to be equivalent to MIN SET COVER in [13].

MINIMUM HITTING SET

Instance: A collection C of subsets of a finite set S.

Solution: A hitting set, i.e., a subset $S' \subseteq S$, such that S' contains at least one element from each subset in C.

Objective Function: Cardinality of the hitting set, i.e., $|S'|$.

This means that approximation algorithms and nonapproximability results for MIN SET COVER carry over to MIN HITTING SET. The constrained variation in which the input is extended with a subset T of S, and the problem is to find the hitting set that contains the largest number of elements from T, is not approximable within $|S|^\epsilon$ for some $\epsilon > 0$. However, some special cases in which, given compact subsets of $\mathbb{R}^d$, the goal is to find a set of straight lines of minimum cardinality so that each of the given subsets is hit by at least one line, are approximable [124]. A description of

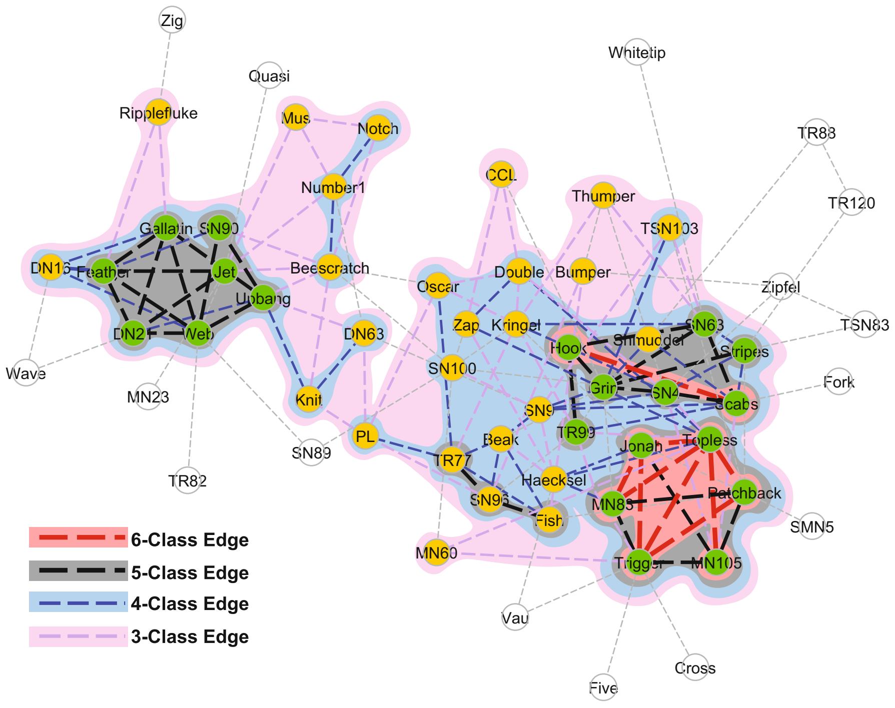


Fig. 7.17 Results of computing a truss decomposition on the complete dolphin interaction network with 62 individuals. The shaded areas represent a kind of "contour plot" that helps to illustrate the result of the calculation. A group of dolphins, {Topless, Trigger, MN83, Patchback, Jonah, MN105}, is connected by edges of the 6-class type. This group is a subset of the set presented in Fig. 7.6 as only "Haecksel" is missing. Note that "Hacksel" is connected via 4-Class edges to three nodes in this group. This indicates that generating a k-truss decomposition in polynomial-time could benefit a population-based search optimization methods by generating initial solutions and by making more effective the local search techniques. It can potentially help other memetic algorithms approaches for NP-hard problems in community structure identification and other graph optimization problems

some applications of the MIN HITTING SET problem can be found in a manuscript explaining how it had a role to solve a long-standing problem related to the Sudoku pastime [201]. The review includes the important application of a generalization of the problem that was used to query pharmacological databases for putative combinations of drugs that would be relevant in chemotherapy treatments [204].

A special-case of hitting set problem in its decision variant is:

HITTING SET FOR SIZE THREE SETS

Instance: A collection C of subsets of size three of a finite set S and an integer $k > 0$.

Parameter: k.

Question: Is there a subset $S' \subseteq S$, with $|S'| \leq k$ which allows S' contain at least one element from each subset in C ?

Interestingly, this problem is a hypergraph generalization of the VERTEX COVER problem. A hyperedge is defined between three vertices (instead of only two as in VERTEX COVER), and the task is to find a minimal subset of vertices covering all hyperedges. Niedermeier and Rossmanith have presented an algorithm that solves this problem in time $O(2.311^k + n)$ [237].

7.8 About Other Contributions in this Collection of Works

The work reviewed here helps us to introduce some of the other contributions in this section that relate to graphs and networks in the book. All these relate, in one way or another, with topics that involve eliciting some interesting structures present on the data, check if they exist, or exploit them to improve the user/consumer experience.

While some of the properties of networks depend on it as a whole (e.g., genus, treewidth, chromatic number, domination number, etc.), sometimes we are interested in identifying some properties of individual nodes. We have discussed one of them in relationship to Fig. 7.9a. Sergio Gómez chapter, "Centrality in networks: Finding the most important nodes", is dedicated to this subject. What we have seen in Fig. 7.9a is only one of the centrality criteria that others are reviewed in the contribution by Gómez. Computation of different measures using the same graph of interactions allows the readers to have an intuition of their possible applications.

In "Network-based models for social recommendation systems", Godoy-Lorite, Guimerà, and Sales-Pardo present a network-based approach for recommendation of products. The predictions of network-based models presented here outperform leading approaches for recommendation. The underlying assumption is that items and individuals can be put together in groups and that the preference of an individual for a particular item is determined only by its membership to a group. Stochastic Block Models are introduced so that predictions can be based on Monte Carlo sampling or Expectation-Maximization methods. They show the performance using several datasets including some which have up to 10 million individual ratings (from MovieLens).

In another contribution, Gabardo, Berretta, and Moscato present "Overlapping communities in co-purchasing and social interaction graphs: a memetic approach". While there is a section entirely dedicated to memetic algorithms in the book, this chapter is indeed in the "overlap" between these two sections. Methodologically, it responds to the need of faster and more efficient algorithms to uncover community structure in networks, and for this reason it is included here. The approach is based on the *"line graph"* of the original network. Any algorithm for non-overlapping communities can be transformed into one that allows overlaps via this approach. Using the Amazon's co-purchasing network the authors centre the study on one particular brand, *Versace*. The analysis of the communities revealed the connections between perfume and body products with other products like sunglasses. A community structure of brands revealed four major groups. The chapter gives insights on the application of the technique by illustrating the results on a dataset related to the TV show *"Game of Thrones"* characters (as generated from mapped interactions from the novel *"A Storm of Swords"*). This helps to compare the results with those presented in the chapter contributed by Sergio Gomez who uses the same network.

In "Using Network Alignment to Identify Conserved Consumer Behaviour Modelling Constructs", Mathieson, de Vries, and Moscato study the problem described in Sect. 7.5.5. This is possibly the first application of the network alignment problem in the social sciences. The authors build on a previous consumer behaviour study relating to pages of brands in a social media setting. Their contribution also gives a proof that the problem is W[1]-complete under certain parameterizations, thus these problems may not be fixed-parameter tractable (unless the class FPT=W[1] [75]). Using both simulated and real data on responses of consumers, and employing a memetic algorithm, the authors show that underlying behavioural "functional constructs" are conserved across different populations and surveys, as well as when the answers present reasonable variations in their natural expected ranges. Network alignment thus can help to "bridge the gap" across surveys that employ different questionnaires while trying to measure similar constructs.

Finally, in "Taming a Graph Hairball: Local Exploration in a Global Context", Abello, Mawhirter, and Sun present a graph decomposition into fixed points of degree peeling. Such a decomposition can be used as an exploratory graph data mining tool. The usefulness of the approach is illustrated using a network derived from the "On-Line Encyclopedia of Integer Sequences" and a product co-purchasing network from Amazon. The authors make a case for user interaction based on an exploration mechanism that employs global landmarks and local substructures present in the graph. Their approach is very general, and it is likely to have important applications to a large number of graph databases (e.g., of products and consumers) which currently lack a user-centred exploratory interface.

Acknowledgements P.M. acknowledges previous funding of his research by the Australian Research Council grants Future Fellowship FT120100060 and Discovery Project DP140104183. He also acknowledges previous support by FAPESP, Brazil (1996–2001). P.M. also thanks N.J. de Vries, M.N. Haque, and A.C. Gabardo for discussions and help in preparation of the figures, and Sergio Gómez for useful comments on earlier versions of this chapter and in particular in the area of multilayer networks.

Appendix: Software

We list here a set of some free software solutions for the analysis and processing of graphs that we have used in the past and may be worth checking when looking for top-of-the-shelf analysis methods.

- **Commetrix**[13]: A software framework for dynamic network visualization and analysis. It supports community extraction, real-time analysis, and visualization of dynamic networks, a better understanding of social network data and clustering of network graphs. An evaluation version of the software (valid for 30 days) is available for download on their webpage. However, the software can also be obtained free of charge after consideration of the user status and affiliation.
- **Cuttlefish**[14]: A software framework for visualizing networks. It also supports manipulation of network layouts using different popular algorithms (such as ARF, weighted-ARF, *k*-core, Kamada-Kawai, Fruchterman-Reingold, iso-m, circle, tree, and radial-tree). This Java-based applet tool can be incorporated into websites.
- **Cytoscape**[15]: A software initially designed for Bioinformatics applications, it becomes a popular general platform for graph/network analysis and visualization [285]. It provides an integrated environment for network analysis and visualization. In a nutshell, Cytoscape facilitates graph isomorphism, calculation of MST, network alignment, Social network connectivity analysis, graph triplet counter, topological algorithms, and many other functionalities through apps. An extensive range of apps available to use in Cytoscape, such as for visualization, network generation, graph analysis, clustering, layouts, and network comparisons. Since it was originally written in Java, it can run on all operating systems (Windows, Linux, Mac OS).
- **EgoNet**[16]: An open-source and free program for collection and analysis of egocentric social networks. It can help in creating questionnaires, data collection and generating network measures which can be exported to be used by other advanced network analysis tools. Java-based cross-platform executables are available for Linux, Mac OS, and Windows.
- **Gephi**[17]: This is a popular suite of tools that allow the exploration and visualization of graphs [18]. It can produce a real-time visualization of networks consisting of up to 100 thousands of nodes and million edges. The layout algorithms provide both the force-and-optimization-based state-of-the-art layout algorithms. In addition to these, it supports the calculation of statistics and metrics from networks. The program runs on all operating systems (Windows, Linux, Mac OS). Many uses, and extensible through plug-ins.
- **GraphChi**[18]: A disk-based large graph analysis software which is capable of analysing web-scale of complex networks, including those with billions of edges in a consumer standard laptop computer. It has graph contraction-based implementation for iterative graph algorithms for MST and minimum spanning

[13]Commetrix:http://www.commetrix.de.

[14]Cuttlefish:http://cuttlefish.sourceforge.net.

[15]Cytoscape:http://www.cytoscape.org.

[16]EgoNet:https://sourceforge.net/projects/egonet/.

[17]Gephi:https://gephi.org.

[18]GraphChi:https://github.com/GraphChi/graphchi-cpp.

forest (MSF), PageRank algorithm and also supports streaming graph updates for big graphs. Written in C++ (source code available), it has been tested on Mac OS X and Linux.

- **GraphStream**[19]: An open-source Java library for modelling and analysis for dynamic graphs. It provides various options for manipulation and analysis of large graphs. Connected Component, eccentricity, strongly connected components, PageRank are some of the featured graph analysis algorithms available in GraphStream library.
- **Graphviz**[20]: An open-source graph visualization software which uses textual description file system, in DOT language. It facilitates flexible options for graph drawing and visualizations. Some examples of features are drawing various types of graphs, visualization using radial, circular, tree and array-based layouts. Apart from visualization algorithms, it also supports graph analysis methods, such as directed acyclic graph, finding biconnected components, connected components, single-source distance filtering, counting graph components finding and augmenting clusters in a graph. The software supports scripting access through API using guile, Java, Perl, PHP, Python, Ruby, and Tcl languages. The source code of the program is available at https://gitlab.com/graphviz/graphviz/.
- **graph-tool**[21]: Efficient Python module for manipulation and statistical analysis of graphs [253]. Several graph algorithms such as isomorphism, subgraph, MST, connected components, and maximum flow are supported in graph-tool. It also provides the calculation of standard statistical graph measures and community structure detection algorithms.
- **GUESS**[22]: An open-source exploratory data analysis and visualization tool for graphs and networks [3]. It supports many popular layout algorithms such as Fruchtermen-Reingold, Kamada-Kawai, Sugiyama, Random, Radial, and Rescaling and rotating. Written in Java, it can run in all operating systems (Windows, Linux, Mac OS).
- **igraph**[23]: Collection of network analysis tools with the importance on efficiency, portability, and ease of use [57]. It is one of the biggest graph libraries written in C, which allows programming in R, Python, and C/C++. It has an extensive range of features, such as graph generation, conversion, manipulation, layout algorithms, calculation of structural statistic and measures, community detection, calculation of cohesive block and coreness, Graph Clustering, and motifs identification.
- **JUNG**[24]: Java Universal Network/Graph Framework (JUNG) is an extensible software library for modelling, analysing, and visualizing of graph data. It

[19]GraphStream:http://graphstream-project.org.

[20]Graphviz:https://www.graphviz.org.

[21]graph-tool:https://graph-tool.skewed.de.

[22]GUESS:http://graphexploration.cond.org.

[23]igraph:http://igraph.org.

[24]JUNG:https://github.com/jrtom/jung.

can generate random graphs, clustering algorithms, calculation of structural measures (such as centrality, PageRank, and HITS), and decomposition of graphs. Its visualization framework consists of Fruchterman-Reingold, Kamada-Kawai, Radial, Tree, Circle and Spring Layout.

- **MapEquation**[25]: Tools to simplify and highlight important structures in complex networks [79]. Centred on community detection using the Infomap algorithm. It has single and multi-level hierarchical layout algorithms for visualization. Written in C++, it can run on all operating systems (Windows, Linux, Mac OS).
- **MeerKat**[26]: A tool to visualize and analysis of social network. It includes several network analysis capabilities such as community detection, layout, and dynamic analysis of networks. Meerkat facilitates several state-of-the-art layout algorithms: such as Fruchterman-Reingold, Circle, Self-Organizing, and Kamada-Kawai. It also provides traditional graph metrics, or statistical information and well-known network measures like PageRank, HITS, Betweenness, and Centrality. The installer is available for Windows, Ubuntu, and Mac OS X.
- **MuxViz**[27]: Framework for the multilayer analysis and visualization of networks [61]. Although it is designed for multilayer networks, single layer networks can also be used as a valid input. It supports Fruchterman-Reingold, Kamada-Kawai visualization for single layer graph and supports topological descriptors calculation for both types of networks. Based on R and Octave, it can run on all operating systems (Windows, Linux, Mac OS).
- **NetMiner**[28]: A proprietary software for network visualization and analysis. It has an inbuilt Python-based automatic script generation that facilitates the work for some users. It supports built-in 3D network visualization and graph mining based on Machine Learning for reduction, classification, clustering, and recommendation. An academic time-limited licence is available to students, researchers, and teachers. The software only works on Windows operating system.
- **NetworKit**[29]: An open-source toolkit for large-scale network analysis. It contains the scalable implementation of graph algorithms to utilize multicore architecture. This Python-based toolkit used OpenMP with C++ for shared-memory parallelism. It has several node centrality measures such as degree, k-core decomposition, PageRank, and Betweenness. The partition algorithms are employed for community detection, connected component identification, and k-core decomposition. It also has several graph modelling and generation approaches. The software requires Linux based environment (such as Mac OS X and Ubuntu). However, the native Linux Subsystem supported by Microsoft is sufficient for using it in Windows 10.

[25]MapEquation:http://www.mapequation.org.

[26]MeerKat:https://www.amii.ca/meerkat/.

[27]MuxViz:http://muxviz.net.

[28]NetMiner:http://www.netminer.com.

[29]NetworKit:https://networkit.iti.kit.edu.

- **NetworkX**[30]: a Python software package which facilitates the creation, manipulation, and study of the structure, dynamics, and functions of complex networks [282]. It provides tools for social, biological, and other network analysis. It has a wide range of algorithms for approximation and heuristics (k-component, clique, clustering, dominating set, independent set, Steiner tree, vertex cover, etc.). NetworkX facilitates the computation of bi-partiteness (matching, projection, and centrality are few of them) and components (strong, weak, biconnected, semi-connectedness, etc.). Among other features we can mention connectivity with k-edge, k-node, disjoint path, min cut, etc.; cores decomposition using k-core, k-crust, k-corona, etc.; network flows and link analysis (PageRank, Hits) are some highlights.
- **Network Workbench**[31]: A toolkit for large-scale network visualization, analysis, and modelling specially designed for biomedical, social science, and physics research. In terms of network analysis, it supports Random, Watts-Strogatz Small world, random and scale-free methods for graph modelling; tree, balloon, radial, circular, Kamada-Kawai, Fruchterman-Reingold, Force-Directed, etc. layout algorithms; community detection, clustering, PageRank and other standard network analysis approaches. The installer is available for Windows, Linux, and Mac OS X.
- **NodeXL**[32]: An open-source template for Microsoft Excel capable of network visualization and analysis. Apart from general network visualization, it is capable of supporting text, sentiment, and social media stream data analysis. It is also capable of calculating network metrics like degree, centrality, PageRank, and clustering coefficients. Dynamic filtering, vertex grouping, Zooming, and scaling are some mentionable features available in the graph layout approaches.
- **Pajek**[33]: Analysis and visualization tool for Windows (can be run under Linux and Mac OS X using Wine) [19]. Includes the generation of random networks, the calculation of structural network descriptors, the operation of networks, community detection, etc. It also includes features for subnetwork extraction, searching for connected components, k-neighbours, max flow, community detection, generation different types of random networks. Several popular layout algorithms like Fruchterman-Reingold, Kamada-Kawai, FishEye transformation, etc. are available.
- **Radatools**[34]: Set of programs for the analysis of complex networks, with main attention to community detection and the finding of structural properties [108]. Written in Ada (source code available as Radalib[35]), it includes native executables for Windows, Linux, and MacOS. It includes the modularity and mesoscales

[30]NetworkX:http://networkx.github.io.

[31]Network Workbench: http://nwb.cns.iu.edu.

[32]NodeXL:https://nodexl.codeplex.com/.

[33]Pajek:http://mrvar.fdv.uni-lj.si/pajek.

[34]Radatools:http://deim.urv.cat/~sergio.gomez/radatools.php.

[35]Radalib: http://deim.urv.cat/~sergio.gomez/radalib.php.

approaches for community detection; weighted network conversion; computation of connected components, degrees, strengths, clustering coefficients, and so on. To support network manipulation functionalities, it provides subgraph extraction, network data type conversion, sorting of nodes based on degree, spanning tree computation, etc.

- **SNAP**[36]: General purpose, high performance system for analysis and manipulation of large networks [171]. Written in C++, yet can be accessed from C++ and Python. It can scale the algorithms to handle network consisting of hundreds of millions of nodes and billions of edge. It has a wide range of features for supporting different types of graphs, graph modelling features, graph manipulation, drawing, community detection, and analysis. Some of the notable features of them are subgraph conversion, connected component, clustering coefficient calculation, BFS and DFS and k-core decomposition.
- **SocNetV**[37]: Social Network Visualizer (SocNetV) is a cross-platform, free software for network analysis and visualization. It has rich feature set ranging from network visualization, layout modelling, network measures calculation, fast algorithms for community detection (clique and triad census) and built-in web crawler for creating the social network. It has embedded many popular social network datasets into the package, such as Zachary, Freeman, Mexican Power Network, and Petersen, and also supports several random network generation approaches. It features the calculation of cohesion metrices (Geodesics Matrix, Eccentricity, Symmetry Test, etc.), connectedness (strongly, weakly connected, and disconnected) and visualization layouts (circular, Spring Embedder, Fruchterman-Reingold, etc.). The source code (http://socnetv.org/downloads) and the installer are available for Windows, Linux, and Mac OS X.
- **Visone**[38]: Tool for the analysis and visualization of social networks [35]. It has features for calculation of node and link-level indices from the graph, such as centrality (Degree, Betweenness, Closeness, Current-flow centralities, Eccentricity, Eigenvector centrality, PageRank, etc.) and clustering coefficient for local density calculation. Written in Java, it can run on all operating systems (Windows, Linux, Mac OS).

Many other options for specific purposes exist, e.g., community detection algorithms, analysis of specific dynamics, benchmarking, etc. The previous list tries to cover only general purpose programs, but a simple online search is enough to discover other interesting tools. There are also complex network packages for proprietary platforms, such as Matlab, that we have not considered in the previous list. The integration of some tools with Python (igraph, NetworkX, graph-tool) and R (igraph, MuxViz) is very useful to expand their possibilities to analyse complex networks using high level programming.

[36]SNAP:http://snap.stanford.edu/snap.

[37]SocNetV:http://socnetv.org.

[38]Visone:https://www.visone.info.

References

1. Dolphins network dataset – KONECT (2016). http://konect.uni-koblenz.de/networks/dolphins
2. Abello, J., Resende, M.G.C., Sudarsky, S.: Massive quasi-clique detection. In: LATIN, *Lecture Notes in Computer Science*, vol. 2286, pp. 598–612. Springer (2002)
3. Adar, E., Kim, M.: Softguess: Visualization and exploration of code clones in context. In: Software Engineering, 2007. ICSE 2007. 29th International Conference on, pp. 762–766. IEEE (2007)
4. Akram, M., Dudek, W.A.: Intuitionistic fuzzy hypergraphs with applications. Information Sciences **218**, 182–193 (2013). https://doi.org/10.1016/j.ins.2012.06.024
5. Akram, M., Sarwar, M.: Transversals of m-polar fuzzy hypergraphs with applications. Journal of Intelligent and Fuzzy Systems **33**(1), 351–364 (2017)
6. Aloise, D., Caporossi, G., Hansen, P., Liberti, L., Perron, S., Ruiz, M.: Modularity maximization in networks by variable neighborhood search. In: D.A. Bader, H. Meyerhenke, P. Sanders, D. Wagner (eds.) Graph Partitioning and Graph Clustering, 10th DIMACS Implementation Challenge Workshop, Georgia Institute of Technology, Atlanta, GA, USA, February 13-14, 2012. Proceedings, *Contemporary Mathematics*, vol. 588, pp. 113–128. American Mathematical Society (2012). https://doi.org/10.1090/conm/588
7. Alonge, G., Ciancamerla, E., Mastrilli, A., Minichino, M., Nicotra, A., Ranno, M., Reali, M., Regina, P.: Towards energy efficiency of interdependent urban networks. IJSPM **11**(6), 529–546 (2016). https://doi.org/10.1504/IJSPM.2016.10001563
8. Amato, F., Moscato, V., Picariello, A., Sperlí, G.: Modelling multimedia social network for topic ranking. In: 2016 30th International Conference on Advanced Information Networking and Applications Workshops (WAINA), pp. 81–86 (2016). https://doi.org/10.1109/WAINA.2016.168
9. Amato, F., Moscato, V., Picariello, A., Sperlì, G.: Multimedia social network modeling: A proposal. In: Tenth IEEE International Conference on Semantic Computing, ICSC 2016, Laguna Hills, CA, USA, February 4-6, 2016, pp. 448–453. IEEE Computer Society (2016). https://doi.org/10.1109/ICSC.2016.20
10. Andre, R., Schlag, S., Schulz, C.: Memetic Multilevel Hypergraph Partitioning. ArXiv e-prints (2017)
11. Arnborg, S., Corneil, D.G., Proskurowski, A.: Complexity of finding embeddings in a k-tree. SIAM Journal on Algebraic Discrete Methods **8**(2), 277–284 (1987). https://doi.org/10.1137/0608024
12. Arthur, D., Motwani, R., Sharma, A., Xu, Y.: Pricing strategies for viral marketing on social networks. In: S. Leonardi (ed.) Internet and Network Economics: 5th International Workshop, WINE 2009, Rome, Italy, December 14-18, 2009. Proceedings, pp. 101–112. Springer Berlin Heidelberg, Berlin, Heidelberg (2009). https://doi.org/10.1007/978-3-642-10841-9_11
13. Ausiello, G., D'Atri, A., Protasi, M.: Structure preserving reductions among convex optimization problems. Journal of Computer and System Sciences **21**(1), 136–153 (1980). https://doi.org/10.1016/0022-0000(80)90046-X
14. Ausiello, G., Laura, L.: Directed hypergraphs: Introduction and fundamental algorithms - A survey. Theor. Comput. Sci. **658**, 293–306 (2017). https://doi.org/10.1016/j.tcs.2016.03.016
15. Balasubramanian, R., Fellows, M., Raman, V.: An improved fixed-parameter algorithm for vertex cover. Information Processing Letters **65**(3), 163–168 (1998)
16. Bampo, M., Ewing, M.T., Mather, D.R., Stewart, D., Wallace, M.: The effects of the social structure of digital networks on viral marketing performance. Information Systems Research **19**(3), 273–290 (2008). https://doi.org/10.1287/isre.1070.0152
17. Bao, P., Shen, H.W., Chen, W., Cheng, X.Q.: Cumulative effect in information diffusion: Empirical study on a microblogging network. PLOS ONE **8**(10), 1–7 (2013). https://doi.org/10.1371/journal.pone.0076027

18. Bastian, M., Heymann, S., Jacomy, M., et al.: Gephi: an open source software for exploring and manipulating networks. ICWSM **8**, 361–362 (2009)
19. Batagelj, V., Mrvar, A.: Pajek – program for large network analysis. Connections **21**(2), 47–57 (1998)
20. Benjamin, A., Chartrand, G., Zhang, P.: The Fascinating World of Graph Theory. Princeton University Press, Princeton, NJ, USA (2015)
21. Benson, A.R., Gleich, D.F., Leskovec, J.: Higher-order organization of complex networks. Science **353**(6295), 163–166 (2016). https://doi.org/10.1126/science.aad9029
22. Berretta, R., Costa, W., Moscato, P.: Combinatorial optimization models for finding genetic signatures from gene expression datasets. In: J.M. Keith (ed.) Bioinformatics: Structure, Function and Applications, pp. 363–377. Humana Press, Totowa, NJ (2008). https://doi.org/10.1007/978-1-60327-429-6_19
23. Berretta, R., Mendes, A., Moscato, P.: Integer programming models and algorithms for molecular classification of cancer from microarray data. In: Estivill-Castro [85], pp. 361–370. http://crpit.com/confpapers/CRPITV38Berretta.pdf
24. Berretta, R., Mendes, A., Moscato, P.: Selection of discriminative genes in microarray experiments using mathematical programming. Journal of Research and Practice in Information Technology **39**(4), 287–299 (2007). http://ws.acs.org.au/jrpit/JRPITVolumes/JRPIT39/JRPIT39.4.287.pdf
25. Bertele, U., Brioschi, F.: Nonserial Dynamic Programming. Academic Press, Inc., Orlando, FL, USA (1972)
26. Betzler, N., van Bevern, R., Fellows, M.R., Komusiewicz, C., Niedermeier, R.: Parameterized algorithmics for finding connected motifs in biological networks. IEEE/ACM Trans. Comput. Biology Bioinform. **8**(5), 1296–1308 (2011). https://doi.org/10.1109/TCBB.2011.19
27. Betzler, N., Fellows, M.R., Komusiewicz, C., Niedermeier, R.: Parameterized algorithms and hardness results for some graph motif problems. In: P. Ferragina, G.M. Landau (eds.) Combinatorial Pattern Matching, 19th Annual Symposium, CPM 2008, Pisa, Italy, June 18-20, 2008, Proceedings, *Lecture Notes in Computer Science*, vol. 5029, pp. 31–43. Springer (2008). https://doi.org/10.1007/978-3-540-69068-9_6
28. Björklund, A., Kaski, P., Kowalik, L.: Constrained multilinear detection and generalized graph motifs. Algorithmica **74**(2), 947–967 (2016). https://doi.org/10.1007/s00453-015-9981-1
29. Boccaletti, S., Bianconi, G., Criado, R., Del Genio, C.I., Gómez-Gardenes, J., Romance, M., Sendina-Nadal, I., Wang, Z., Zanin, M.: The structure and dynamics of multilayer networks. Physics Reports **544**(1), 1–122 (2014)
30. Bodlaender, H.L., Lokshtanov, D., Penninkx, E.: Planar capacitated dominating set is w[1]-hard. In: J. Chen, F.V. Fomin (eds.) Parameterized and Exact Computation: 4th International Workshop, IWPEC 2009, Copenhagen, Denmark, September 10-11, 2009, Revised Selected Papers, pp. 50–60. Springer Berlin Heidelberg, Berlin, Heidelberg (2009). https://doi.org/10.1007/978-3-642-11269-0_4
31. Bonchi, F., Castillo, C., Gionis, A., Jaimes, A.: Social network analysis and mining for business applications. ACM Transactions on Intelligent Systems and Technology (TIST) **2**(3), 22 (2011)
32. Bonet, M., Steel, M., Warnow, T., Yooseph, S.: Better methods for solving parsimony and compatibility. In: S. Istrail, P. Pevzner, M. Waterman (eds.) Proceedings of the 2nd Annual International Conference on Computational Molecular Biology (RECOMB-98), pp. 40–49. ACM Press, New York (1998)
33. Borradaile, G.: Planarity testing. In: Encyclopedia of Algorithms, pp. 1584–1585. Springer New York (2016)
34. Brandes, U., Delling, D., Gaertler, M., Görke, R., Hoefer, M., Nikoloski, Z., Wagner, D.: On modularity clustering. IEEE Trans. Knowl. Data Eng. **20**(2), 172–188 (2008). https://doi.org/10.1109/TKDE.2007.190689
35. Brandes, U., Wagner, D.: Analysis and visualization of social networks. Graph drawing software pp. 321–340 (2004)

36. Bretto, A.: Applications of Hypergraph Theory: A Brief Overview, pp. 111–116. Springer International Publishing, Heidelberg (2013). https://doi.org/10.1007/978-3-319-00080-0_7
37. Brunato, M., Hoos, H.H., Battiti, R.: On effectively finding maximal quasi-cliques in graphs. In: LION, *Lecture Notes in Computer Science*, vol. 5313, pp. 41–55. Springer (2007)
38. Bruno, F., Palopoli, L., Rombo, S.E.: New trends in graph mining: Structural and node-colored network motifs. IJKDB **1**(1), 81–99 (2010). https://doi.org/10.4018/jkdb.2010100206
39. Bunke, H., Dickinson, P., Kraetzl, M., Neuhaus, M., Stettler, M.: Matching of hypergraphs — algorithms, applications, and experiments. In: H. Bunke, A. Kandel, M. Last (eds.) Applied Pattern Recognition, pp. 131–154. Springer Berlin Heidelberg, Berlin, Heidelberg (2008). https://doi.org/10.1007/978-3-540-76831-9_6
40. Buss, J., Goldsmith, J.: Nondeterminism within P. SIAM Journal on Computing **22**, 560–572 (1993)
41. Cafieri, S., Hansen, P., Liberti, L.: Improving heuristics for network modularity maximization using an exact algorithm. Discrete Applied Mathematics **163**, 65–72 (2014). https://doi.org/10.1016/j.dam.2012.03.030
42. Cai, L., Chen, J., Downey, R., Fellows, M.: On the parameterized complexity of short computation and factorization. Archives for Math. Logic **36**, 321–337 (1997)
43. Cao, S., Dehmer, M., Kang, Z.: Network entropies based on independent sets and matchings. Applied Mathematics and Computation **307**(Supplement C), 265 – 270 (2017). https://doi.org/10.1016/j.amc.2017.02.021
44. Carrabs, F., Cerrone, C., Cerulli, R.: A memetic algorithm for the weighted feedback vertex set problem. Networks **64**(4), 339–356 (2014). https://doi.org/10.1002/net.21577
45. Chalupa, D.: Construction of near-optimal vertex clique covering for real-world networks. Computing and Informatics **34**(6), 1397–1417 (2015). http://www.cai.sk/ojs/index.php/cai/article/view/1276
46. Chawla, S., Davis, J.G., Pandey, G.: On local pruning of association rules using directed hypergraphs. In: ICDE, p. 832. IEEE Computer Society (2004)
47. Chen, J., Kanj, I., Jia, W.: Vertex cover: further observations and further improvements. In: Proc. 25th Int. Worksh. Graph-Theoretic Concepts in Computer Science, no. 1665 in Lecture Notes in Computer Science, pp. 313–324. Springer-Verlag (1999). http://www.cs.tamu.edu/faculty/chen/wg.ps
48. Chen, P., Chou, C., Chen, M.: Distributed algorithms for k-truss decomposition. In: J.J. Lin, J. Pei, X. Hu, W. Chang, R. Nambiar, C.C. Aggarwal, N. Cercone, V. Honavar, J. Huan, B. Mobasher, S. Pyne (eds.) 2014 IEEE International Conference on Big Data, Big Data 2014, Washington, DC, USA, October 27-30, 2014, pp. 471–480. IEEE (2014). https://doi.org/10.1109/BigData.2014.7004264
49. Chen, W., Wang, C., Wang, Y.: Scalable influence maximization for prevalent viral marketing in large-scale social networks. In: KDD, pp. 1029–1038. ACM (2010)
50. Chou, Y.H., Wang, E.T., Chen, A.L.P.: Finding maximal quasi-cliques containing a target vertex in a graph. In: DATA, pp. 5–15. SciTePress (2015)
51. Christakis, N.A., Fowler, J.H.: The spread of obesity in a large social network over 32 years. New England Journal of Medicine **357**(4), 370–379 (2007). https://doi.org/10.1056/NEJMsa066082. PMID: 17652652
52. Clark, M.B., Johnston, R.L., Inostroza-Ponta, M., Fox, A.H., Fortini, E., Moscato, P., Dinger, M.E., , Mattick, J.S.: Genome-wide analysis of long noncoding rna stability. Genome Research **22**, 885–898 (2012)
53. Conte, A., Firmani, D., Mordente, C., Patrignani, M., Torlone, R.: Fast enumeration of large k-plexes. In: KDD, pp. 115–124. ACM (2017)
54. Conway, D.: Modeling network evolution using graph motifs. CoRR **abs/1105.0902** (2011). http://arxiv.org/abs/1105.0902
55. Cotta, C., Moscato, P.: The k-$_{\text{feature}}$ $_{\text{set}}$ problem is $W[2]$-complete. J. Comput. Syst. Sci. **67**(4), 686–690 (2003). https://doi.org/10.1016/S0022-0000(03)00081-3
56. Cotta, C., Sloper, C., Moscato, P.: Evolutionary search of thresholds for robust feature set selection: Application to the analysis of microarray data. In: G.R. Raidl, S. Cagnoni,

J. Branke, D. Corne, R. Drechsler, Y. Jin, C.G. Johnson, P. Machado, E. Marchiori, F. Rothlauf, G.D. Smith, G. Squillero (eds.) Applications of Evolutionary Computing, EvoWorkshops 2004: EvoBIO, EvoCOMNET, EvoHOT, EvoIASP, EvoMUSART, and EvoSTOC, Coimbra, Portugal, April 5-7, 2004, Proceedings, *Lecture Notes in Computer Science*, vol. 3005, pp. 21–30. Springer (2004). https://doi.org/10.1007/978-3-540-24653-4_3
57. Csardi, G., Nepusz, T.: The igraph software package for complex network research. InterJournal, Complex Systems **1695**(5), 1–9 (2006)
58. Danziger, M.M., Bashan, A., Berezin, Y., Shekhtman, L.M., Havlin, S.: An Introduction to Interdependent Networks, pp. 189–202. Springer International Publishing, Cham (2014). https://doi.org/10.1007/978-3-319-08672-9_24
59. Davies, S., Russe, S.: Np-completeness of searches for smallest possible feature sets. In: AAAI Symposium on Intelligent Relevance, pp. 37–39. AAAI Press (1994)
60. Davies, T., Marchione, E.: Event networks and the identification of crime pattern motifs. PLOS ONE **10**(11), 1–19 (2015). https://doi.org/10.1371/journal.pone.0143638
61. De Domenico, M., Porter, M.A., Arenas, A.: Muxviz: a tool for multilayer analysis and visualization of networks. Journal of Complex Networks **3**(2), 159 (2015). https://doi.org/10.1093/comnet/cnu038
62. De Masi, G., Iori, G., Caldarelli, G.: Fitness model for the italian interbank money market. Phys. Rev. E **74**, 066,112 (2006). https://doi.org/10.1103/PhysRevE.74.066112
63. Dehmer, M., Emmert-Streib, F., Mehler, A. (eds.): Towards an Information Theory of Complex Networks - Statistical Methods and Applications. Birkhäuser (2011). https://doi.org/10.1007/978-0-8176-4904-3
64. Desmier, E.: Co-evolution pattern mining in dynamic attributed graphs. (fouille de motifs de co-evolution dans des graphes dynamiques attribués). Ph.D. thesis, INSA de Lyon, Lyon - Villeurbanne, France (2014). https://tel.archives-ouvertes.fr/tel-01127630
65. Díaz, J., Petit, J., Serna, M.J.: A survey of graph layout problems. ACM Comput. Surv. **34**(3), 313–356 (2002). https://doi.org/10.1145/568522.568523
66. Didimo, W., Liotta, G., Montecchiani, F.: Network visualization for financial crime detection. J. Vis. Lang. Comput. **25**(4), 433–451 (2014). https://doi.org/10.1016/j.jvlc.2014.01.002
67. Do, L., Lauw, H.W., Wang, K.: Mining revenue-maximizing bundling configuration. PVLDB **8**(5), 593–604 (2015)
68. Dondi, R., Fertin, G., Vialette, S.: Finding approximate and constrained motifs in graphs. In: R. Giancarlo, G. Manzini (eds.) Combinatorial Pattern Matching - 22nd Annual Symposium, CPM 2011, Palermo, Italy, June 27-29, 2011. Proceedings, *Lecture Notes in Computer Science*, vol. 6661, pp. 388–401. Springer (2011). https://doi.org/10.1007/978-3-642-21458-5_33
69. Doshi, V., Shah, D., Médard, M., Effros, M.: Functional compression through graph coloring. IEEE Trans. Information Theory **56**(8), 3901–3917 (2010). https://doi.org/10.1109/TIT.2010.2050835
70. Downey, R., Fellows, M.: Fixed-parameter tractability and completeness I: Basic results. SIAM Journal on Computing **24**(4), 873–921 (1995)
71. Downey, R., Fellows, M.: Parameterized computational feasibility. In: P. Clote, J. Remmel (eds.): Feasible Mathematics II, pp. 219–244. Boston: Birkhäuser (1995)
72. Downey, R., Fellows, M., Stege, U.: Computational Tractability: The View From Mars. Bulletin of the European Association for Theoretical Computer Science **69**, 73–97 (1999). http://www.neci.nj.nec.com/homepages/fortnow/beatcs/column69.ps
73. Downey, R., Fellows, M., Stege, U.: Parameterized Complexity: A framework for systematically confronting computational intractability. In: Contemporary Trends in Discrete Mathematics: Form DIMACS to DIMATIA to the future, AMS-DIMACS Proceedings Series, pp. 49–99. AMS (1999)
74. Downey, R.G., Fellows, M.R.: Parameterized Complexity. Monographs in Computer Science. Springer (1999). https://doi.org/10.1007/978-1-4612-0515-9
75. Downey, R.G., Fellows, M.R.: Fundamentals of Parameterized Complexity. Texts in Computer Science. Springer (2013). https://doi.org/10.1007/978-1-4471-5559-1

76. Downey, R.G., Fellows, M.R., Stege, U.: Computational tractability: The view from mars. Bulletin of the EATCS **69**, 73–97 (1999)
77. Du, Y., Wang, J., Li, Q.: An android malware detection approach using community structures of weighted function call graphs. IEEE Access **5**, 17,478–17,486 (2017). https://doi.org/10.1109/ACCESS.2017.2720160
78. Ducournau, A., Bretto, A.: Random walks in directed hypergraphs and application to semi-supervised image segmentation. Computer Vision and Image Understanding **120**, 91–102 (2014)
79. Edler, D., Rosvall, M.: The mapequation software package (2013)
80. Eguchi, K., Murata, T.: Constrained community detection in multiplex networks. In: G.L. Ciampaglia, A.J. Mashhadi, T. Yasseri (eds.) Social Informatics - 9th International Conference, SocInfo 2017, Oxford, UK, September 13-15, 2017, Proceedings, Part I, *Lecture Notes in Computer Science*, vol. 10539, pp. 75–87. Springer (2017). https://doi.org/10.1007/978-3-319-67217-5_6
81. van den Eijkhof, F., Bodlaender, H.L., Koster, A.M.C.A.: Safe reduction rules for weighted treewidth. Algorithmica **47**(2), 139–158 (2007). https://doi.org/10.1007/s00453-006-1226-x
82. English, A.: The early beginnings of social network analysis and the potential use for homeland security. https://goo.gl/1NtfpP (2016). (Accessed on 14/12/2017)
83. Ensor, A., Lillo, F.: Colored-edge graph approach for the modeling of multimodal transportation systems. APJOR **33**(1) (2016). https://doi.org/10.1142/S0217595916500056
84. Esper, A., Badr, Y., Biennier, F.: Hypergraph of services for business interconnectivity and collaboration. In: B. Vallespir, T. Alix (eds.) Advances in Production Management Systems. New Challenges, New Approaches - IFIP WG 5.7 International Conference, APMS 2009, Bordeaux, France, September 21-23, 2009, Revised Selected Papers, *IFIP Advances in Information and Communication Technology*, vol. 338, pp. 571–578. Springer (2009). https://doi.org/10.1007/978-3-642-16358-6_71
85. Estivill-Castro, V. (ed.): Computer Science 2005, Twenty-Eighth Australasian Computer Science Conference (ACSC2005), Newcastle, NSW, Australia, January/February 2005, *CRPIT*, vol. 38. Australian Computer Society (2005)
86. Fellows, M.: On the complexity of vertex set problems. Tech. rep., University of New Mexico (1988)
87. Fellows, M., Langston, M.: Nonconstructive advances in polynomial-time complexity. Information Processing Letters **28**, 157–162 (1987/88)
88. Fellows, M.R., Fertin, G., Hermelin, D., Vialette, S.: Upper and lower bounds for finding connected motifs in vertex-colored graphs. J. Comput. Syst. Sci. **77**(4), 799–811 (2011). https://doi.org/10.1016/j.jcss.2010.07.003
89. Fellows, M.R., Rosamond, F.A., Rotics, U., Szeider, S.: Clique-width is np-complete. SIAM J. Discrete Math. **23**(2), 909–939 (2009). https://doi.org/10.1137/070687256
90. Fomin, F.V., Thilikos, D.M.: Branchwidth of graphs. In: M.Y. Kao (ed.) Encyclopedia of Algorithms, pp. 232–237. Springer (2016). https://doi.org/10.1007/978-1-4939-2864-4_55
91. Fortunato, S.: Community detection in graphs. Physics Reports **486**(3), 75–174 (2010). https://doi.org/10.1016/j.physrep.2009.11.002
92. Fortunato, S., Hric, D.: Community detection in networks: A user guide. Physics Reports **659**(Supplement C), 1–44 (2016). https://doi.org/10.1016/j.physrep.2016.09.002. Community detection in networks: A user guide
93. Freeman, L.C.: The Development of Social Network Analysis: A Study in the Sociology of Science. Empirical Press, Vancouver, BC, Canada (2004)
94. Friggeri, A., Chelius, G., Fleury, E.: Triangles to capture social cohesion. In: PASSAT/SocialCom 2011, Privacy, Security, Risk and Trust (PASSAT), 2011 IEEE Third International Conference on and 2011 IEEE Third International Conference on Social Computing (SocialCom), Boston, MA, USA, 9-11 Oct., 2011, pp. 258–265. IEEE (2011). https://doi.org/10.1109/PASSAT/SocialCom.2011.169
95. Friggeri, A., Fleury, E.: Maximizing the cohesion is np-hard. CoRR **abs/1109.1994** (2011). http://arxiv.org/abs/1109.1994

96. Friggeri, A., Fleury, E.: Finding cohesive communities with C^3. Research Report RR-7947, INRIA (2012). https://hal.inria.fr/hal-00692548
97. Fuentes, J., Sáez, P., Gutierrez, G., Scherson, I.D.: A method to find functional dependencies through refutations and duality of hypergraphs. Comput. J. **58**(5), 1186–1198 (2015)
98. Gabardo, A.C., Berretta, R., de Vries, N.J., Moscato, P.: Where does my brand end? an overlapping community approach. In: G. Leu, H.K. Singh, S. Elsayed (eds.) Intelligent and Evolutionary Systems: The 20th Asia Pacific Symposium, IES 2016, Canberra, Australia, November 2016, Proceedings, pp. 133–148. Springer International Publishing, Cham (2017). https://doi.org/10.1007/978-3-319-49049-6_10
99. Gabardo, A.C., Lopes, H.S.: Using social network analysis to unveil cartels in public bids. In: Network Intelligence Conference (ENIC), 2014 European, pp. 17–21. IEEE Computer Society (2014). https://doi.org/10.1109/ENIC.2014.11
100. Galeana-Sánchez, H., Manrique, M.: Directed hypergraphs: A tool for researching digraphs and hypergraphs. Discussiones Mathematicae Graph Theory **29**(2), 313–335 (2009)
101. Garnovetter, M.S.: The strength of weak ties. American Journal of Sociology **78**, 1360–1380 (1973)
102. Gebremedhin, A.H.: The enabling power of graph coloring algorithms in automatic differentiation and parallel processing. In: U. Naumann, O. Schenk, H.D. Simon, S. Toledo (eds.) Combinatorial Scientific Computing, 01.02. - 06.02.2009, *Dagstuhl Seminar Proceedings*, vol. 09061. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, Germany (2009). http://drops.dagstuhl.de/opus/volltexte/2009/2093/
103. Ghamry, W.K., Shukry, S.: On the interplay of network structure and routing strategies on network design methods for mitigation of intentional attacks in scale-free networks. J. Network Syst. Manage. **25**(3), 508–535 (2017). https://doi.org/10.1007/s10922-016-9400-1
104. Gibson, H., Faith, J., Vickers, P.: A survey of two-dimensional graph layout techniques for information visualisation. Information Visualization **12**(3-4), 324–357 (2013). https://doi.org/10.1177/1473871612455749
105. Girvan, M., Newman, M.E.J.: Community structure in social and biological networks. PNAS **99**(12), 7821–7826 (2002)
106. Goldbeck, N., Angeloudis, P., Ochieng, W.Y.: A dynamic network flow model for interdependent infrastructure and supply chain networks with uncertain asset operability. In: T. Bektas, S. Coniglio, A. Martinez-Sykora, S. Voß (eds.) Computational Logistics - 8th International Conference, ICCL 2017, Southampton, UK, October 18-20, 2017, Proceedings, *Lecture Notes in Computer Science*, vol. 10572, pp. 513–528. Springer (2017). https://doi.org/10.1007/978-3-319-68496-3_34
107. Golovach, P.A., Johnson, M., Paulusma, D., Song, J.: A survey on the computational complexity of coloring graphs with forbidden subgraphs. Journal of Graph Theory **84**(4), 331–363 (2017). https://doi.org/10.1002/jgt.22028
108. Gómez, S., Fernández, A.: Radatools software, communities detection in complex networks and other tools (2011)
109. Gonçalves, J.N.C., Rodrigues, H.S., Monteiro, M.T.T.: A contribution of dynamical systems theory and epidemiological modeling to a viral marketing campaign. In: A.M. Madureira, A. Abraham, D. Gamboa, P. Novais (eds.) Intelligent Systems Design and Applications: 16th International Conference on Intelligent Systems Design and Applications (ISDA 2016) held in Porto, Portugal, December 16-18, 2016, pp. 974–983. Springer International Publishing, Cham (2017). https://doi.org/10.1007/978-3-319-53480-0_96
110. Gong, M., Cai, Q., Ma, L., Wang, S., Lei, Y.: Computational Intelligence for Network Structure Analytics. Springer (2017). https://doi.org/10.1007/978-981-10-4558-5
111. Gong, M., Fu, B., Jiao, L., Du, H.: Memetic algorithm for community detection in networks. Phys. Rev. E **84**, 056,101 (2011). https://doi.org/10.1103/PhysRevE.84.056101
112. González, A.D., Dueñas-Osorio, L., Sánchez-Silva, M., Medaglia, A.L.: The interdependent network design problem for optimal infrastructure system restoration. Comp.-Aided Civil and Infrastruct. Engineering **31**(5), 334–350 (2016). https://doi.org/10.1111/mice.12171

113. Gonzalez, T.F.: Handbook of Approximation Algorithms and Metaheuristics (Chapman & Hall/Crc Computer & Information Science Series). Chapman & Hall/CRC (2007)
114. Grohe, M.: Tangles and connectivity in graphs. In: A. Dediu, J. Janousek, C. Martín-Vide, B. Truthe (eds.) Language and Automata Theory and Applications - 10th International Conference, LATA 2016, Prague, Czech Republic, March 14-18, 2016, Proceedings, *Lecture Notes in Computer Science*, vol. 9618, pp. 24–41. Springer (2016). https://doi.org/10.1007/978-3-319-30000-9_2
115. Grohe, M., Marx, D.: On tree width, bramble size, and expansion. J. Comb. Theory, Ser. B **99**(1), 218–228 (2009). https://doi.org/10.1016/j.jctb.2008.06.004
116. Guisado-Gámez, J., Prat-Pérez, A., Larriba-Pey, J.: Query expansion via structural motifs in wikipedia graph. CoRR **abs/1602.07217** (2016). http://arxiv.org/abs/1602.07217
117. Gurciullo, S.: Organised crime infiltration in the legitimate private economy - an empirical network analysis approach. CoRR **abs/1403.5071** (2014). http://arxiv.org/abs/1403.5071
118. Hamza, H.S., Deogun, J.S.: WDM multistage interconnection networks architectures for enhancing supernetworks switching infrastructure. In: D.A. Bader, M. Parashar, S. Varadarajan, V.K. Prasanna (eds.) High Performance Computing - HiPC 2005, 12th International Conference, Goa, India, December 18-21, 2005, Proceedings, *Lecture Notes in Computer Science*, vol. 3769, pp. 444–453. Springer (2005). https://doi.org/10.1007/11602569_46
119. Hansberg, A., Meierling, D., Volkmann, L.: Distance domination and distance irredundance in graphs. Electr. J. Comb. **14**(1) (2007). http://www.combinatorics.org/Volume_14/Abstracts/v14i1r35.html
120. Haque, M.N., Mathieson, L., Moscato, P.: A Memetic Algorithm for community detection by maximising the Connected Cohesion. In: 2017 IEEE Symposium Series on Computational Intelligence (SSCI), Huawii, USA, 27 Nov-1 Dec, pp. 2475–2482. IEEE (2017). https://doi.org/10.1145/3067695.3076106
121. Haque, M.N., Noman, N., Berretta, R., Moscato, P.: Heterogeneous ensemble combination search using genetic algorithm for class imbalanced data classification. PLOS ONE **11**(1), 1–28 (2016). https://doi.org/10.1371/journal.pone.0146116
122. Harvey, D.J., Wood, D.R.: Parameters tied to treewidth. Journal of Graph Theory **84**(4), 364–385 (2017). https://doi.org/10.1002/jgt.22030
123. Haslett, G., Bullock, S., Brede, M.: Planar growth generates scale-free networks. J. Complex Networks **4**(4), 500–516 (2016). https://doi.org/10.1093/comnet/cnw005
124. Hassin, R., Meggido, N.: Approximation algorithms for hitting objects with straight lines. Discrete Applied Mathematics **30**, 29–42 (1991)
125. Hatano, D., Fukunaga, T., Maehara, T., Kawarabayashi, K.: Lagrangian decomposition algorithm for allocating marketing channels. In: AAAI, pp. 1144–1150. AAAI Press (2015)
126. Havlin, S., Kenett, D.Y., Ben-Jacob, E., Bunde, A., Cohen, R., Hermann, H., Kantelhardt, J., Kertész, J., Kirkpatrick, S., Kurths, J., et al.: Challenges in network science: Applications to infrastructures, climate, social systems and economics. The European Physical Journal Special Topics **214**, 273–293 (2012)
127. Heine, C., Leitte, H., Hlawitschka, M., Iuricich, F., Floriani, L.D., Scheuermann, G., Hagen, H., Garth, C.: A survey of topology-based methods in visualization. Comput. Graph. Forum **35**(3), 643–667 (2016). https://doi.org/10.1111/cgf.12933
128. Helm, S.: Viral marketing - establishing customer relationships by 'word-of-mouse'. Electronic Markets **10**(3), 158–161 (2000)
129. Hicks, I.V.: Graphs, branchwidth, and tangles! oh my! Networks **45**(2), 55–60 (2005). https://doi.org/10.1002/net.20050
130. Hildebrand, C., Hofstetter, R., Herrmann, A.: Modeling viral marketing dynamics in social networks - findings from computational experiments with agent-based simulation models. In: ICIS. Association for Information Systems (2012)
131. Holme, P.: Temporal networks. In: Encyclopedia of Social Network Analysis and Mining, pp. 2119–2129. Springer (2014). https://doi.org/10.1007/978-1-4614-6170-8_42
132. Holme, P., Saramäki, J.: Temporal networks. Physics Reports **519**(3), 97–125 (2012). https://doi.org/10.1016/j.physrep.2012.03.001. Temporal Networks

133. Hou, R., Wu, J., Chen, Y., Zhang, H.: Note on edge-colored graphs for networks with homogeneous faults. Comput. J. **59**(10), 1470–1478 (2016). https://doi.org/10.1093/comjnl/bxw012
134. Hou, R., Wu, J., Chen, Y., Zhang, H., Sui, X.: Constructing edge-colored graph for heterogeneous networks. J. Comput. Sci. Technol. **30**(5), 1154–1160 (2015). https://doi.org/10.1007/s11390-015-1551-0
135. Howlader, J., Mal, A.K.: Sealed-bid auction: a cryptographic solution to bid-rigging attack in the collusive environment. Security and Communication Networks **8**(18), 3415–3440 (2015). https://doi.org/10.1002/sec.1268
136. Hu, D., Yan, J., Algesheimer, R., Meierer, M.: Understanding moderators of peer influence for engineering viral marketing seeding simulations and strategies. In: ICIS. Association for Information Systems (2016)
137. Hu, J., Fang, D., Wei, X., Xie, J.: Colluder detection based on hypergraph decomposition. In: Ninth International Conference on Computational Intelligence and Security, CIS 2013, Emei Mountain, Sichan Province, China, December 14-15, 2013, pp. 630–634. IEEE Computer Society (2013). https://doi.org/10.1109/CIS.2013.138
138. Huang, X., Lakshmanan, L.V.S., Xu, J.: Community search over big graphs: Models, algorithms, and opportunities. In: 33rd IEEE International Conference on Data Engineering, ICDE 2017, San Diego, CA, USA, April 19-22, 2017, pp. 1451–1454. IEEE Computer Society (2017). https://doi.org/10.1109/ICDE.2017.211
139. Huang, Y., Dai, H.: On information spreading in multiplex networks with gossip mechanism. In: IEEE International Conference on Communications, ICC 2017, Paris, France, May 21-25, 2017, pp. 1–6. IEEE (2017). https://doi.org/10.1109/ICC.2017.7997367
140. Iacovacci, J., Lacasa, L.: Visibility graph motifs. CoRR **abs/1512.00297** (2015). http://arxiv.org/abs/1512.00297
141. Inafuku, K., Fushimi, T., Satoh, T.: Extraction method of typical purchase patterns based on motif analysis of directed graphs. In: Proceedings of the 18th International Conference on Information Integration and Web-based Applications and Services, iiWAS '16, pp. 86–95. ACM, New York, NY, USA (2016). https://doi.org/10.1145/3011141.3011178
142. Inostroza-Ponta, M., Berretta, R., Mendes, A., Moscato, P.: An automatic graph layout procedure to visualize correlated data. In: M. Bramer (ed.) Artificial Intelligence in Theory and Practice: IFIP 19th World Computer Congress, TC 12: IFIP AI 2006 Stream, August 21–24, 2006, Santiago, Chile, pp. 179–188. Springer US, Boston, MA (2006). https://doi.org/10.1007/978-0-387-34747-9_19
143. Inostroza-Ponta, M., Berretta, R., Moscato, P.: Qapgrid: A two level qap-based approach for large-scale data analysis and visualization. PLOS ONE **6**(1), 1–18 (2011). https://doi.org/10.1371/journal.pone.0014468
144. Jansen, B.M.: The power of data reduction: Kernels for fundamental graph problems. Ph.D. thesis, Utrecht University (2013)
145. Jerrum, M., Meeks, K.: The parameterised complexity of counting connected subgraphs and graph motifs. J. Comput. Syst. Sci. **81**(4), 702–716 (2015). https://doi.org/10.1016/j.jcss.2014.11.015
146. Jiang, J.J., Wen, S., Yu, S., Xiang, Y., Zhou, W., Hassan, H.: The structure of communities in scale-free networks. Concurrency and Computation: Practice and Experience **29**(14) (2017). https://doi.org/10.1002/cpe.4040
147. Jimenez, F., Sanhueza, C., Berretta, R., Moscato, P.: A multi-objective approach for the (α, β)-k-feature set problem using memetic algorithms. In: P.A.N. Bosman (ed.) Genetic and Evolutionary Computation Conference, Berlin, Germany, July 15-19, 2017, Companion Material Proceedings, pp. 207–208. ACM (2017). https://doi.org/10.1145/3067695.3076106
148. Johnson, D.: The NP-completeness column: An ongoing guide: The many faces of polynomial time. Journal of Algorithms **8**(2), 285–303 (1987)

149. Johnstone, D., Milward, E.A., Berretta, R., Moscato, P., for the Alzheimer's Disease Neuroimaging Initiative: Multivariate protein signatures of pre-clinical alzheimer's disease in the alzheimer's disease neuroimaging initiative (adni) plasma proteome dataset. PLOS ONE **7**(4), 1–17 (2012)
150. Kabir, H., Madduri, K.: Shared-memory graph truss decomposition. CoRR **abs/1707.02000** (2017). http://arxiv.org/abs/1707.02000
151. Kaluza, P., Kölzsch, A., Gastner, M.T., Blasius, B.: The complex network of global cargo ship movements. Journal of the Royal Society Interface **7**(48), 1093–1103 (2010)
152. Kang, C., Kraus, S., Molinaro, C., Spezzano, F., Subrahmanian, V.: Diffusion centrality: A paradigm to maximize spread in social networks. Artificial Intelligence **239**, 70–96 (2016). https://doi.org/10.1016/j.artint.2016.06.008
153. Kao, T., Porter, M.A.: Layer communities in multiplex networks. CoRR **abs/1706.04147** (2017). http://arxiv.org/abs/1706.04147
154. Kaple, M., Kulkarni, K., Potika, K.: Viral marketing for smart cities: Influencers in social network communities. In: Third IEEE International Conference on Big Data Computing Service and Applications, BigDataService 2017, Redwood City, CA, USA, April 6-9, 2017, pp. 106–111. IEEE (2017). https://doi.org/10.1109/BigDataService.2017.46
155. Karami, E., Glisic, S.: Joint optimization of scheduling and routing in multicast wireless ad hoc networks using soft graph coloring and nonlinear cubic games. IEEE Trans. Vehicular Technology **60**(7), 3350–3360 (2011). https://doi.org/10.1109/TVT.2011.2161355
156. Karp, R.M.: Reducibility among combinatorial problems. In: R.E. Miller, J.W. Thatcher, J.D. Bohlinger (eds.) Complexity of Computer Computations: Proceedings of a symposium on the Complexity of Computer Computations, held March 20–22, 1972, at the IBM Thomas J. Watson Research Center, Yorktown Heights, New York, and sponsored by the Office of Naval Research, Mathematics Program, IBM World Trade Corporation, and the IBM Research Mathematical Sciences Department, pp. 85–103. Springer US, Boston, MA (1972). https://doi.org/10.1007/978-1-4684-2001-2_9
157. Kaveh-Yazdy, F., Kong, X., Li, J., Li, F., Xia, F.: Customer rating prediction using hypergraph kernel based classification. In: AMT, *Lecture Notes in Computer Science*, vol. 8210, pp. 187–192. Springer (2013)
158. Kawase, Y., Matsui, T., Miyauchi, A.: Additive approximation algorithms for modularity maximization. In: S. Hong (ed.) 27th International Symposium on Algorithms and Computation, ISAAC 2016, December 12-14, 2016, Sydney, Australia, *LIPIcs*, vol. 64, pp. 43:1–43:13. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik (2016). https://doi.org/10.4230/LIPIcs.ISAAC.2016.43
159. Khan, A., Zehnder, B., Kossmann, D.: Revenue maximization by viral marketing: A social network host's perspective. In: ICDE, pp. 37–48. IEEE Computer Society (2016). https://doi.org/10.1109/ICDE.2016.7498227
160. Khan, B.S., Niazi, M.A.: Network community detection: A review and visual survey. CoRR **abs/1708.00977** (2017). http://arxiv.org/abs/1708.00977
161. Khor, S.: Application of graph coloring to biological networks. CoRR **abs/0912.3461** (2009). http://arxiv.org/abs/0912.3461
162. Kivelä, M., Arenas, A., Barthelemy, M., Gleeson, J.P., Moreno, Y., Porter, M.A.: Multilayer networks. Journal of Complex Networks **2**(3), 203–271 (2014)
163. Krot, A.M., Prokhorenkova, L.O.: Local clustering coefficient in generalized preferential attachment models. In: D.F. Gleich, J. Komjáthy, N. Litvak (eds.) Algorithms and Models for the Web Graph - 12th International Workshop, WAW 2015, Eindhoven, The Netherlands, December 10-11, 2015, Proceedings, *Lecture Notes in Computer Science*, vol. 9479, pp. 15–28. Springer (2015). https://doi.org/10.1007/978-3-319-26784-5_2
164. Krot, A.M., Prokhorenkova, L.O.: Assortativity in generalized preferential attachment models. In: A. Bonato, F.C. Graham, P. Pralat (eds.) Algorithms and Models for the Web Graph - 13th International Workshop, WAW 2016, Montreal, QC, Canada, December 14-15, 2016, Proceedings, *Lecture Notes in Computer Science*, vol. 10088, pp. 9–21 (2016). https://doi.org/10.1007/978-3-319-49787-7_2

165. Kruskal, J.: On the shortest spanning subtree of a graph and the travelling salesman problem. Proc. Amer. Math. Soc. **7**, 48–50 (1956)
166. Lamm, S., Sanders, P., Schulz, C., Strash, D., Werneck, R.F.: Finding near-optimal independent sets at scale. Journal of Heuristics **23**(4), 207–229 (2017). https://doi.org/10.1007/s10732-017-9337-x
167. Lancia, G., Mathieson, L., Moscato, P.: Separating sets of strings by finding matching patterns is almost always hard. Theor. Comput. Sci. **665**, 73–86 (2017). https://doi.org/10.1016/j.tcs.2016.12.018
168. Laura, L., Nanni, U., Temperini, M.: The organization of large-scale repositories of learning objects with directed hypergraphs. In: ICWL Workshops, *Lecture Notes in Computer Science*, vol. 8699, pp. 23–33. Springer (2014)
169. Lee, W., Kim, J., Yu, H.: CT-IC: continuously activated and time-restricted independent cascade model for viral marketing. In: ICDM, pp. 960–965. IEEE Computer Society (2012)
170. Leskovec, J., Adamic, L.A., Huberman, B.A.: The dynamics of viral marketing. ACM Trans. Web **1**(1) (2007). https://doi.org/10.1145/1232722.1232727
171. Leskovec, J., Sosič, R.: Snap: Stanford network analysis platform (2013)
172. Li, D., Xu, Z., Li, S., Sun, X.: Link prediction in social networks based on hypergraph. In: L. Carr, A.H.F. Laender, B.F. Lóscio, I. King, M. Fontoura, D. Vrandecic, L. Aroyo, J.P.M. de Oliveira, F. Lima, E. Wilde (eds.) 22nd International World Wide Web Conference, WWW '13, Rio de Janeiro, Brazil, May 13-17, 2013, Companion Volume, pp. 41–42. International World Wide Web Conferences Steering Committee / ACM (2013). http://dl.acm.org/citation.cfm?id=2487802
173. Li, L., Li, T.: News recommendation via hypergraph learning: encapsulation of user behavior and news content. In: S. Leonardi, A. Panconesi, P. Ferragina, A. Gionis (eds.) Sixth ACM International Conference on Web Search and Data Mining, WSDM 2013, Rome, Italy, February 4-8, 2013, pp. 305–314. ACM (2013). https://doi.org/10.1145/2433396.2433436
174. Li, M., Zhao, C., Tang, J.: Hybrid image summarization by hypergraph partition. Neurocomputing **119**, 41–48 (2013)
175. Li, X., Huang, L.: Evaluation of software architectures reliability based on hypergraph grammar. In: 37th Annual IEEE Computer Software and Applications Conference, COMPSAC 2013, Kyoto, Japan, July 22-26, 2013, pp. 601–602. IEEE Computer Society (2013). https://doi.org/10.1109/COMPSAC.2013.95
176. Li, X., Smith, J.D., Dinh, T.N., Thai, M.T.: Why approximate when you can get the exact? optimal targeted viral marketing at scale. CoRR **abs/1701.08462** (2017)
177. Li, Y., Guo, C.: Hypergraph-based spectral clustering for categorical data. In: Seventh International Conference on Advanced Computational Intelligence, ICACI 2015, Wuyi, China, March 27-29, 2015, pp. 396–401. IEEE (2015). https://doi.org/10.1109/ICACI.2015.7184738
178. Li, Y., Li, Y., Zhang, B., Shen, H.: Preserving private cloud service data based on hypergraph anonymization. In: 2013 International Conference on Parallel and Distributed Computing, Applications and Technologies, pp. 192–197 (2013). https://doi.org/10.1109/PDCAT.2013.37
179. Li, Y., Shen, H.: On identity disclosure control for hypergraph-based data publishing. IEEE Trans. Information Forensics and Security **8**(8), 1384–1396 (2013)
180. Liao, F.: Modeling duration choice in space-time multi-state supernetworks for individual activity-travel scheduling. Transportation Research Part C: Emerging Technologies **69**, 16–35 (2016). https://doi.org/10.1016/j.trc.2016.05.011
181. Liao, F., Arentze, T., Timmermans, H.: Multi-state supernetworks: recent progress and prospects. Journal of Traffic and Transportation Engineering (English Edition) **1**(1), 13–27 (2014). https://doi.org/10.1016/S2095-7564(15)30085-4
182. Liao, F., Arentze, T.A., Timmermans, H.J.P.: Constructing personalized transportation networks in multi-state supernetworks: a heuristic approach. International Journal of Geographical Information Science **25**(11), 1885–1903 (2011). https://doi.org/10.1080/13658816.2011.556119

183. Liao, F., Rasouli, S., Timmermans, H.J.P.: Incorporating activity-travel time uncertainty and stochastic space-time prisms in multistate supernetworks for activity-travel scheduling. International Journal of Geographical Information Science **28**(5), 928–945 (2014). https://doi.org/10.1080/13658816.2014.887086
184. Lim, E.P., Chen, H., Chen, G.: Business intelligence and analytics: Research directions. ACM Transactions on Management Information Systems (TMIS) **3**(4), 17 (2013)
185. Lipton, Z.C.: Does deep learning come from the devil? https://goo.gl/p9KyZG (2015). (Accessed on 14/12/2017)
186. Liu, F., Yang, L.: A novel cell detection method using deep convolutional neural network and maximum-weight independent set. In: L. Lu, Y. Zheng, G. Carneiro, L. Yang (eds.) Deep Learning and Convolutional Neural Networks for Medical Image Computing - Precision Medicine, High Performance and Large-Scale Datasets, Advances in Computer Vision and Pattern Recognition, pp. 63–72. Springer (2017). https://doi.org/10.1007/978-3-319-42999-1_5
187. Liu, G., Wong, L.: Effective pruning techniques for mining quasi-cliques. In: ECML/PKDD (2), *Lecture Notes in Computer Science*, vol. 5212, pp. 33–49. Springer (2008)
188. Liu, P., Liao, F., Huang, H.J., Timmermans, H.: Dynamic activity-travel assignment in multi-state supernetworks. Transportation Research Procedia **7**, 24–43 (2015). https://doi.org/10.1016/j.trpro.2015.06.002. 21st International Symposium on Transportation and Traffic Theory Kobe, Japan, 5-7 August, 2015
189. Liu, P., Liao, F., Huang, H.J., Timmermans, H.: Dynamic activity-travel assignment in multi-state supernetworks under transport and location capacity constraints. Transportmetrica A: Transport Science **12**(7), 572–590 (2016). https://doi.org/10.1080/23249935.2016.1189739
190. Liu, Q., Fang, J.Q., Li, Y.: Some characteristics of supernetworks based on unified hybrid network theory framework. International Journal of Modern Physics C **28**(05), 1750,057 (2017). https://doi.org/10.1142/S0129183117500577
191. Liu, Y., Shao, J., Xiao, J., Wu, F., Zhuang, Y.: Hypergraph spectral hashing for image retrieval with heterogeneous social contexts. Neurocomputing **119**, 49–58 (2013)
192. Liu, Z., Nagurney, A.: Financial networks with intermediation and transportation network equilibria: A supernetwork equivalence and reinterpretation of the equilibrium conditions with computations. Comput. Manag. Science **4**(3), 243–281 (2007). https://doi.org/10.1007/s10287-006-0010-3
193. Long, C., Wong, R.C.: Minimizing seed set for viral marketing. In: 2011 IEEE 11th International Conference on Data Mining, pp. 427–436 (2011). https://doi.org/10.1109/ICDM.2011.99
194. Lü, Z., Hao, J.K.: A memetic algorithm for graph coloring. European Journal of Operational Research **203**(1), 241–250 (2010). https://doi.org/10.1016/j.ejor.2009.07.016
195. Lu, Z., Zhou, H., Li, V.O.K., Long, Y.: Pricing game of celebrities in sponsored viral marketing in online social networks with a greedy advertising platform. In: 2016 IEEE International Conference on Communications (ICC), pp. 1–6. IEEE (2016). https://doi.org/10.1109/ICC.2016.7511396
196. Lusseau, D., Schneider, K., Boisseau, O.J., Haase, P., Slooten, E., Dawson, S.M.: The bottlenose dolphin community of Doubtful Sound features a large proportion of long-lasting associations. Behavioral Ecology and Sociobiology **54**, 396–405 (2003)
197. Ma'ayan, A., Cecchi, G.A., Wagner, J., Rao, A.R., Iyengar, R., Stolovitzky, G.: Ordered cyclic motifs contribute to dynamic stability in biological and engineered networks. Proceedings of the National Academy of Sciences **105**(49), 19,235–19,240 (2008). https://doi.org/10.1073/pnas.0805344105
198. Malliaros, F.D., Vazirgiannis, M.: Clustering and community detection in directed networks: A survey. Physics Reports **533**(4), 95–142 (2013). https://doi.org/10.1016/j.physrep.2013.08.002. Clustering and Community Detection in Directed Networks: A Survey
199. Marshak, C.Z.: Applications of network science to criminal networks, university education, and ecology. Ph.D. thesis, University of California, Los Angeles, USA (2017). http://www.escholarship.org/uc/item/6zs7394q

200. Mathieson, L., Mendes, A., Marsden, J., Pond, J., Moscato, P.: Computer-aided breast cancer diagnosis with optimal feature sets: Reduction rules and optimization techniques. In: J.M. Keith (ed.) Bioinformatics: Volume II: Structure, Function, and Applications, pp. 299–325. Springer New York, New York, NY (2017). https://doi.org/10.1007/978-1-4939-6613-4_17
201. McGuire, G., Tugemann, B., Civario, G.: There is no 16-clue sudoku: Solving the sudoku minimum number of clues problem via hitting set enumeration. Experimental Mathematics **23**(2), 190–217 (2014). https://doi.org/10.1080/10586458.2013.870056
202. McNabb, L., Laramee, R.S.: Survey of surveys (sos) - mapping the landscape of survey papers in information visualization. Comput. Graph. Forum **36**(3), 589–617 (2017). https://doi.org/10.1111/cgf.13212
203. Meghanathan, N.: Correlation analysis between maximal clique size and centrality metrics for random networks and scale-free networks. Computer and Information Science **9**(2), 41–57 (2016). https://doi.org/10.5539/cis.v9n2p41
204. Mellor, D., Prieto, E., Mathieson, L., Moscato, P.: A kernelisation approach for multiple d-hitting set and its application in optimal multi-drug therapeutic combinations. PLOS ONE **5**(10), 1–13 (2010). https://doi.org/10.1371/journal.pone.0013055
205. Meyer, H., Schering, A., Heuer, A.: The hydra.powergraph system - building digital archives with directed and typed hypergraphs. Datenbank-Spektrum **17**(2), 113–129 (2017). https://doi.org/10.1007/s13222-017-0253-x
206. Miao, H., Liu, X., Huang, B., Getoor, L.: A hypergraph-partitioned vertex programming approach for large-scale consensus optimization. In: BigData Conference, pp. 563–568. IEEE (2013)
207. Miao, Z., Balasundaram, B.: Approaches for finding cohesive subgroups in large-scale social networks via maximum *k*-plex detection. Networks **69**(4), 388–407 (2017)
208. Michael, R.G., David, S.J.: Computers and intractability: a guide to the theory of np-completeness. WH Free. Co., San Fr pp. 90–91 (1979)
209. Michalak, K., Korczak, J.J.: Graph mining approach to suspicious transaction detection. In: M. Ganzha, L.A. Maciaszek, M. Paprzycki (eds.) Federated Conference on Computer Science and Information Systems - FedCSIS 2011, Szczecin, Poland, 18-21 September 2011, Proceedings, pp. 69–75 (2011). http://ieeexplore.ieee.org/document/6078254/
210. Mohite, M., Narahari, Y.: Incentive compatible influence maximization in social networks and application to viral marketing. CoRR **abs/1102.0918** (2011)
211. Mohite, M., Narahari, Y.: Incentive compatible influence maximization in social networks and application to viral marketing. In: AAMAS, pp. 1081–1082. IFAAMAS (2011)
212. Mokken, R.J.: Cliques, clubs and clans. Quality and Quantity **13**, 161–173 (1979)
213. Molnár, B.: Applications of Hypergraphs in Informatics: A Survey and Opportunities for Research, vol. 42, pp. 261–282 (2014)
214. Moreno, J.: Who Shall Survive? A New Approach to the Problem of Human Interrelations. No. 58 in Nervous and Mental Disease Publishing. Nervous and Mental Disease Publishing Co., Washington D.C. (1934)
215. Morselli, C., Boivin, R.: Introduction to the special issue on crime and networks. Social Networks **51**, 1–2 (2017). https://doi.org/10.1016/j.socnet.2017.08.005
216. Moscato, P., Berretta, R., Hourani, M., Mendes, A., Cotta, C.: Genes related with alzheimer's disease: A comparison of evolutionary search, statistical and integer programming approaches. In: F. Rothlauf, J. Branke, S. Cagnoni, D.W. Corne, R. Drechsler, Y. Jin, P. Machado, E. Marchiori, J. Romero, G.D. Smith, G. Squillero (eds.) Applications of Evolutionary Computing, EvoWorkshops 2005: EvoBIO, EvoCOMNET, EvoHOT, EvoIASP, EvoMUSART, and EvoSTOC, Lausanne, Switzerland, March 30 - April 1, 2005, Proceedings, *Lecture Notes in Computer Science*, vol. 3449, pp. 84–94. Springer (2005). https://doi.org/10.1007/978-3-540-32003-6_9
217. Moscato, P., Mathieson, L., Mendes, A., Berretta, R.: The electronic primaries: Predicting the U.S. presidency using feature selection with safe data reduction. In: Estivill-Castro [85], pp. 371–380. http://crpit.com/confpapers/CRPITV38Moscato.pdf

218. Murthy, V.K., Krishnamurthy, E.V.: Learning to capture the functions of genetic regulatory networks using graph motifs. IJAIP **4**(2), 185–197 (2012). https://doi.org/10.1504/IJAIP.2012.048146
219. Mylonakis, N., Orejas, F., Fiadeiro, J.L.: A semantics of business configurations using symbolic graphs. In: SCC, pp. 146–153. IEEE Computer Society (2015)
220. Naeni, L.M., Berretta, R., Moscato, P.: Ma-net: A reliable memetic algorithm for community detection by modularity optimization. In: H. Handa, H. Ishibuchi, Y.S. Ong, K.C. Tan (eds.) Proceedings of the 18th Asia Pacific Symposium on Intelligent and Evolutionary Systems, Volume 1, pp. 311–323. Springer International Publishing, Cham (2015). https://doi.org/10.1007/978-3-319-13359-1_25
221. Naeni, L.M., de Vries, N.J., Reis, R., Arefin, A.S., Berretta, R., Moscato, P.: Identifying communities of trust and confidence in the charity and not-for-profit sector: A memetic algorithm approach. In: 2014 IEEE Fourth International Conference on Big Data and Cloud Computing, BDCloud 2014, Sydney, Australia, December 3-5, 2014, pp. 500–507. IEEE Computer Society (2014). https://doi.org/10.1109/BDCloud.2014.83
222. Nagurney, A.: Supernetworks. In: M.G.C. Resende, P.M. Pardalos (eds.) Handbook of Optimization in Telecommunications, pp. 1073–1119. Springer (2006). https://doi.org/10.1007/978-0-387-30165-5_37
223. Nagurney, A., Cruz, J., Matsypura, D.: Dynamics of global supply chain supernetworks. Mathematical and Computer Modelling **37**(9), 963–983 (2003). https://doi.org/10.1016/S0895-7177(03)00112-2.
224. Nagurney, A., Cruz, J., Matsypura, D.: Dynamics of global supply chain supernetworks. Mathematical and Computer Modelling **37**(9), 963–983 (2003). https://doi.org/10.1016/S0895-7177(03)00112-2
225. Nagurney, A., Dong, J.: Supernetworks: Decision-Making for the Information Age. Elgar, Edward Publishing, Incorporated (2002)
226. Nagurney, A., Dong, J.: Management of knowledge intensive systems as supernetworks: Modeling, analysis, computations, and applications. Mathematical and Computer Modelling **42**(3-4), 397–417 (2005). https://doi.org/10.1016/j.mcm.2004.01.015
227. Naik, S.A., Yu, Q.: Maximizing influence of viral marketing via evolutionary user selection. In: ASONAM, pp. 1435–1436. ACM (2013)
228. Natschläger, C., Geist, V., Illibauer, C., Hutter, R.: Modelling business process variants using graph transformation rules. In: S. Hammoudi, L.F. Pires, B. Selic, P. Desfray (eds.) MODELSWARD 2016 - Proceedings of the 4rd International Conference on Model-Driven Engineering and Software Development, Rome, Italy, 19-21 February, 2016., pp. 65–74. SciTePress (2016). https://doi.org/10.5220/0005665800650074
229. Nešetřil, J., Milková, E., Nešetřilová, H.: Otakar borůvka on minimum spanning tree problem translation of both the 1926 papers, comments, history. Discrete Mathematics **233**(1), 3–36 (2001). https://doi.org/10.1016/S0012-365X(00)00224-7. Czech and Slovak 2
230. Newman, M.E.: Fast algorithm for detecting community structure in networks. Physical review E **69**(6), 066,133 (2004)
231. Newman, M.E.: Modularity and community structure in networks. Proc Natl Acad Sci U S A **103**(23), 8577–8582 (2006). https://doi.org/10.1073/pnas.0601602103. http://www.ncbi.nlm.nih.gov/sites/entrez?cmd=retrieve&db=pubmed&list_uids=16723398&dopt=AbstractPlus
232. Newman, M.E.: Modularity and community structure in networks. Proceedings of the national academy of sciences **103**(23), 8577–8582 (2006)
233. Nešetřil, J., Nešetřilová, H.: The origins of minimal spanning tree algorithms - Borůvka and Jarník. Documenta Mathematica - Extra Volume ISMP pp. 127– 141 (2012)
234. Nguyen, H.T., Dinh, T.N., Thai, M.T.: Cost-aware targeted viral marketing in billion-scale networks. In: INFOCOM, pp. 1–9. IEEE (2016). https://doi.org/10.1109/INFOCOM.2016.7524377
235. Niedermeier, R.: Some prospects for efficient fixed parameter algorithms. In: B. Rovan (ed.) Theory and Practice of Informatics, Seminar on Current Trends in Theory and Practice of Informatics, vol. 1521, pp. 168–185. Springer-Verlag (1998)

236. Niedermeier, R.: Ubiquitous parameterization - invitation to fixed-parameter algorithms. In: Mathematical Foundations of Computer Science 2004, 29th International Symposium, MFCS 2004, Prague, Czech Republic, August 22-27, 2004, Proceedings, pp. 84–103 (2004). https://doi.org/10.1007/978-3-540-28629-5_4
237. Niedermeier, R., Rossmanith, P.: An efficient fixed parameter algorithm for 3-hitting set. Tech. Rep. WSI-99-18, Universität Tübingen, Wilhelm-Schickard-Institut für Informatik (1999). Technical Report, Revised version accepted in *Journal of Discrete Algorithms*, August 2000
238. Niedermeier, R., Rossmanith, P.: Upper bounds for vertex cover futher improved. In: STACS '99, vol. 1563, pp. 561–570. Springer-Verlag (1999)
239. Niedermeier, R., Rossmanith, P.: A general method to speed up fixed-parameter-tractable algorithms. Information Processing Letters **73**, 125–129 (2000)
240. Omodei, E., de Domenico, M., Arenas, A.: Evaluating the impact of interdisciplinary research: A multilayer network approach. Network Science **5**(2), 235?246 (2017). https://doi.org/10.1017/nws.2016.15
241. Opuszko, M., Ruhland, J.: Effects of the network structure on the dynamics of viral marketing. In: Wirtschaftsinformatik, p. 94 (2013)
242. Ostroumova Prokhorenkova, L.: General results on preferential attachment and clustering coefficient. Optimization Letters **11**(2), 279–298 (2017). https://doi.org/10.1007/s11590-016-1030-8
243. Pajouh, F.M., Miao, Z., Balasundaram, B.: A branch-and-bound approach for maximum quasi-cliques. Annals OR **216**(1), 145–161 (2014)
244. Paoletti, T.: Leonard euler's solution to the konigsberg bridge problem | mathematical association of america. https://goo.gl/Xeo1uL (2011). (Accessed on 14/12/2017)
245. Papadimitriou, C., Yannakakis, M.: On limited nondeterminism and the complexity of the V-C dimension. In: J. Allender, Eric; Beigel, Richard; Cai, Jin-yi; Feigenbaum, Joan; Homer, Steven; Lange, Klaus-Jörn; Lutz, Jack; Mahaney, Stephen; Straubing, Howard; Torán (ed.) Proceedings of the 8th Annual Conference on Structure in Complexity Theory (SCTC '93), pp. 12–18. IEEE Computer Society Press, San Diego, CA, USA (1993)
246. Papadimitriou, C., Yannakakis, M.: On limited nondeterminism and the complexity of the V-C dimension. Journal of Computer and System Sciences **53**(2), 161–170 (1996)
247. Pardalos, P.M., Rebennack, S.: Computational challenges with cliques, quasi-cliques and clique partitions in graphs. In: SEA, *Lecture Notes in Computer Science*, vol. 6049, pp. 13–22. Springer (2010)
248. Patrignani, M.: Planarity testing and embedding. In: Handbook of Graph Drawing and Visualization, pp. 1–42. Chapman and Hall/CRC (2013)
249. Pattillo, J., Veremyev, A., Butenko, S., Boginski, V.: On the maximum quasi-clique problem. Discrete Applied Mathematics **161**(1), 244–257 (2013). https://doi.org/10.1016/j.dam.2012.07.019
250. Pattillo, J., Veremyev, A., Butenko, S., Boginski, V.: On the maximum quasi-clique problem. Discrete Applied Mathematics **161**(1-2), 244–257 (2013). https://doi.org/10.1016/j.dam.2012.07.019
251. Pattillo, J., Youssef, N., Butenko, S.: On clique relaxation models in network analysis. European Journal of Operational Research **226**(1), 9–18 (2013). https://doi.org/10.1016/j.ejor.2012.10.021
252. de Paula, M.R., Berretta, R., Moscato, P.: A fast meta-heuristic approach for the (α, β)-k-feature set problem. J. Heuristics **22**(2), 199–220 (2016). https://doi.org/10.1007/s10732-015-9307-0
253. Peixoto, T.P.: The graph-tool python library. figshare (2014)
254. Penny, R., Bowles, R., Bouhana, N.: Social network modelling for counter extremism - comparing criminality in two activist networks. In: SIMULTECH 2013 - Proceedings of the 3rd International Conference on Simulation and Modeling Methodologies, Technologies and Applications, Reykjavík, Iceland, 29-31 July, 2013, pp. 382–388 (2013). https://doi.org/10.5220/0004586403820388

255. Pham, C.V., Thai, M.T., Ha, D., Ngo, D.Q., Hoang, H.X.: Time-critical viral marketing strategy with the competition on online social networks. In: H.T. Nguyen, V. Snasel (eds.) Computational Social Networks: 5th International Conference, CSoNet 2016, Ho Chi Minh City, Vietnam, August 2-4, 2016, Proceedings, pp. 111–122. Springer International Publishing, Cham (2016). https://doi.org/10.1007/978-3-319-42345-6_10
256. Polyvyanyy, A., Weske, M.: Hypergraph-based modeling of ad-hoc business processes. In: D. Ardagna, M. Mecella, J. Yang (eds.) Business Process Management Workshops: BPM 2008 International Workshops, Milano, Italy, September 1-4, 2008. Revised Papers, pp. 278–289. Springer Berlin Heidelberg, Berlin, Heidelberg (2009). https://doi.org/10.1007/978-3-642-00328-8_27
257. Prim, R.: The shortest connecting network and some generalization. Bell Systems Tech. J. **36**, 1389?1401 (1957)
258. Prokhorenkova, L.O.: Global clustering coefficient in scale-free weighted and unweighted networks. Internet Mathematics **12**(1-2), 54–67 (2016). https://doi.org/10.1080/15427951.2015.1092482
259. Prokhorenkova, L.O., Samosvat, E.: Global clustering coefficient in scale-free networks. In: A. Bonato, F.C. Graham, P. Pralat (eds.) Algorithms and Models for the Web Graph - 11th International Workshop, WAW 2014, Beijing, China, December 17-18, 2014, Proceedings, *Lecture Notes in Computer Science*, vol. 8882, pp. 47–58. Springer (2014). https://doi.org/10.1007/978-3-319-13123-8_5
260. Puthiyedth, N., Riveros, C., Berretta, R., Moscato, P.: A new combinatorial optimization approach for integrated feature selection using different datasets: A prostate cancer transcriptomic study. PLOS ONE **10**(6), 1–26 (2015). https://doi.org/10.1371/journal.pone.0127702
261. Puthiyedth, N., Riveros, C., Berretta, R., Moscato, P.: Identification of differentially expressed genes through integrated study of alzheimer?s disease affected brain regions. PLOS ONE **11**(4), 1–29 (2016). https://doi.org/10.1371/journal.pone.0152342
262. Qiao, Y., Wu, J., Zhang, L., Wang, C.: Viral marketing for digital goods in social networks. In: L. Chen, C.S. Jensen, C. Shahabi, X. Yang, X. Lian (eds.) Web and Big Data: First International Joint Conference, APWeb-WAIM 2017, Beijing, China, July 7–9, 2017, Proceedings, Part I, pp. 377–390. Springer International Publishing, Cham (2017). https://doi.org/10.1007/978-3-319-63579-8_29
263. Radivojevic, M., Grujic, J.: Community structure of copper supply networks in the prehistoric balkans: An independent evaluation of the archaeological record from the 7th to the 4th millennium BC. CoRR **abs/1705.05406** (2017). http://arxiv.org/abs/1705.05406
264. Raeder, T., Chawla, N.V.: Market basket analysis with networks. Social Netw. Analys. Mining **1**(2), 97–113 (2011). https://doi.org/10.1007/s13278-010-0003-7
265. Rahman, A., Poirel, C.L., Badger, D., Estep, C., Murali, T.M.: Reverse engineering molecular hypergraphs. IEEE/ACM Trans. Comput. Biology Bioinform. **10**(5), 1113–1124 (2013)
266. Ramaswamy, M., Sarkar, S., Chen, Y.: Using directed hypergraphs to verity rule-based expert systems. IEEE Trans. Knowl. Data Eng. **9**(2), 221–237 (1997)
267. Rapoport, A.: Contribution to the theory of random and biased nets. The bulletin of mathematical biophysics **19**(4), 257–277 (1957). https://doi.org/10.1007/BF02478417
268. Redmond, U., Harrigan, M., Cunningham, P.: Mining dense structures to uncover anomalous behaviour in financial network data. In: M. Atzmueller, A. Chin, D. Helic, A. Hotho (eds.) Modeling and Mining Ubiquitous Social Media - International Workshops MSM 2011, Boston, MA, USA, October 9, 2011, and MUSE 2011, Athens, Greece, September 5, 2011, Revised Selected Papers, *Lecture Notes in Computer Science*, vol. 7472, pp. 60–76. Springer (2011). https://doi.org/10.1007/978-3-642-33684-3_4
269. Reeves-Latour, M., Morselli, C.: Bid-rigging networks and state-corporate crime in the construction industry. Social Networks **51**, 158–170 (2017). https://doi.org/10.1016/j.socnet.2016.10.003
270. Richardson, M., Domingos, P.M.: Mining knowledge-sharing sites for viral marketing. In: KDD, pp. 61–70. ACM (2002)

271. Riveros, C., Mellor, D., Gandhi, K.S., McKay, F.C., Cox, M.B., Berretta, R., Vaezpour, S.Y., Inostroza-Ponta, M., Broadley, S.A., Heard, R.N., Vucic, S., Stewart, G.J., Williams, D.W., Scott, R.J., Lechner-Scott, J., Booth, D.R., Moscato, P., for the ANZgene Multiple Sclerosis Genetics Consortium: A transcription factor map as revealed by a genome-wide gene expression analysis of whole-blood mrna transcriptome in multiple sclerosis. PLOS ONE **5**(12), 1–28 (2010). https://doi.org/10.1371/journal.pone.0014176
272. Rizzi, R., Mahata, P., Mathieson, L., Moscato, P.: Hierarchical clustering using the arithmetic-harmonic cut: Complexity and experiments. PLOS ONE **5**(12), 1–8 (2010). https://doi.org/10.1371/journal.pone.0014067
273. Rodrigues, H.S., Fonseca, M.J.: Viral marketing as epidemiological model. CoRR **abs/1507.06986** (2015)
274. Rodrigues, H.S., Fonseca, M.J.: Can information be spread as a virus? viral marketing as epidemiological model. CoRR **abs/1611.04529** (2016)
275. Rosen, K.H.: Discrete Mathematics and Its Applications, 5th edn. McGraw-Hill Higher Education (2002)
276. Rosso, O.A., Mendes, A., Berretta, R., Rostas, J.A., Hunter, M., Moscato, P.: Distinguishing childhood absence epilepsy patients from controls by the analysis of their background brain electrical activity (ii): A combinatorial optimization approach for electrode selection. Journal of Neuroscience Methods **181**(2), 257–267 (2009). https://doi.org/10.1016/j.jneumeth.2009.04.028
277. Sabar, N.R., Ayob, M., Qu, R., Kendall, G.: A graph coloring constructive hyper-heuristic for examination timetabling problems. Appl. Intell. **37**(1), 1–11 (2012). https://doi.org/10.1007/s10489-011-0309-9
278. Sadasivam, G.S., Saranya, K.G., Karrthik, K.G.: Hypergraph-based wikipedia search with semantics. IJWS **2**(1/2), 66–79 (2013)
279. Samuel, É.: Recherche de motifs dans des images : apport des graphes plans. (searching for patterns in images : what plane graphs can bring). Ph.D. thesis, Jean Monnet University, Saint-Étienne, France (2011). https://tel.archives-ouvertes.fr/tel-00719187
280. Schiller, B., Jager, S., Hamacher, K., Strufe, T.: Stream - A stream-based algorithm for counting motifs in dynamic graphs. In: A. Dediu, F.H. Quiroz, C. Martín-Vide, D.A. Rosenblueth (eds.) Algorithms for Computational Biology - Second International Conference, AlCoB 2015, Mexico City, Mexico, August 4-5, 2015, Proceedings, *Lecture Notes in Computer Science*, vol. 9199, pp. 53–67. Springer (2015). https://doi.org/10.1007/978-3-319-21233-3_5
281. Schlauch, W.E.: Analysis of different random graph models in the identification of network motifs in complex networks. Ph.D. thesis, Kaiserslautern University of Technology, Germany (2017). https://kluedo.ub.uni-kl.de/frontdoor/index/index/docId/4615
282. Schult, D.A., Swart, P.: Exploring network structure, dynamics, and function using networkx. In: Proceedings of the 7th Python in Science Conferences (SciPy 2008), vol. 2008, pp. 11–16 (2008)
283. Seidman, S.B., Foster, B.L.: A graph theoretic generalization of the clique concept. The Journal of Mathematical Sociology **6**(1), 139–154 (1978). https://doi.org/10.1080/0022250X.1978.9989883
284. Shadoan, R., Weaver, C.: Visual analysis of higher-order conjunctive relationships in multidimensional data using a hypergraph query system. IEEE Trans. Vis. Comput. Graph. **19**(12), 2070–2079 (2013)
285. Shannon, P., Markiel, A., Ozier, O., Baliga, N.S., Wang, J.T., Ramage, D., Amin, N., Schwikowski, B., Ideker, T.: Cytoscape: a software environment for integrated models of biomolecular interaction networks. Genome research **13**(11), 2498–2504 (2003)
286. Shen, H., Lv, S., Dong, X., Deng, J., Wang, X., Zhou, X.: Hypergraph modeling and approximation algorithms for the minimum length link scheduling in multiuser MIMO networks. J. Applied Mathematics **2013**, 982,713:1–982,713:9 (2013)
287. Simha, R., Tripathi, R., Thakur, M.: Mining associations using directed hypergraphs. In: ICDE Workshops, pp. 190–197. IEEE Computer Society (2012)

288. Smith, J.C., Ulusal, E., Hicks, I.V.: A combinatorial optimization algorithm for solving the branchwidth problem. Comp. Opt. and Appl. **51**(3), 1211–1229 (2012). https://doi.org/10.1007/s10589-011-9397-z
289. Solé-Ribalta, A., Domenico, M.D., Gómez, S., Arenas, A.: Random walk centrality in interconnected multilayer networks. CoRR **abs/1506.07165** (2015). http://arxiv.org/abs/1506.07165
290. Solé-Ribalta, A., Gómez, S., Arenas, A.: Congestion induced by the structure of multiplex networks. CoRR **abs/1602.07474** (2016). http://arxiv.org/abs/1602.07474
291. Sonuc, S.B., Smith, J.C., Hicks, I.V.: A branch-and-price-and-cut method for computing an optimal bramble. Discrete Optimization **18**, 168–188 (2015). https://doi.org/10.1016/j.disopt.2015.09.005
292. Soramäki, K., Bech, M.L., Arnold, J., Glass, R.J., Beyeler, W.E.: The topology of interbank payment flows. Physica A: Statistical Mechanics and its Applications **379**(1), 317–333 (2007). https://doi.org/10.1016/j.physa.2006.11.093
293. Spadon, G., Scabora, L.C., Araujo, M.V.S., Oliveira, P.H., Machado, B.B., de Sousa, E.P.M., Jr., C.T., Jr., J.F.R.: Complex network tools to understand the behavior of criminality in urban areas. CoRR **abs/1612.06115** (2016). http://arxiv.org/abs/1612.06115
294. Srihari, S., Ng, H.K., Ning, K., Leong, H.W.: Detecting hubs and quasi cliques in scale-free networks. In: ICPR, pp. 1–4. IEEE Computer Society (2008)
295. Srivastava, A.: Motif analysis in the amazon product co-purchasing network. CoRR **abs/1012.4050** (2010). http://arxiv.org/abs/1012.4050
296. Stavropoulos, E.C., Verykios, V.S., Kagklis, V.: A transversal hypergraph approach for the frequent itemset hiding problem. Knowledge and Information Systems **47**(3), 625–645 (2016). https://doi.org/10.1007/s10115-015-0862-3
297. Stell, J.G.: Formal concept analysis over graphs and hypergraphs. In: M. Croitoru, S. Rudolph, S. Woltran, C. Gonzales (eds.) Graph Structures for Knowledge Representation and Reasoning - Third International Workshop, GKR 2013, Beijing, China, August 3, 2013. Revised Selected Papers, *Lecture Notes in Computer Science*, vol. 8323, pp. 165–179. Springer (2013). https://doi.org/10.1007/978-3-319-04534-4_11
298. Strehl, A., Ghosh, J.: Cluster ensembles — A knowledge reuse framework for combining multiple partitions. Journal of Machine Learning Research **3**, 583–617 (2002)
299. Subramani, M.R., Rajagopalan, B.: Knowledge-sharing and influence in online social networks via viral marketing. Commun. ACM **46**(12), 300–307 (2003)
300. Sun, X.Q., Shen, H.W., Cheng, X.Q., Zhang, Y.: Detecting anomalous traders using multi-slice network analysis. Physica A: Statistical Mechanics and its Applications **473**(Supplement C), 1–9 (2017). https://doi.org/10.1016/j.physa.2016.12.052
301. Tang, J., Tang, X., Yuan, J.: Profit maximization for viral marketing in online social networks. In: ICNP, pp. 1–10. IEEE Computer Society (2016). https://doi.org/10.1109/ICNP.2016.7784445
302. Taramasco, C., Cointet, J.P., Roth, C.: Academic team formation as evolving hypergraphs. Scientometrics **85**(3), 721–740 (2010). https://doi.org/10.1007/s11192-010-0226-4
303. Tayebi, M.A., Glässer, U.: Social Network Analysis in Predictive Policing - Concepts, Models and Methods. Lecture Notes in Social Networks. Springer (2016). https://doi.org/10.1007/978-3-319-41492-8
304. Theodoridis, A., Kotropoulos, C., Panagakis, Y.: Music recommendation using hypergraphs and group sparsity. In: IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2013, Vancouver, BC, Canada, May 26-31, 2013, pp. 56–60. IEEE (2013). https://doi.org/10.1109/ICASSP.2013.6637608
305. Thomassen, C.: Kuratowski's theorem. Journal of Graph Theory **5**(3), 225–241 (1981)
306. Thomassen, C.: The graph genus problem is np-complete. J. Algorithms **10**(4), 568–576 (1989). https://doi.org/10.1016/0196-6774(89)90006-0

307. Tomita, E., Tanaka, A., Takahashi, H.: The worst-case time complexity for generating all maximal cliques and computational experiments. Theoretical Computer Science **363**(1), 28–42 (2006). https://doi.org/10.1016/j.tcs.2006.06.015. Computing and Combinatorics
308. Torres, L.M., Wagler, A.K.: Analyzing the dynamics of deterministic systems from a hypergraph theoretical point of view. RAIRO - Operations Research **47**(3), 321–330 (2013)
309. Tsourakakis, C.E., Bonchi, F., Gionis, A., Gullo, F., Tsiarli, M.A.: Denser than the densest subgraph: extracting optimal quasi-cliques with quality guarantees. In: KDD, pp. 104–112. ACM (2013)
310. Vazirani, V.V.: Approximation Algorithms. Springer-Verlag New York, Inc., New York, NY, USA (2001)
311. Veremyev, A., Prokopyev, O.A., Butenko, S., Pasiliao, E.L.: Exact mip-based approaches for finding maximum quasi-cliques and dense subgraphs. Comp. Opt. and Appl. **64**(1), 177–214 (2016)
312. Vieira, M.E.S., López-Ardao, J.C., Fernández-Veiga, M., Rodríguez-Pérez, M., López-García, C.: Mining relationships in learning-oriented social networks. Comp. Applic. in Engineering Education **25**(5), 769–784 (2017). https://doi.org/10.1002/cae.21835
313. Villalobos-Cid, M., Chacón, M., Zitko, P., Inostroza-Ponta, M.: A new strategy to evaluate technical efficiency in hospitals using homogeneous groups of casemix. Journal of Medical Systems **40**(4), 103 (2016). https://doi.org/10.1007/s10916-016-0458-9
314. Vörös, A., Snijders, T.A.: Cluster analysis of multiplex networks: Defining composite network measures. Social Networks **49**(Supplement C), 93–112 (2017). https://doi.org/10.1016/j.socnet.2017.01.002
315. Wakolbinger, T., Nagurney, A.: Dynamic supernetworks for the integration of social networks and supply chains with electronic commerce: modeling and analysis of buyer–seller relationships with computations. NETNOMICS: Economic Research and Electronic Networking **6**(2), 153–185 (2004). https://doi.org/10.1007/s11066-004-4339-x
316. Wang, J., Cheng, J.: Truss decomposition in massive networks. PVLDB **5**(9), 812–823 (2012). http://vldb.org/pvldb/vol5/p812_jiawang_vldb2012.pdf
317. Wang, L., Liu, S., Pan, L., Wu, L., Meng, X.: Enterprise relationship network: Build foundation for social business. In: 2014 IEEE International Congress on Big Data, Anchorage, AK, USA, June 27 - July 2, 2014, pp. 347–354. IEEE (2014). https://doi.org/10.1109/BigData.Congress.2014.57
318. Wang, X., El-Farra, N.H., Palazoglu, A.: Proactive reconfiguration of heat-exchanger supernetworks. Industrial & Engineering Chemistry Research **54**(37), 9178–9190 (2015). https://doi.org/10.1021/acs.iecr.5b00598
319. Wang, Y., Zheng, B.: Hypergraph index: an index for context-aware nearest neighbor query on social networks. Social Netw. Analys. Mining **3**(4), 813–828 (2013)
320. Wang, Z., Chen, Q., Hou, B., Suo, B., Li, Z., Pan, W., Ives, Z.G.: Parallelizing maximal clique and k-plex enumeration over graph data. J. Parallel Distrib. Comput. **106**, 79–91 (2017)
321. Wang, Z., Zhang, F., Wang, Z.: Research of return supply chain supernetwork model based on variational inequalities. In: 2007 IEEE International Conference on Automation and Logistics, pp. 25–30 (2007). https://doi.org/10.1109/ICAL.2007.4338524
322. WANG Zhong-Tuo, W.Z.P.: Elementary study of supernetworks. Chinese Journal of Management **5**(1), 1 (2008). http://202.114.18.199/Jwk_glxb//EN/abstract/article_8930.shtml
323. Wei, K., Dinneen, M.J.: Runtime analysis comparison of two fitness functions on a memetic algorithm for the clique problem. In: Proceedings of the IEEE Congress on Evolutionary Computation, CEC 2014, Beijing, China, July 6-11, 2014, pp. 133–140. IEEE (2014). https://doi.org/10.1109/CEC.2014.6900359
324. Weihe, K.: Covering trains by stations or the power of data reduction. In: R. Battiti, A. Bertossi (eds.) Proceedings of "Algorithms and Experiments" (ALEX98), Trento, Italy, Feb. 9-11, pp. 1–8. Berlin, Heidelberg (1998)

325. Weihe, K.: Motifs in networks. In: A.S. Schulz, M. Skutella, S. Stiller, D. Wagner (eds.) Gems of Combinatorial Optimization and Graph Algorithms, pp. 59–68. Springer (2015). https://doi.org/10.1007/978-3-319-24971-1_6
326. Whelan, C., Sönmez, M.K.: Computing graphlet signatures of network nodes and motifs in cytoscape with graphletcounter. Bioinformatics **28**(2), 290–291 (2012). https://doi.org/10.1093/bioinformatics/btr637
327. Williamson, D.P., Shmoys, D.B.: The Design of Approximation Algorithms, 1st edn. Cambridge University Press, New York, NY, USA (2011)
328. Wu, H., Zhang, Z., Yue, K., Zhang, B., Liu, W.: Maximizing the cooperative influence spread in a social network oriented to viral marketing. In: APWeb Workshops, *Lecture Notes in Computer Science*, vol. 9865, pp. 3–15 (2016)
329. Wu, L., He, M., Han, Y.: Hypergraph clustering-based cloud manufacturing service management method. In: J. Hou, A.J.C. Trappey, C. Wu, K. Chang, C. Liao, W. Shen, J.A. Barthès, J. Luo (eds.) Proceedings of the IEEE 18th International Conference on Computer Supported Cooperative Work in Design, CSCWD 2014, Hsinchu, Taiwan, May 21-23, 2014, pp. 220–225. IEEE (2014). https://doi.org/10.1109/CSCWD.2014.6846845
330. Xiao, M., Lin, W., Dai, Y., Zeng, Y.: A fast algorithm to compute maximum *k*-plexes in social network analysis. In: AAAI, pp. 919–925. AAAI Press (2017)
331. Xu, H., Zeng, G.: Modeling and verifying composite dynamic evolution of software architectures using hypergraph grammars. International Journal of Software Engineering and Knowledge Engineering **23**(6), 775–800 (2013)
332. Xue, J.: Graph coloring. In: C.A. Floudas, P.M. Pardalos (eds.) Encyclopedia of Optimization, Second Edition, pp. 1444–1448. Springer (2009). https://doi.org/10.1007/978-0-387-74759-0_253
333. Xue, W., Xiaoyu, L., Niu, B.: An advanced maximum utility spectrum allocation algorithm based on graph coloring theory. Ad Hoc & Sensor Wireless Networks **36**(1-4), 69–86 (2017). http://www.oldcitypublishing.com/journals/ahswn-home/ahswn-issue-contents/ahswn-volume-36-number-1-4-2017/ahswn-36-1-4-p-69-86/
334. Yang, T., Zhang, R., Cheng, X., Yang, L.: Graph coloring based resource sharing (GCRS) scheme for D2D communications underlaying full-duplex cellular networks. IEEE Trans. Vehicular Technology **66**(8), 7506–7517 (2017). https://doi.org/10.1109/TVT.2017.2657791
335. Yannakakis, M.: Perspectives on database theory. In: Proceedings of the IEEE Symposium on the Foundations of Computer Science, pp. 224–246 (1995)
336. Yaros, J.R., Imielinski, T.: Crowdsourced stock clustering through equity analyst hypergraph partitioning. In: Proceedings of the 2013 IEEE Conference on Computational Intelligence for Financial Engineering & Economics, CIFEr 2013, IEEE Symposium Series on Computational Intelligence (SSCI), 16-19 April 2013, Singapore, pp. 110–117. IEEE (2013). https://doi.org/10.1109/CIFEr.2013.6611705
337. Yin, G., Wei, J., Dong, H.: A cellular automaton based network diffusion model: Preparation for more scalable viral marketing. In: CTS, pp. 308–315. IEEE (2012)
338. Yoshida, T., Yamada, Y.: A community structure-based approach for network immunization. Computational Intelligence **33**(1), 77–98 (2017). https://doi.org/10.1111/coin.12082
339. Yu, X., Yin, X., Han, W., Gao, J., Gu, X.: Scalable routing in 3d high genus sensor networks using graph embedding. In: A.G. Greenberg, K. Sohraby (eds.) Proceedings of the IEEE INFOCOM 2012, Orlando, FL, USA, March 25-30, 2012, pp. 2681–2685. IEEE (2012). https://doi.org/10.1109/INFCOM.2012.6195678
340. Zais, M., Laguna, M.: A graph coloring approach to the deployment scheduling and unit assignment problem. J. Scheduling **19**(1), 73–90 (2016). https://doi.org/10.1007/s10951-015-0434-0
341. Zhan, H., Madduri, K.: Analyzing community structure in networks. In: 2017 IEEE International Parallel and Distributed Processing Symposium Workshops, IPDPS Workshops 2017, Orlando / Buena Vista, FL, USA, May 29 - June 2, 2017, pp. 1540–1549. IEEE Computer Society (2017). https://doi.org/10.1109/IPDPSW.2017.154

342. Zhang, H., Wu, W., Zhao, L.: A study of knowledge supernetworks and network robustness in different business incubators. Physica A: Statistical Mechanics and its Applications **447**, 545–560 (2016). https://doi.org/10.1016/j.physa.2015.12.051
343. Zhang, Y., Wang, J., Zeng, Z., Zhou, L.: Parallel mining of closed quasi-cliques. In: IPDPS, pp. 1–10. IEEE (2008)
344. Zhao, P., Li, Y., Xie, H., Wu, Z., Xu, Y., Ma, R.T.B., Lui, J.C.S.: Impact of online activities on influence maximization: A random walk approach. CoRR **abs/1602.03966** (2016)
345. Zhu, W.Y., Peng, W.C., Chen, L.J., Zheng, K., Zhou, X.: Exploiting viral marketing for location promotion in location-based social networks. ACM Trans. Knowl. Discov. Data **11**(2), 25:1–25:28 (2016). https://doi.org/10.1145/3001938
346. Zou, Z., Zhu, R.: Truss decomposition of uncertain graphs. Knowledge and Information Systems **50**(1), 197–230 (2017). https://doi.org/10.1007/s10115-016-0943-y
347. Mohammad Nazmul Haque, Luke Mathieson, and Pablo Moscato. A memetic algorithm for community detection by maximising the connected cohesion. In *2017 IEEE Symposium Series on Computational Intelligence, SSCI 2017, Honolulu, HI, USA, November 27 - Dec. 1, 2017*, pages 2475–2482, 2017.

Chapter 8
Centrality in Networks: Finding the Most Important Nodes

Sergio Gómez

Abstract Real networks are heterogeneous structures, with edges unevenly distributed among nodes, presenting community structure, motifs, transitivity, rich clubs, and other kinds of topological patterns. Consequently, the roles played by nodes in a network can differ greatly. For example, some nodes may be connectors between parts of the network, others may be central or peripheral, etc. The objective of this chapter is to describe how we can find the most important nodes in networks. The idea is to define a centrality measure for each node in the network, sort the nodes according to their centralities, and fix our attention to the first ranked nodes, which can be considered as the most relevant ones with respect to this centrality measure.

Keywords Centrality · Betweenness centrality · Closeness centrality · Degree centrality · Eigenvector centrality · Hubs and authorities centrality · Katz centrality · PageRank centrality · Random walk centrality · Versatility centrality

8.1 Introduction

Datasets in business and consumer analytics can be frequently represented in the form of networks, in which the nodes represent any kind of item, e.g., products, consumers, brands, firms, etc., while the links represent relationships between them. For example, in co-purchasing networks, the links could account for pairs of products bought together, whereas in international trade networks the edges could represent the amount of a product which is exported from one country to another. The possibilities are infinite, and the extraction of information from these networks

S. Gómez (✉)
Universitat Rovira i Virgili, Tarragona, Spain
e-mail: sergio.gomez@urv.cat

P. Moscato, N. J. de Vries (eds.), *Business and Consumer Analytics: New Ideas*,
https://doi.org/10.1007/978-3-030-06222-4_8

is the object of study in several fields, from complex networks and complex systems to data science, among others. Here we aim at finding the most important nodes in a network, which could be crucial in many business applications.

The importance of a node in a complex network depends on the structural characteristic or dynamic behaviour we could be interested in. Consequently, the literature is full of different definitions, all of them perfectly meaningful under specific set-ups. Our objective is to explain the rationale behind the most widely used centrality measures, to be able to decide which one is the more adequate for our needs. Most of them are easy to describe and understand, some are also easy to calculate with the appropriate tools, while others represent a computational challenge which requires the use of complex algorithms which are not easy to implement. Fortunately, several software applications and packages exist which simplify the finding of the centralities of the nodes in complex networks.

The structure of this section is as follows. First, after this short introduction, we will introduce in Sect. 8.2 the mathematical notation to represent complex networks, which is needed to formalize the different network centralities. Next, we will describe in Sect. 8.3 the most relevant approaches and definitions for the centralities of the nodes in unweighted networks, which represent the core of this chapter of the book. Then, we will show in Sects. 8.4 and 8.5 how it is possible to extend the centralities to cope with two different types of more general networks: weighted networks and multilayer networks. This will be followed in Sect. 8.6 by the calculation of the most central nodes for several synthetic and real networks. In Sect. 8.7 we enumerate the main software tools to calculate the centrality in networks, and we conclude in Sect. 8.8 with a summary of centrality in networks and their applications.

8.2 Mathematical Notation

The main mathematical object in the study of complex networks is the adjacency matrix $A = (a_{ij})$, which encodes the full topology of the network or graph: $a_{ij} = 1$ if there is an edge from node i to node j, and $a_{ij} = 0$ otherwise.[1] We suppose the network has N nodes, thus $i, j \in \{1, \dots, N\}$. If the direction of the links is not important, the network is called undirected, and the adjacency matrix is symmetric, $A = A^T$, where A^T denotes the transpose of matrix A. For undirected networks, the degree k_i of a node is its number of neighbours, and is calculated as

[1] In some texts the adjacency matrix is defined in the opposite direction, i.e., a_{ij} is used to encode an edge from node j to node i, for example, in [44]. This is an important issue to care about when dealing with directed networks.

$$k_i = \sum_{j=1}^{N} a_{ij} \,. \tag{8.1}$$

Directed networks require the distinction between the links that arrive to a node and those that depart from it, therefore it is convenient to distinguish between the output and input degrees:

$$k_i^{\text{out}} = \sum_{j=1}^{N} a_{ij} \,, \tag{8.2}$$

$$k_i^{\text{in}} = \sum_{j=1}^{N} a_{ji} \,. \tag{8.3}$$

Of course, if the network is undirected, $k_i^{\text{out}} = k_i^{\text{in}} = k_i$. We will try to describe the centrality measures in the general case of directed networks, since undirected networks can be considered just as particular cases. However, there are definitions of centrality which do not make sense or cannot be calculated for certain kinds of networks, thus we will explicitly explain the applicability of each centrality type. We will also suppose there are no self-loops in the network, thus all the diagonal elements of the adjacency matrix are zero, $a_{ii} = 0$. The number of edges is calculated by just taking the sum of all the components of the adjacency matrix:

$$2L = \sum_{i=1}^{N} \sum_{j=1}^{N} a_{ij} \,. \tag{8.4}$$

The number of edges is L for undirected networks, but $2L$ for directed ones. The reason is that the adjacency matrix of undirected networks counts every edge twice, $a_{ij} = a_{ji} = 1$.

8.3 Centrality in Networks

The list of centralities we are going to describe is the following:

- Degree centrality
- Closeness centrality
- Betweenness centrality
- Eigenvector centrality
- Katz centrality
- Hubs and authorities centrality
- PageRank centrality
- Random walk centralities.

A more complete review on centrality, covering many other definitions [40], algorithms [36], and several advanced concepts [41] (personalization, axiomatization, stability, and sensitivity) can be found in [17].

8.3.1 Degree Centrality

The first and simplest proposal of a centrality measure for the nodes in a network is the degree,

$$C_i^{(\mathrm{deg})} = k_i \,. \tag{8.5}$$

This is a concept which was developed in the context of social networks a long time ago [30, 57]. The idea was that a person having many direct connections to other people must be central with respect to the communication between them, acquiring a sense of being in the mainstream of information. On the contrary, people with a low degree could miss most of the information flowing in the network, thus playing a residual role.

Nodes with high degree, clearly above the average in the network, are called hubs. The discovery that many real-world networks have power-law degree distributions [4], with only a few hubs collecting a great proportion of the overall links in the network, was in fact one of the cornerstones in the development of the actual theory of complex networks.

Sometimes it is useful to normalize the centralities considering their maximum value, which for the degree equals $N - 1$, thus

$$C_i^{(\mathrm{deg,norm})} = \frac{k_i}{N-1} \,. \tag{8.6}$$

However, normalization is usually not needed, since what matters is the rank of the nodes after sorting them according to the selected centrality measure (which does not change with normalization).

Several additional centrality measures were defined as variants of the degree (see, e.g., [32, 48, 52, 57]), but they have become outdated, so we just skip them.

The degree is a simple and effective centrality measure for undirected networks, but not for directed ones, in which we have to distinguish between incoming and outgoing links. A possible approach could be to take as centrality the sum, $k_i^{\mathrm{in}} + k_i^{\mathrm{out}}$, i.e., the total number of connections discarding their directionality, or the average of both input and output degrees, $(k_i^{\mathrm{in}} + k_i^{\mathrm{out}})/2$; the average is more convenient because it coincides with the degree when applied to undirected networks:

$$C_i^{(\mathrm{deg,avg})} = \frac{k_i^{\mathrm{in}} + k_i^{\mathrm{out}}}{2} \,. \tag{8.7}$$

Another alternative consists in defining two degree centralities, one for the incoming and the other for the outgoing links, since they measure different things: a node with high input degree centrality represents a node which is in good position to receive information, while large output degree centralities correspond to important sources of information:

$$C_i^{(\text{deg,out})} = k_i^{\text{out}}\,, \tag{8.8}$$

$$C_i^{(\text{deg,in})} = k_i^{\text{in}}\,, \tag{8.9}$$

Now, it becomes clear why the importance of a node is closely related to the process or property we are interested in, since even degree centrality admits several diverging interpretations in directed networks.

8.3.2 Closeness Centrality

If you have items distributed within a circle, its centre has the property that all the items are at a distance equal or smaller than the radius, while other positions may be as much as twice that distance. This suggests that a measure of centrality in networks could consider the distances to the rest of the nodes, and thus central nodes would be close to all of them. The advantage of being central in this way comes from the possibility of sending or broadcasting information, being sure the time needed to reach the whole network is as short as possible. Closeness centrality is based on this idea: for each node, you calculate the distance to all the other vertices in the network, and define a centrality in which shorter distances imply higher closeness centrality, and vice versa. There are several ways of expressing this concept mathematically. First, let us call d_{ij} the distance between nodes i and j. The distance in a graph is defined as the minimum number of hops (following links) needed to move from one node to another, or, in other words, the length of the shortest path between them. Then, the closeness centrality [9, 54] reads

$$C_i^{(\text{clos})} = \frac{1}{\sum\limits_{j=1}^{N} d_{ij}}\,, \tag{8.10}$$

which can be normalized [10] as

$$C_i^{(\text{clos,norm1})} = \frac{N-1}{\sum\limits_{j=1}^{N} d_{ij}}\,, \tag{8.11}$$

or also as

$$C_i^{(\text{clos,norm2})} = \frac{N}{\sum_{j=1}^{N} d_{ij}} . \tag{8.12}$$

The difference between using $N - 1$ or N is irrelevant for the ranking of the nodes. The $N - 1$ makes sense since the distance from a node to itself is always zero, $d_{ii} = 0$, but the N provides simpler expressions for certain analytic derivations.

Here we are supposing the network is connected (strongly connected if directed), otherwise some of the distances are infinity and the closeness centrality of all nodes becomes zero. To avoid these infinities, a simple heuristic consists of replacing each infinite distance by N, i.e., a value larger than all the finite distances. An alternative definition which maintains the infinities and works even if the network is not connected is found by just swapping the reciprocal and sum operations [25]:

$$C_i^{(\text{clos2})} = \frac{1}{N-1} \sum_{\substack{j=1 \\ j \neq i}}^{N} \frac{1}{d_{ij}} , \tag{8.13}$$

where, by convenience, $d_{ij} = \infty$ if there is no path between i and j, i.e., $1/d_{ij} = 0$. The term $1/d_{ii}$ is explicitly excluded from the sum to avoid the corresponding infinity. Equation (8.13) may be viewed as a centrality based on the harmonic mean of the distances, and has the advantage that most of the contribution comes from the distances to the closer nodes.

The calculation of all distances in general unweighted graphs requires a breadth-first search (BFS), which takes $O(L + N)$ time, for each of the nodes, thus representing a total cost $O(NL+N^2)$. For sparse networks we have that $L \sim O(N)$, thus the total cost is reduced to $O(N^2)$. For the weighted networks that we will cover in Sect. 8.4, BFS must be substituted with Dijkstra's algorithm, with a cost $O(L + N \log N)$, thus raising the total cost of calculating the closeness centralities to $O(NL + N^2 \log N)$ in general networks, and to $O(N^2 \log N)$ for sparse ones. Alternatively, the Floyd-Warshall algorithm finds all the shortest paths with an $O(N^3)$ upper bound [27], which may be useful to reduce part of the overhead (but not the total cost) in the application of multiple Dijkstra's algorithms.

Likewise to degree centrality, closeness centrality also admits output and input versions for directed networks, depending on whether the distances are computed from or to the reference node, respectively. Note that distances are not symmetric in directed networks. Since we already have several definitions for the closeness centrality, the addition of input and output closeness centralities multiplies the options. This is important to be aware of, since different software may choose and implement centralities in distinctive ways, thus being not exactly comparable.

8.3.3 Betweenness Centrality

Betweenness is another of the traditional centrality measures developed in the study social science. Here we fix our attention in the nodes which are crossed when you follow shortest paths. A node which falls in the communication paths between many pairs of nodes plays an important role, since it can control the flow of information. Formally, the standard measure for this property is called betweenness centrality [2, 29], and is defined as

$$C_i^{(\text{betw})} = \frac{1}{(N-1)(N-2)} \sum_{\substack{s,d=1 \\ s\neq d\neq i}}^{N} \frac{\sigma_{sd}(i)}{\sigma_{sd}} . \tag{8.14}$$

The sum covers all source/target pairs of nodes, excluding node i, σ_{sd} represents the number of shortest paths from source node s to destination node d, and $\sigma_{sd}(i)$ is the number of those paths that include node i. In other words, the betweenness is the average fraction of paths that cross a node. This expression of the betweenness measure is valid for both directed and undirected networks, and includes the optional normalization factor. If there are no paths between the origin s and the destination d (disconnected graph), then $\sigma_{sd} = 0$ and it becomes necessary to define $\sigma_{sd}(i)/\sigma_{sd} = 0$. An example of a node with high betweenness would be a node which is a bridge between two disconnected parts of the network: to go from one part of the network to the other you are forced to cross the bridge, no matter if this node has just a few links. Betweenness naturally appears in communication dynamics on top of complex networks, e.g., it can be shown that the onset of congestion in a simple model of routing is related to the maximum betweenness of the system [34].

As we have explained for the closeness centrality, the calculation of shortest paths represents a costly task. Fortunately, we may apply the Brandes' algorithm for betweenness centrality, with a cost $O(NL + N^2 \log N)$, which is reduced to $O(NL + N^2)$ for the unweighted networks we have considered so far [16].

Some variations on the definition of betweenness exist; the most remarkable one being the possibility of including node i as both source s and destination d [44], which we have forbidden in our previous definition. The decision of including or not the end-points of the paths when calculating the betweenness depends on the particular dynamics you may be interested in. For example, in routing dynamics in which a queue is attached to each node, it is possible to decide between putting the created packets in the queue of the source node [60], or skipping this queue and enqueuing them directly to the first neighbour in the path [34]. Both alternatives are acceptable, but they lead to slightly different values of the betweenness. Another variant of betweenness is the one which calculates the number of shortest paths at the level of edges, thus defining a link betweenness, the natural extension to links of the vertex betweenness. We are not going to consider link centralities in the rest of this chapter, but it may be useful for the reader to know of their existence and one of their paradigmatic examples.

8.3.4 Eigenvector Centrality

All the previous centrality measures take into account the topological position of nodes in the network, but not the importance of the nodes themselves. It could be desirable, for example, that a node be considered as important if its neighbours are also important. This leads to a recursive definition of centrality, in which the centrality of a node depends on the centralities of the neighbours, which are also unknown. Fortunately, it is possible to write self-consistent equations which can easily be solved using linear algebra techniques. The simplest of this kind of approach consists of defining the centrality of a node as proportional to the sum of the centralities of the neighbours, so as the larger the importance of the neighbours, the more central the node is [13–15]. In mathematical terms,

$$\lambda C_i^{(\mathrm{eig})} = \sum_{j=1}^{N} a_{ji} C_j^{(\mathrm{eig})} , \tag{8.15}$$

where λ is the proportionality constant. The a_{ji} term emphasizes that node i receives the contribution to centrality from its neighbours through the incoming links. For example, in the World Wide Web network, building a website with many links to important sites is easy to build and has no cost, so it gives no information at all. However, receiving hyperlinks from relevant sites is a good indicator of quality, and can be used to measure the centrality of the website.

Equation (8.15) is expressed in matrix form as

$$A^T \mathbf{C}^{(\mathrm{eig})} = \lambda \mathbf{C}^{(\mathrm{eig})} , \tag{8.16}$$

which means the vector of centralities $\mathbf{C}^{(\mathrm{eig})}$ is an eigenvector of A^T (or equivalently, a left-eigenvector of A) with eigenvalue λ. Since the components of the adjacency matrix are all non-negative, we can apply the Perron-Frobenius theorem [31, 51], which ensures that, if the matrix is irreducible, there exists a unique solution of Eq. (8.16) in which all the centralities $C_i^{(\mathrm{eig})}$ are positive (up to positive common factors), and which corresponds to the largest eigenvalue $\lambda > 0$. The matrix is irreducible if the graph is strongly connected (or simply connected, if the network is undirected). For directed networks this condition is difficult to be fulfilled, thus eigenvector centrality is basically used only for undirected networks. Some variants of the eigenvector centrality, such as Katz, HITS, or PageRank, are more adequate for directed networks.

The calculation of the eigenvector centrality can easily be performed by power iteration: initialize all the centralities to one (or to a random positive vector), multiply by A^T, normalize the vector, and repeat the multiplication-normalization steps until convergence. Common normalizations used are those in which the sum of all centralities are either 1 or N. Again, the normalization does not affect the ranking of the nodes, thus any choice is equally acceptable. The cost of power iteration in $O(Lr)$ for general graphs, and $O(Nr)$ for sparse ones, where r is the number of iterations needed until convergence to the desired precision. Convergence is guaranteed if the adjacency matrix has a non-degenerate maximum (in magnitude) eigenvalue, and the initial vector has a non-zero component in the direction of the leading eigenvector. The convergence is geometric with ratio $|\lambda_2/\lambda_1|$, i.e., the quotient between the second and the first eigenvalues, thus it depends on the structure of the whole network and it is impossible to predict the value of r without a computation which may take longer than the calculation of the eigenvector centrality itself.

8.3.5 Katz Centrality

Katz centrality is a proposal that lays between degree and eigenvector centrality. It was introduced as a way of generalizing the degree centrality, taking into account not only the immediate neighbours but also the nodes reachable in larger number of steps [37]. Since you want that the closer the nodes, the larger their influence, a decay parameter $\alpha < 1$ is introduced to weight the contributions of nodes at increasing path lengths. It is defined in this way:

$$C_i^{(\mathrm{katz})} = \sum_{k=1}^{\infty} \sum_{j=1}^{N} \alpha^k (A^k)_{ji} \, . \tag{8.17}$$

The power matrix A^k accounts for the number of paths of length k between every pair of nodes, e.g., $(A^3)_{ji} = \sum_r \sum_s a_{jr} a_{rs} a_{si}$, where the paths start at node j, then go to node r, next to s, and finally arrive to i, for all possible values of the intermediate nodes r and s. Denoting I the identity matrix of order N, and $\mathbf{1}$ the vector of length N with all components equal to 1, we can write

$$\mathbf{C}^{(\mathrm{katz})} = \sum_{k=1}^{\infty} (\alpha A^T)^k \mathbf{1} = \Big((I - \alpha A^T)^{-1} - I \Big) \mathbf{1} \, , \tag{8.18}$$

which after some algebra becomes

$$\mathbf{C}^{(\mathrm{katz})} = \alpha A^T (\mathbf{C}^{(\mathrm{katz})} + \mathbf{1}) \, , \tag{8.19}$$

or in components

$$C_i^{(\mathrm{katz})} = \alpha \sum_{j=1}^{N} a_{ji} (C_j^{(\mathrm{katz})} + 1) \, . \tag{8.20}$$

Equations (8.19) and (8.20) are closely related to the eigenvector centrality Eqs. (8.16) and (8.15), respectively. Basically, the Katz centrality of a node is related to the centralities of the incoming neighbours, likewise eigenvector centrality, but with the addition of one unit per neighbour. This means all nodes have a minimum level of centrality, different from zero, which helps to avoid the problems of eigenvector centrality with non-strongly connected components. Of course, the α parameter has to be small enough to ensure the convergence of Eq. (8.17) and of the iteration process. It can be shown that proper values of the parameter must be in the interval $0 < \alpha < 1/\lambda$, where λ is the maximum eigenvalue of the adjacency matrix A.

Katz centrality can be extended by replacing the vector **1** by any other set of constants:

$$\mathbf{C}^{(\mathrm{katz2})} = \alpha A^T \mathbf{C}^{(\mathrm{katz2})} + \boldsymbol{\beta}\,, \tag{8.21}$$

This is useful to allow each node i to have a minimum centrality β_i, which could be set even from external information of the nodes, unrelated to the network structure. When α approaches zero most of the contribution to the Katz centrality comes from the constant term $\boldsymbol{\beta}$, while α values close to its upper bound $1/\lambda$ give the dominant role to the eigenvector term. In practice, most of the authors use values of the parameter near the upper bound.

Although Katz centrality could be computed with a matrix inversion using Eq. (8.18), it is more efficient to directly solve Eqs. (8.19) or (8.21) by iteration until convergence, in a similar way as for the eigenvector centrality, and equivalent cost $O(Lr)$ for general complex networks.

8.3.6 Hubs and Authorities Centrality

In directed networks, nodes can have very different roles if we consider only the input or output links. The idea of the hyperlink-induced topic search (HITS) approach, also known as hubs and authorities' algorithm [39], is to assign to each node a couple of scores: a hub centrality, which takes into account the role of the node in sending links, and an authority centrality, measuring the capacity of the node to receive links. Following the same approach that eigenvector centrality, the importance as authority depends on the relevance of the hubs that send the incoming links, and the other way around, important hubs give more weight as authorities to the receiver nodes. Denoting $C_i^{(\mathrm{hub})}$ and $C_i^{(\mathrm{auth})}$ the hub and authority centralities of node i, the following recursive definition holds:

$$C_i^{(\mathrm{auth})} = \alpha \sum_{j=1}^{N} a_{ji} C_j^{(\mathrm{hub})}\,, \tag{8.22}$$

$$C_i^{(\mathrm{hub})} = \beta \sum_{j=1}^{N} a_{ij} C_j^{(\mathrm{auth})} . \tag{8.23}$$

In matrix form,

$$\mathbf{C}^{(\mathrm{auth})} = \alpha A^T \mathbf{C}^{(\mathrm{hub})} , \tag{8.24}$$

$$\mathbf{C}^{(\mathrm{hub})} = \beta A \mathbf{C}^{(\mathrm{auth})} , \tag{8.25}$$

which can be combined to form two decoupled equations:

$$A^T A \mathbf{C}^{(\mathrm{auth})} = \gamma \mathbf{C}^{(\mathrm{auth})} , \tag{8.26}$$

$$A A^T \mathbf{C}^{(\mathrm{hub})} = \gamma \mathbf{C}^{(\mathrm{hub})} , \tag{8.27}$$

where $\gamma = (\alpha\beta)^{-1}$. Applying the Perron-Frobenius theorem as for the eigenvector centrality, and realizing that matrices $A^T A$ and AA^T are symmetric, then the authorities and hubs centralities are given by the leading eigenvector of their respective matrices. Moreover, it can be shown that the eigenvalues of $A^T A$ and AA^T are exactly the same, thus the two equations are consistent and γ is the maximum eigenvalue of any of them. Additionally, multiplying both sides of the first equation by A and of the second equation by A^T, we get

$$AA^T (A\mathbf{C}^{(\mathrm{auth})}) = \gamma (A\mathbf{C}^{(\mathrm{auth})}) , \tag{8.28}$$

$$A^T A (A^T \mathbf{C}^{(\mathrm{hub})}) = \gamma (A^T \mathbf{C}^{(\mathrm{hub})}) , \tag{8.29}$$

which means that hubs and authorities centralities are related in the following way:

$$\mathbf{C}^{(\mathrm{auth})} = A^T \mathbf{C}^{(\mathrm{hub})} , \tag{8.30}$$

$$\mathbf{C}^{(\mathrm{hub})} = A \mathbf{C}^{(\mathrm{auth})} . \tag{8.31}$$

This framework was designed to rank web pages, but is perfectly valid for all kinds of directed networks, e.g., citations or trade networks. When the network is undirected the distinction between hubs and authorities disappears, and their centralities coincide with those obtained by eigenvector centrality.

The most effective way of calculating these centralities is by iteration of Eqs. (8.30) and (8.31), with a normalization after each step, and a total cost equivalent to eigenvector and Katz centralities. The result is both hubs and authorities centralities at the same time, and it skips the excessive time consuming matrix multiplications in Eqs. (8.28) and (8.29).

8.3.7 PageRank Centrality

PageRank has become a notorious centrality measure since it lays at the core of the Google search engine. When you make a search query, the PageRank score of each web page is used to sort the results, which are then presented to the user. Of course, PageRank is in fact used in conjunction with other heuristics and criteria, but at least it provides a good starting point.

The rationale behind PageRank is similar to eigenvector centrality, but with a relevant distinction: when a node receives a link from an important source, it is not the same if that site has many links or just a few. If the number is large, the contribution is diluted, and should be penalized. Thus, it seems reasonable to normalize the score of a node by its number of outgoing links, before adding it to the score of the receiver. The full equation for the PageRank centrality is the following [19]:

$$C_i^{(\mathrm{pr})} = \alpha \sum_{j=1}^{N} a_{ji} \frac{C_j^{(\mathrm{pr})}}{k_j^{\mathrm{out}}} + \frac{1-\alpha}{N} \,. \tag{8.32}$$

The constant term plays an equivalent role as in Katz centrality, ensuring the equation has a unique and non-trivial solution for directed networks, while parameter α, known as the dumping factor, controls the fraction of contribution between the eigenvector and constant terms. Note that PageRank is already normalized, $\sum_i C_i^{(\mathrm{pr})} = 1$, as can be easily checked by summing both sides of Eq. (8.32) for all the nodes i. For nodes with no outbound links, $k_j^{\mathrm{out}} = 0$, but the numerator is also zero, thus a simple solution is to replace k_j^{out} by $\max(k_j^{\mathrm{out}}, 1)$; otherwise, the terms $0/0$ are just supposed to be 0.

We may also write Eq. (8.32) in matrix form:

$$\mathbf{C}^{(\mathrm{pr})} = \alpha A^T D^{-1} \mathbf{C}^{(\mathrm{pr})} + \frac{1-\alpha}{N} \mathbf{1} \,, \tag{8.33}$$

where D is the diagonal matrix with elements $D_{ii} = \max(k_i^{\mathrm{out}}, 1)$. In this way, the solution is given by:

$$\begin{aligned} \mathbf{C}^{(\mathrm{pr})} &= \frac{1-\alpha}{N} (I - \alpha A^T D^{-1})^{-1} \mathbf{1} \\ &= \frac{1-\alpha}{N} D (D - \alpha A^T)^{-1} \mathbf{1} \,. \end{aligned} \tag{8.34}$$

Anyhow, the common way of solving Eq. (8.32) is by iteration, as explained for the previous eigenvector, Katz and HITS centralities. The dumping factor was set by the authors to $\alpha = 0.85$, but this is a quite arbitrary selection which can be tuned as desired.

8.3.8 Random Walk Centralities

Looking at Eq. (8.32) for the PageRank, a new interpretation comes out when we realize that

$$P_{ij} = \frac{a_{ij}}{k_i^{\text{out}}} \tag{8.35}$$

represents the probability that a random walker follows a link from node i to node j [43, 49, 61]. Matrix P, which may be written as

$$P = D^{-1} A \,, \tag{8.36}$$

is right stochastic, since $\sum_j P_{ij} = 1$ for all rows i, i.e., $P\mathbf{1} = \mathbf{1}$. The probability π_i that a random walk is found in node i is obtained by solving the eigenvector equation

$$P^T \boldsymbol{\pi} = \boldsymbol{\pi} \,. \tag{8.37}$$

Using P, the PageRank equation becomes

$$\mathbf{C}^{(\text{pr})} = \alpha P^T \mathbf{C}^{(\text{pr})} + \frac{1-\alpha}{N}\mathbf{1} \,. \tag{8.38}$$

This equation corresponds to the dynamics of a random walker which, with probability α, follows a random link of the current node, and with probability $1-\alpha$ jumps to a random node; this behaviour justifies why the second term is also referred to as the teleportation term, and it is necessary to escape from nodes without output links. Moreover, $\mathbf{C}^{(\text{pr})}$ turns out to be the occupation probability of this random walker, thus providing a physical interpretation: PageRank centrality is equal to the probability of the random walker being found at each of the nodes.

If we remove the teleportation term by setting the dumping factor to $\alpha = 1$, the PageRank equation is simplified to $\mathbf{C}^{(\text{pr})} = P^T \mathbf{C}^{(\text{pr})}$, which has a simple solution for unweighted networks: $\mathbf{C}^{(\text{pr})} = \mathbf{k} = \mathbf{C}^{(\text{deg})}$, i.e., the PageRank becomes proportional to the degree. In the general case of directed networks and with teleportation this solution does not hold, but it suggests that PageRank is a kind of modified version of the degree centrality.

We have shown so far that a random walk dynamics on complex networks gives an alternative explanation of PageRank to the one inspired by eigenvector and Katz centrality. However, this is not the only centrality measure that can be defined using random walks. In fact, random walks constitute a good proxy for the spreading of information in networks, and we can take advantage of it to introduce new measures of the importance of nodes. In particular, we are going to briefly describe random-walk closeness centrality and random-walk betweenness centrality [46].

In the definition of betweenness centrality given in Sect. 8.3.3, only nodes crossed by shortest paths are considered. This makes sense for certain dynamics, e.g., vehicles trying to reach their destination minimizing the travel distance, or servers dispatching packets using the standard Internet protocols. The same can be said about closeness centrality, which implicitly assumes that shortest-path distances are the way to go from one node to another. However, if we consider rumours, news, fads, or epidemics, to name a few, their spreading is more random, and for sure they do not follow shortest paths. This is where random walkers stand out, as an alternative and often better model of information spreading that can help in the introduction of additional measures of centrality. In fact, real propagation usually lays somewhere in-between shortest paths and random walks, the two extreme cases.

A measure of random-walk betweenness centrality requires the computation of the probability that a random walk crosses a certain node while travelling between all other pairs of nodes. This is accomplished by introducing a new transition matrix $P^{[d]}$ with an absorbing state at the destination node d (when the random walker arrives to d, it is removed from the system),

$$P_{ij}^{[d]} = \begin{cases} 0 & \text{if } i = d \\ P_{ij} & \text{otherwise,} \end{cases} \tag{8.39}$$

and calculating the expected number of times the random walker crosses node i (in any number of steps) when starting at node s and before reaching the destination d,

$$\begin{aligned} q_{si}^{[d]} &= \sum_{n=0}^{\infty} \frac{1}{k_i^{\text{out}}} \left[(P^{[d]})^n \right]_{si} \\ &= \left[(I - P^{[d]})^{-1} D^{-1} \right]_{si} \\ &= \left[(D - A^{[d]})^{-1} \right]_{si} , \end{aligned} \tag{8.40}$$

where $A^{[d]}$ is defined as in Eq. (8.39). The n-th power of $P^{[d]}$ term expresses the probability of arriving from node s to node i in n steps, and the $1/k_i^{\text{out}}$ adds the condition of leaving that node by any of the links, thus effectively crossing node i.

Since random walkers may be trapped in parts of the network which are not really related to the paths between nodes s and d, it is essential to cancel out, for each link, the flux in opposite directions. For example, the net flux through the link (i, j) is equal to $|q_{si}^{[d]} - q_{sj}^{[d]}|$, which yields a net flux at node i equal to

$$f_i^{(sd)} = \begin{cases} \dfrac{1}{2} \displaystyle\sum_{j=1}^{N} a_{ji} \left| q_{si}^{[d]} - q_{sj}^{[d]} \right| & \text{if } i \neq s, d \\ 1 & \text{otherwise.} \end{cases} \tag{8.41}$$

Finally, random-walk betweenness centrality is obtained by averaging the node flux over all possible origins and destinations,

$$C_i^{(\text{rwbetw})} = \frac{1}{N(N-1)} \sum_{\substack{s,d=1 \\ s \neq d}}^{N} f_i^{(sd)} . \tag{8.42}$$

In this case we have allowed node i to be in the end-points of the paths, unlike for shortest-path betweenness, but we could restore the same semantics by setting $f_s^{(sd)} = f_d^{(sd)} = 0$ in Eq. (8.41) and normalizing the centrality by $(N-1)(N-2)$ instead of $N(N-1)$.

The computing cost of this centrality is high due to the need of finding the inverse of N matrices (one for each absorbing node d), all of size $N \times N$. However, it can be shown that the same solution can be obtained with just one matrix inversion, by choosing an arbitrary node v (e.g., the first one), finding the corresponding inverse matrix $R = (D - A^{[v]})^{-1}$, and calculating the flux of all nodes using the components of matrix R as

$$f_i^{(sd)} = \begin{cases} \dfrac{1}{2} \displaystyle\sum_{j=1}^{N} a_{ji} \left|(r_{si} - r_{di}) - (r_{sj} - r_{dj})\right| & \text{if } i \neq s, d \\ 1 & \text{otherwise.} \end{cases} \tag{8.43}$$

In this way, the total cost of computing the random-walk betweenness centrality becomes $O((N+L)N^2)$, where $O(N^3)$ corresponds to the matrix inversion and $O(N^2 L)$ to the calculation of the net flux of all nodes. For sparse matrices, the cost is reduced to $O(N^3)$.

Random-walk closeness centrality follows the same idea: the distance between two nodes s and d is replaced by the average time needed by a random walker to reach d when starting the walk at s. This quantity receives the name of mean first-passage time (MFPT), and has the property of not being symmetric even for undirected networks. The MFPT in which origin and destination are the same node is known as mean return time. The derivation of MFPT matrix T is quite involved [44, 62], but we can give the recipe for its calculation, based on the fundamental matrix Z of the random walk dynamics:

$$Z = (I - (P - P^{\infty}))^{-1} , \tag{8.44}$$

where $P^{\infty} = \lim_{n \to \infty} P^n = \mathbf{1}\boldsymbol{\pi}^T$. Then, the MFPT from node s to node d is given by

$$T_{sd} = \frac{z_{dd} - z_{sd}}{\pi_d} , \tag{8.45}$$

the average first-passage time becomes

$$h_d = \frac{1}{N} \sum_{s=1}^{N} \mathrm{T}_{sd} , \tag{8.46}$$

and the random-walk closeness centrality is just defined as

$$C_i^{(\mathrm{rwclos})} = \frac{1}{h_i} . \tag{8.47}$$

Note that we have based the definition on the paths arriving to the node for which we are calculating the centrality, thus using the same choice as for the PageRank and other centralities. The cost for random-walk closeness centrality comes from the matrix inversion needed to find the fundamental matrix Z, thus it is $O(N^3)$.

8.4 Centrality in Weighted Networks

Weighted networks are those for which a certain value is assigned to each of the edges. The standard interpretation is that the larger the weight, the more connected or related the nodes are. Flows, similarities, strengths of social ties, capacities, correlations, intensities, and proximities are examples of this kind of weighted relationships. The matrix of weights w_{ij} may be seen as a generalization of the adjacency matrix, in the sense that we may consider that a null weight corresponds to the absence of a link, and in many cases we may just replace the adjacency matrix by the weights matrix to obtain generalizations of the unweighted concepts, centrality being one of them [1, 5]. Note also that the adjacency matrix is recovered if we suppose all the weights are equal to 1. The natural generalization of the degree is called the strength of the node and is given by

$$w_i = \sum_{j=1}^{N} w_{ij} . \tag{8.48}$$

Directed networks require the distinction between input and output strengths,

$$w_i^{\mathrm{out}} = \sum_{j=1}^{N} w_{ij} , \tag{8.49}$$

$$w_i^{\mathrm{in}} = \sum_{j=1}^{N} w_{ji} , \tag{8.50}$$

and the total strength of the network reads

$$2w = \sum_{i=1}^{N} \sum_{j=1}^{N} w_{ij} \, . \tag{8.51}$$

With these ingredients, the generalization of the degree centrality would be the strength centrality, which could be normalized using the maximum strength. In the same way, eigenvector, hubs and authorities, and PageRank centralities are obtained by simple substitution of the adjacency matrix components and the degrees by weights and strengths, respectively. The Katz centrality also admits this treatment in its interpretation as an eigenvalue problem, but it is questionable the meaning of the powers of the weights matrix.

For the random-walk centralities, the weights allow to have different transition probabilities from a node to each of its neighbours,

$$P_{ij} = \frac{w_{ij}}{w_i^{\text{out}}} \, , \tag{8.52}$$

and once they are determined, the definitions of PageRank, random-walk betweenness, and random-walk closeness remain the same.

The problem arises when we want to generalize centralities based on distances, like closeness or betweenness. The first option consists in discarding the weights, something which also applies to the cases above. However, when the relationship between nodes represents distances, they cannot be ignored. For example, in geographical and transportation networks we may have the distances between connected nodes available. Now, the shortest path between two nodes is not the path with the least number of hops, but the path for which the sum of the distances of the edges (the length of the path) is the smallest one. In these cases, the definition of closeness and betweenness centralities does not need to be changed, but the algorithms to calculate them require important modifications. For instance, while a breadth-first search is enough to find the distances in unweighted networks, a Dijkstra's algorithm is necessary to cope with the distances of the edges.

8.5 Centrality in Multilayer Networks

Another important class of networks which deserves special treatment with regard to centrality is that of interconnected multilayer networks [12, 38]. In multilayer networks the nodes are distributed in layers, with intra-layer and inter-layer links connecting nodes in the same and different layers, respectively. If every node represents a different entity, no matter in which layer it is located, it is perfectly meaningful to calculate the centralities of the nodes as if the network were not multilayer, i.e., disregarding the structure in layers. Alternatively, we could just find the centralities of the nodes inside the layers, considering each layer as a separate network, ignoring the inter-layer links. These procedures lead to two centralities per node, one global and the other local to the layer. Thus, a node can be at the same time very central in a layer, but not so important for the whole multilayer network.

In interconnected multilayer networks the same node may be present in several layers at the same time, and this fact affects the definition of centrality itself. If one node has a different centrality in each layer, how do we have to aggregate them to produce a single centrality for the node? There have been several proposals of ways to define eigenvector centralities [35, 58] and PageRank [8] for multiplex networks, which are the particular case of multilayer interconnected networks in which inter-layer links only connect instances of the same node in different layers, but not different nodes. A more general framework makes use of the tensorial formulation of multilayer networks [23], which has allowed a grounded development of the extension of centrality measures to general multilayer networks [24, 59]. The remarkable finding is that centrality in interconnected multilayer networks reveals the most versatile nodes, in the sense that the highest centrality (versatility) is assigned to nodes which are not necessarily very central in any layer but which are fundamental for the cohesiveness and integration of the whole structure [24].

We are not going to develop all the theory of centrality (versatility) for multilayer networks, but it is easy to show the main ideas with eigenvector centrality. First, the replacement of the adjacency matrix for multilayer networks is the adjacency tensor $M^{i\alpha}_{j\beta}$, representing the links between nodes i in layer α and nodes j in layer β. The eigentensor equation becomes:

$$\sum_{i=1}^{N}\sum_{\alpha=1}^{U} M^{i\alpha}_{j\beta} C^{(\text{eigvers})}_{i\alpha} = \lambda C^{(\text{eigvers})}_{j\beta}\,, \tag{8.53}$$

where U is the number of layers. After solving this equation for the largest eigenvalue, the final eigenvector centrality (versatility) is obtained by summing up the contributions at each layer:

$$C^{(\text{eigvers})}_{j} = \sum_{\beta=1}^{U} C^{(\text{eigvers})}_{j\beta}\,. \tag{8.54}$$

Note that Eq. (8.53) takes into account the complete structure of the multilayer network, unlike some approaches in which layers are analysed as isolated layers, thus losing the information of the inter-layer connectivity.

In a similar way, centralities based on distances or random walkers make use of the full structure of the network, but at the same time the multiplicity of the nodes in the different layers poses restrictions on the paths. For example, although paths may change layer crossing inter-layer links, it is natural to consider that shortest paths from an origin to a destination must start and end, respectively, in the layers that minimize the distance. As a consequence, shortest paths in multilayer networks cannot be found by iterating over all pairs of nodes, ignoring the multilayer structure. This demonstrates the fundamental differences between standard and interconnected multilayer networks, and how they affect the structural and dynamical properties on top of them.

Table 8.1 Most central nodes of undirected network in Figs. 8.1 and 8.2

Centrality	Most central	Second most central	Third most central
Degree	28	19, 24	6, 7, 10, 11, 16, 18, 20, 22
Closeness	18	24	16
Eigenvector	19	20, 22	21, 23
Katz	28	19	20, 22
Betweenness	28	24	18
PageRank	28	24	36, 39
Random-walk betweenness	28	18	24
Random-walk closeness	18	16	24

Table 8.2 Most central nodes of directed network in Figs. 8.3 and 8.4

Centrality	Most central	Second most central	Third most central
Input degree	8, 14	25, 27	23
Output degree	13	1, 12	23, 27
Input closeness	14	22	20
Output closeness	13	19	16
Eigenvector	23	27	25
Katz	27	25	28, 29, 30, 31
Hub	13	12	1
Authority	14	8	7
Betweenness	20	23	8
PageRank	31	30	27

8.6 Examples

We have designed a couple of small networks, one undirected and the other directed, to grasp the differences between the most central nodes according to each of the definitions of centrality we have elaborated above. Figures 8.1 and 8.2 show them for the undirected network, while Figs. 8.3 and 8.4 for the directed network. In addition, Tables 8.1 and 8.2 enumerate the most, second most, and third most central nodes for the undirected and directed networks, respectively. These networks have been designed in such a way that each centrality measure leads to different most central nodes, with few coincidences, to emphasize the topological features which distinguish them.

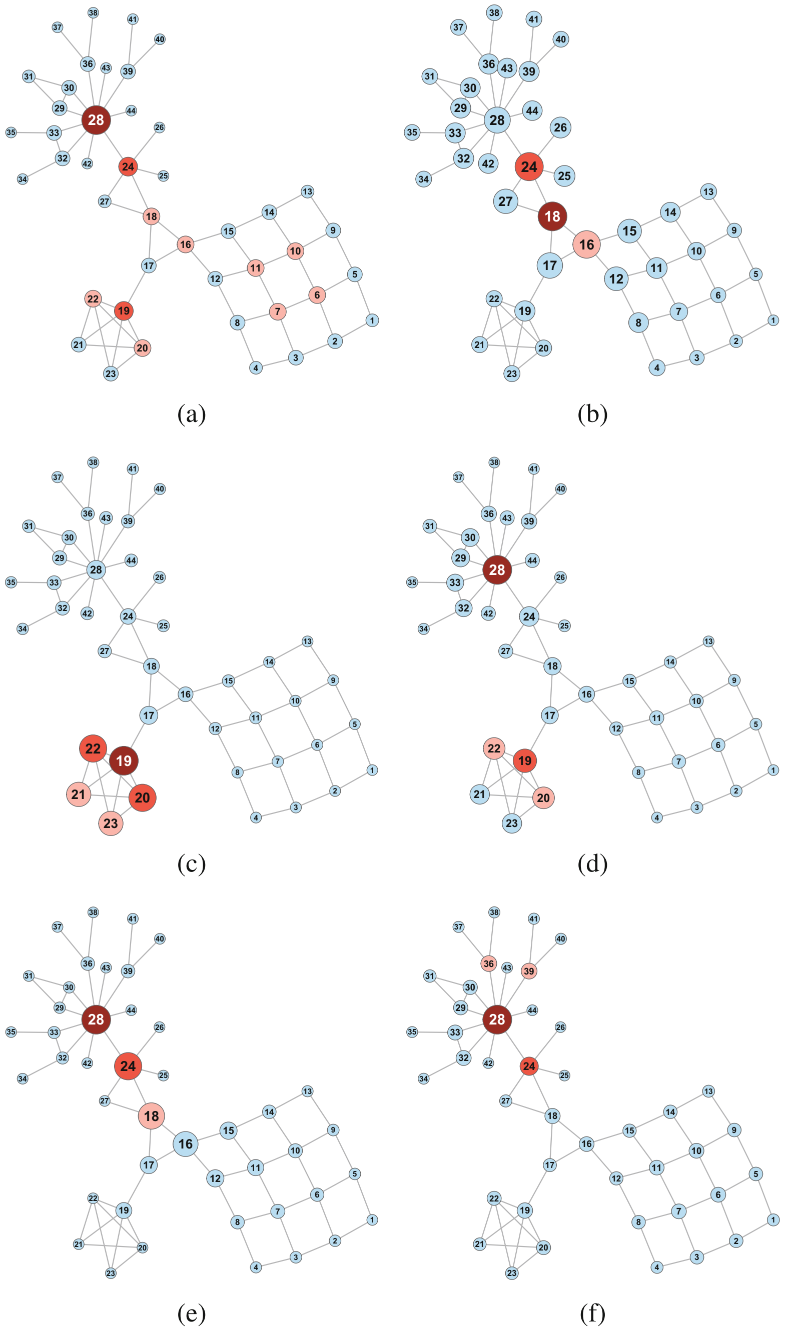


Fig. 8.1 Six different types of centralities for an undirected network. Nodes with highest centrality in dark red (and white node label), second largest centrality in red, third largest centrality in light red, and rest of nodes in blue. Sizes proportional to centrality with an offset. (**a**) Degree centrality. (**b**) Closeness centrality. (**c**) Eigenvector centrality. (**d**) Katz centrality. (**e**) Betweenness centrality. (**f**) PageRank centrality

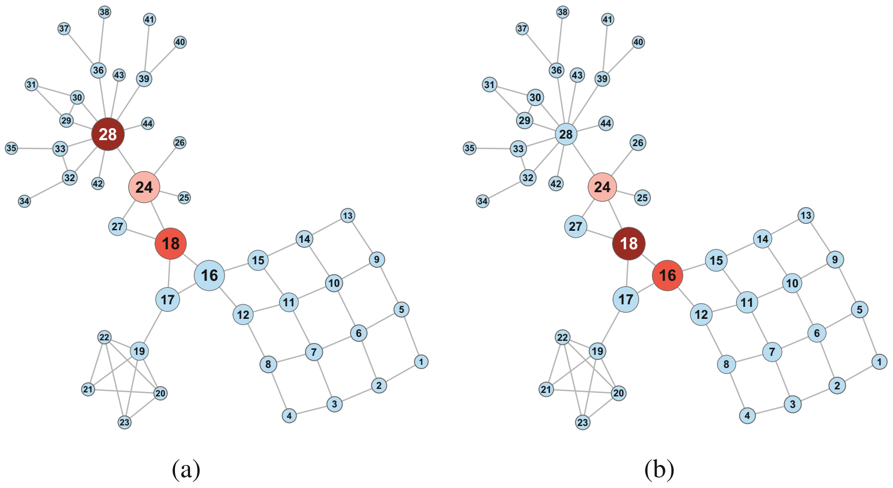


Fig. 8.2 Two random-walk based centralities for an undirected network. As in the previous figure, nodes with highest centrality in dark red (and white node label), second largest centrality in red, third largest centrality in light red, and rest of nodes in blue. Sizes proportional to centrality with an offset. (**a**) Random-walk betweenness centrality. (**b**) Random-walk closeness centrality

Looking at Figs. 8.1, 8.2, 8.3, and 8.4 we observe several patterns that deserve a few words. First, the symmetries present in the networks are responsible for the existence of several distinct nodes with exactly the same centralities. For example, in the undirected unweighted network in Figs. 8.1 and 8.2, nodes 20 and 22 have always the same centrality, no matter the definition we choose, and the same happens for many other tuples of nodes: (12, 15), (21, 23), (25, 26), (36, 39), (37, 38, 40, 41), (42, 43, 44), etc.; real networks are usually not so symmetric. In the directed network there are less symmetries: (4, 5, 6), (10, 11), and (17, 18). With respect to the relationship among the different centrality measures, the most remarkable similarity appears between shortest-path betweenness centrality and random-walk betweenness centrality, which is not surprising at all, but there are also important differences. For example, node 27 has zero betweenness in the undirected network since there is no shortest path crossing it, but it is among the ten nodes with highest random-walk betweenness since it lays in a region which communicates well-separated parts of the network. Another important feature in both networks is that nodes with high degree frequently appear as top ranked for many centralities, showing the relevance of degree in the analysis of complex networks.

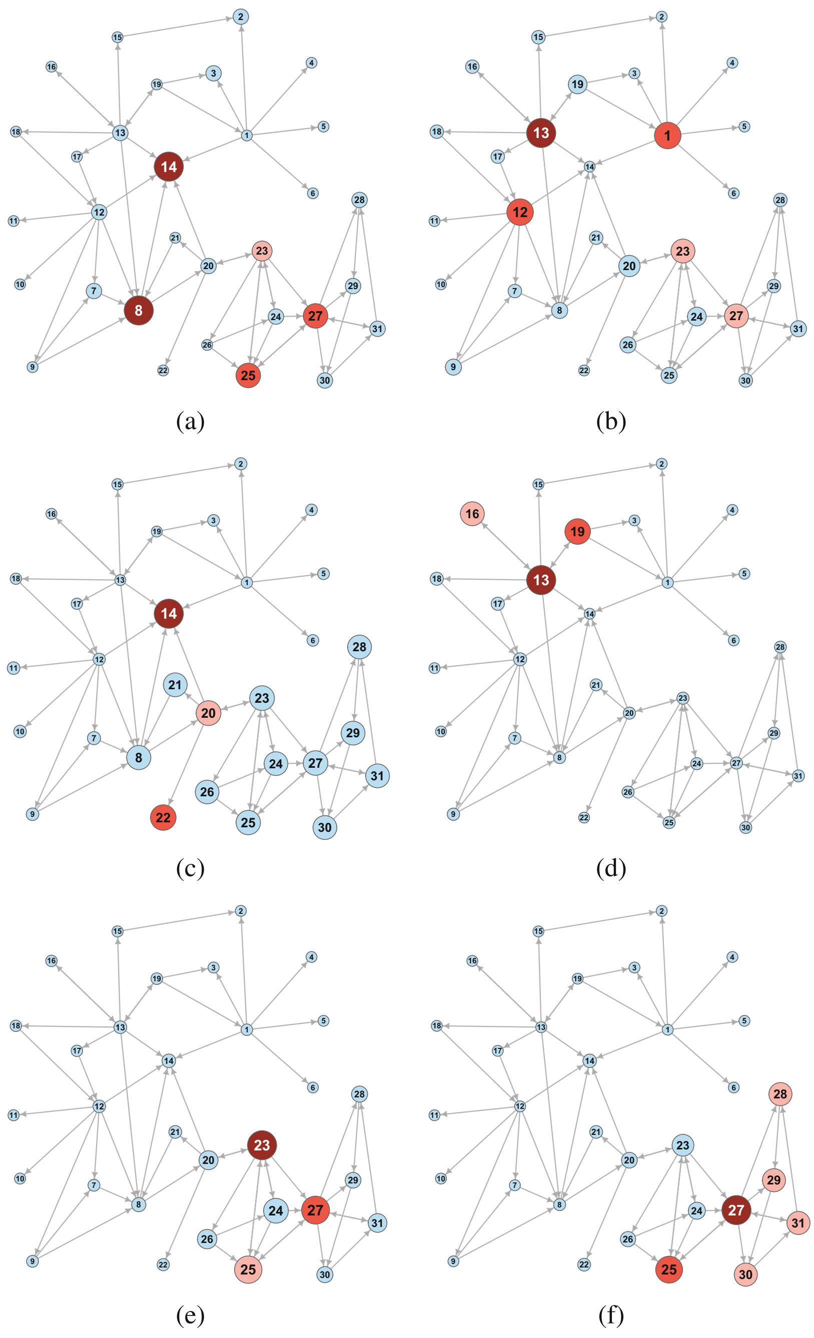


Fig. 8.3 Input and output degree, input and output closeness, eigenvector, and Katz centralities for a directed network. Nodes with highest centrality in dark red (and white node label), second largest centrality in red, third largest centrality in light red, and rest of nodes in blue. Sizes proportional to centrality with an offset. (**a**) Input degree centrality. (**b**) Output degree centrality. (**c**) Input closeness centrality. (**d**) Output closeness centrality. (**e**) Eigenvector centrality. (**f**) Katz centrality

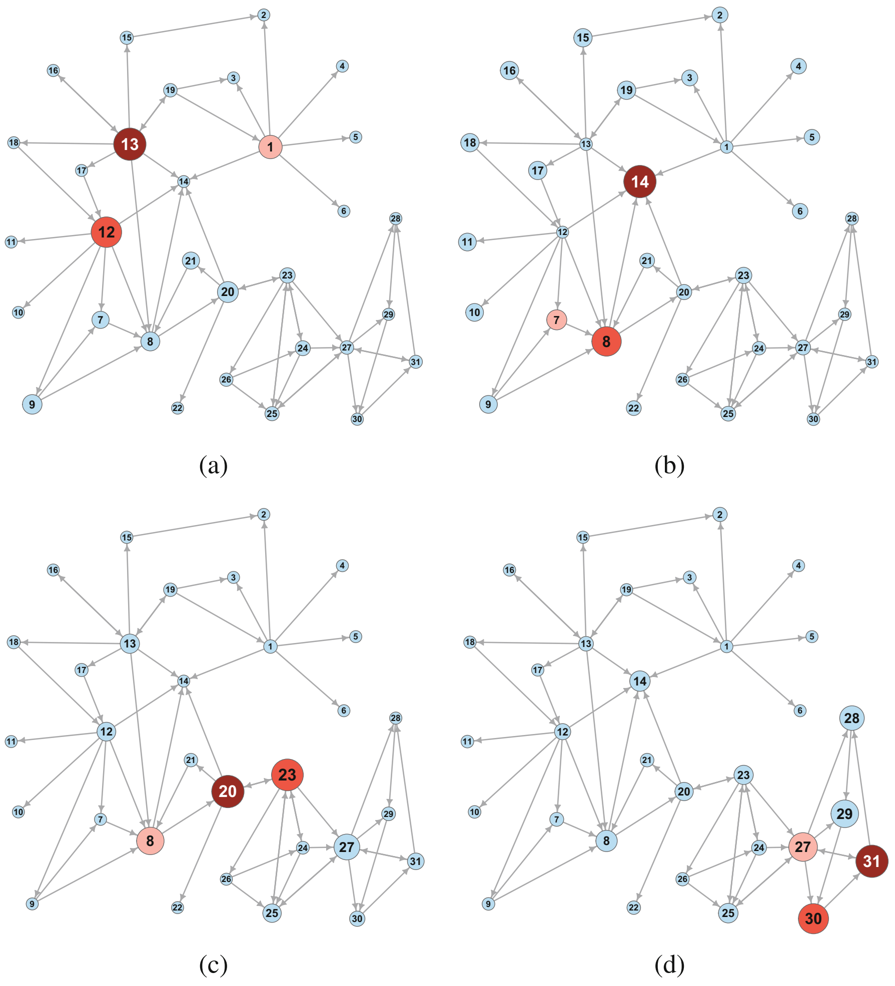


Fig. 8.4 Hub, authority, betweenness, and PageRank centralities for a directed network. Nodes with highest centrality in dark red (and white node label), second largest centrality in red, third largest centrality in light red, and rest of nodes in blue. Sizes proportional to centrality with an offset. (**a**) Hub centrality. (**b**) Authority centrality. (**c**) Betweenness centrality. (**d**) PageRank centrality

Although not directly related to the main topic of the book, we are going to analyse now a real network which is easy to recognize for a large audience, and whose results help in the understanding of the different definitions of centrality in networks. This is the Network of Thrones[2] [11], a network compiled from the third

[2]Network of Thrones: https://www.macalester.edu/~abeverid/thrones.html.

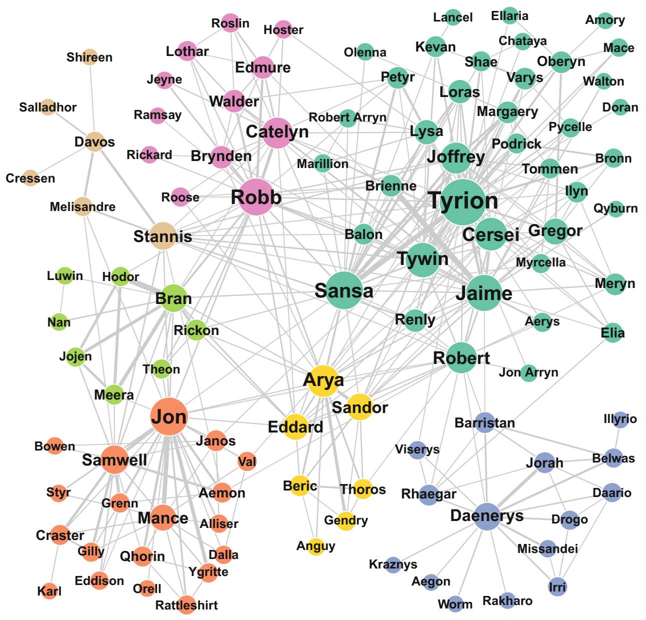
Table 8.3 Most central nodes of the Network of Thrones, using weighted centralities

Rank	Degree	Strength	Clos	Betw	Eigenv	Katz	PageRank	RWbetw	RWclos
1	Tyrion	Tyrion	Tyrion	Robb	Tyrion	Tyrion	Tyrion	Tyrion	Tyrion
2	Sansa	Jon	Sansa	Tyrion	Sansa	Sansa	Jon	Jon	Sansa
3	Jon	Sansa	Jaime	Sansa	Jaime	Jaime	Daenerys	Robb	Jaime
4	Robb	Jaime	Robb	Jon	Joffrey	Joffrey	Jaime	Jaime	Robb
5	Jaime	Bran	Tywin	Jaime	Cersei	Cersei	Sansa	Sansa	Tywin
6	Tywin	Robb	Cersei	Robert	Robb	Robb	Robb	Robert	Cersei
7	Cersei	Samwell	Brienne	Daenerys	Tywin	Bran	Bran	Daenerys	Robert
8	Arya	Arya	Joffrey	Stannis	Bran	Jon	Samwell	Bran	Arya
9	Catelyn	Joffrey	Catelyn	Samwell	Arya	Tywin	Arya	Stannis	Joffrey
10	Joffrey	Daenerys	Arya	Tywin	Brienne	Arya	Joffrey	Samwell	Jon
11	Robert	Cersei	Margaery	Arya	Catelyn	Hodor	Cersei	Arya	Catelyn
12	Samwell	Tywin	Bran	Bran	Margaery	Brienne	Tywin	Tywin	Stannis

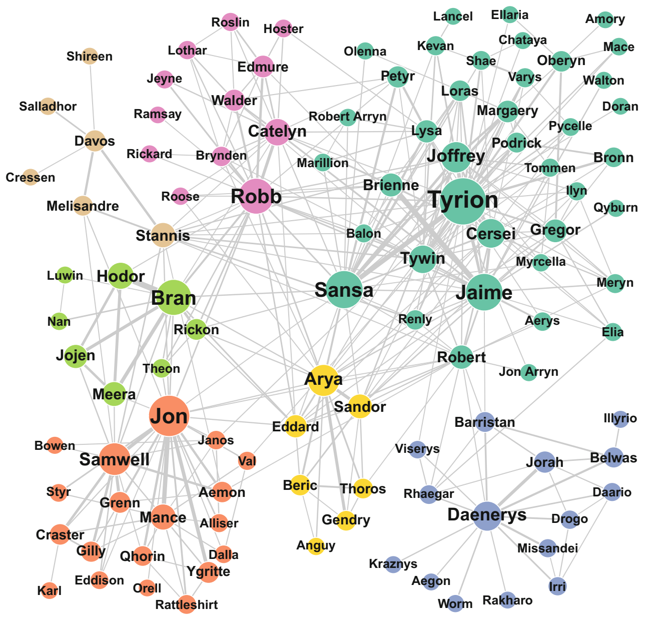
volume *"A Storm of Swords"* of the book series *"A Song of Ice and Fire"*, written by the novelist and screenwriter George R. R. Martin, and widely popularized by the HBO TV series "Game of Thrones", created by David Benioff and D. B. Weiss. The network contains the 107 characters of "A Storm of Swords" connected with 352 weighted edges. Two characters (nodes) are linked when their names are found in the book separated by at most 15 words, meaning they have interacted in some way. The weight counts the number of this kind of interactions.

Since the Network of Thrones is weighted, we have opted here to use the weighted versions of several centrality measures, namely strength, closeness, betweenness, eigenvector, Katz, PageRank, random-walk betweenness, and random-walk closeness. For the weighted closeness and betweenness, we have replaced the original weights w_{ij} by distances defined as $d_{ij} = 1/w_{ij}$, to take into account that the larger the weight, the smaller the distance (or dissimilarity) between the nodes. In Table 8.3 we show the 12 most central nodes for the degree centrality and the eight weighted centralities mentioned above. Unlike the previous synthetic networks, several characters are always among the most central nodes, with Tyrion Lannister on top of them, followed by Sansa Stark, Jaime Lannister, and Robb Stark, and to lower extend Jon Snow, Tywin Lannister and Cersei Lannister.

Figures 8.5, 8.6, 8.7, and 8.8 show the centralities of the Network of Thrones as proportional to the size of the nodes (and of the font of the names). The colours of the nodes correspond to the seven modules found using two different community detection approaches [21, 28], which produce exactly the same partition: modularity optimization [47] (using a combination of extremal optimization [26], tabu search [3], and fast algorithm [45]) and Infomap [53]. These communities are highly correlated with the different locations where the action takes place. A discussion in terms of *overlapping communities* for this network can be found in Chap. 9.

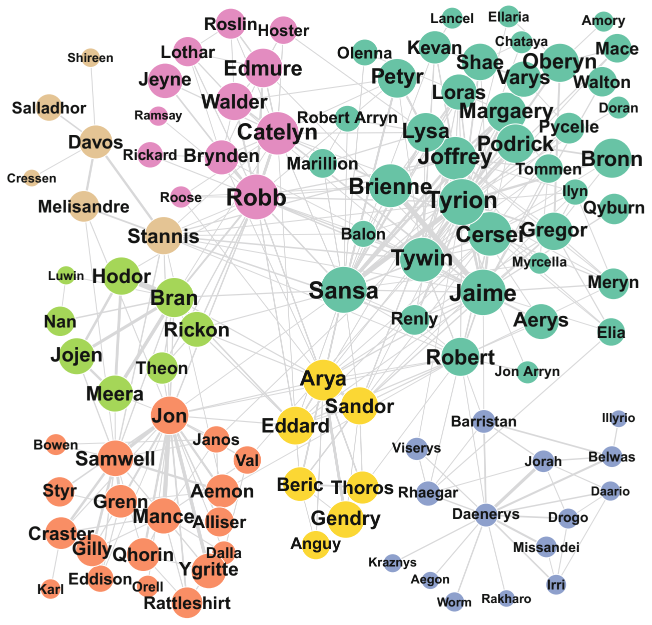


(a)

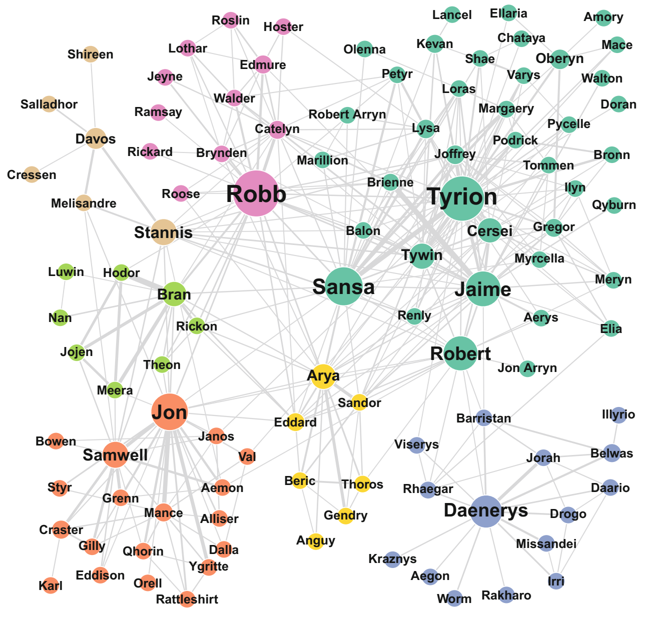


(b)

Fig. 8.5 Degree and strength centralities for the Network of Thrones. Nodes are coloured according to the modules found by community detection algorithms. Sizes of nodes proportional to centrality with an offset. Width of links proportional to weights. (**a**) Degree centrality. (**b**) Strength centrality

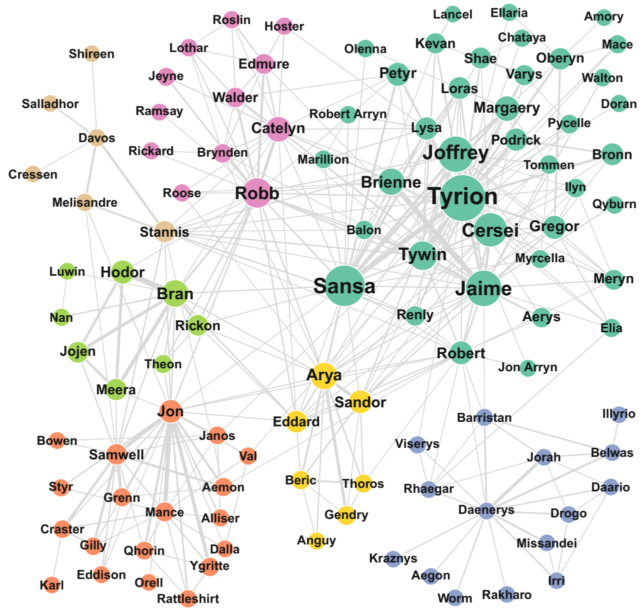


(a)

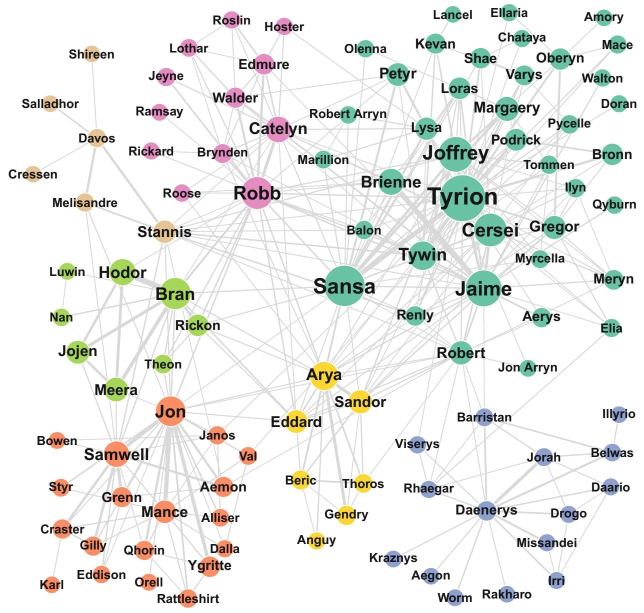


(b)

Fig. 8.6 Weighted closeness and weighted betweenness centralities for the Network of Thrones. Nodes are coloured according to the modules found by community detection algorithms. Sizes of nodes proportional to centrality with an offset. Width of links proportional to weights. (**a**) Weighted closeness centrality. (**b**) Weighted betweenness centrality



(a)



(b)

Fig. 8.7 Weighted eigenvector and weighted Katz centralities for the Network of Thrones. Nodes are coloured according to the modules found by community detection algorithms. Sizes of nodes proportional to centrality with an offset. Width of links proportional to weights. (**a**) Weighted eigenvector centrality. (**b**) Weighted Katz centrality

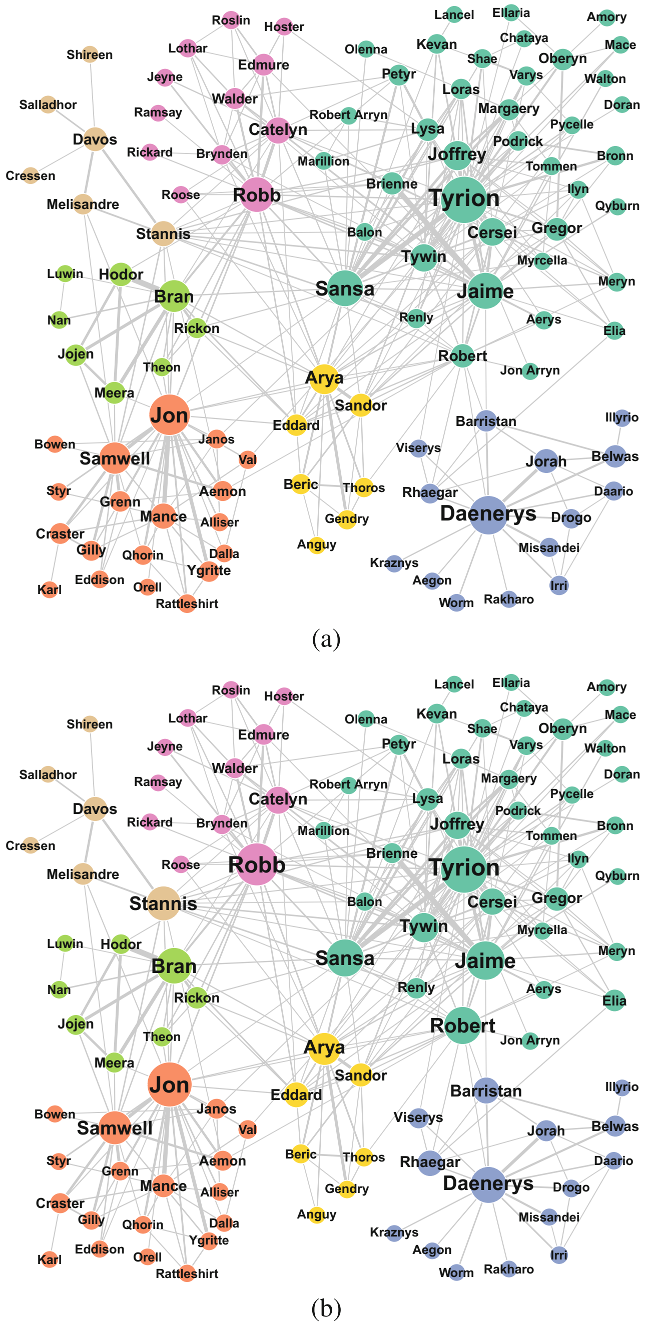


Fig. 8.8 Weighted PageRank and weighted random-walk betweenness centralities for the Network of Thrones. Nodes are coloured according to the modules found by community detection algorithms. Sizes of nodes proportional to centrality with an offset. Width of links proportional to weights. (**a**) Weighted PageRank centrality. (**b**) Weighted random-walk betweenness centrality

8.7 Software and Cost

Here comes a list of software tools which can be used to calculate centralities in complex networks:

- Pajek[3]: Analysis and visualization tool for Windows (can be run under Linux and MacOS using Wine) [7]. Allows the calculation of several centralities: degree, strength, closeness, betweenness, hubs and authorities (HITS), and a few additional ones not described above.
- Gephi[4]: Visualization and exploration software [6]. Calculates degree, strength, eigenvector, HITS, and PageRank centralities.
- Radatools[5]: Set of programs for the analysis of complex networks, with main attention to community detection and the finding of structural properties [33]. Calculates degree, strength, betweenness (weighted and unweighted, directed and undirected, for nodes and edges), and other centralities.
- Cytoscape[6]: Originally designed for biological research, now it is a general platform for complex network analysis and visualization [56]. It does not directly calculate centralities, but there are plug-ins which can be used to find some of them.
- igraph[7]: Collection of network analysis tools with the emphasis on efficiency, portability, and ease of use [20]. Calculates degree, strength, betweenness, closeness, eigenvector, HITS, and PageRank centralities.
- NetworkX[8]: Python software package for the creation, manipulation, and study of the structure, dynamics, and functions of complex networks [55]. Calculates degree, strength, closeness, betweenness, eigenvector, HITS, Katz and PageRank centralities, and a few additional ones.
- SNAP[9]: General purpose, high performance system for analysis, and manipulation of large networks [42]. Calculates degree, strength, closeness, betweenness, eigenvector, and HITS centralities.
- Visone[10]: Tool for the analysis and visualization of social networks [18]. Calculates degree, strength, closeness, betweenness, eigenvector, HITS and PageRank centralities, and a few additional ones.
- MuxViz[11]: Framework for the multilayer analysis and visualization of networks [22]. Calculates the generalizations of centralities to multilayer networks (versatilities), including degree, eigenvector, Katz, HITS, and PageRank centralities.

[3]Pajek: http://mrvar.fdv.uni-lj.si/pajek.

[4]Gephi: https://gephi.org.

[5]Radatools: http://deim.urv.cat/~sergio.gomez/radatools.php.

[6]Cytoscape: http://www.cytoscape.org.

[7]igraph: http://igraph.org.

[8]NetworkX: http://networkx.github.io.

[9]SNAP: http://snap.stanford.edu/snap.

[10]Visone: https://www.visone.info.

[11]MuxViz: http://muxviz.net.

Table 8.4 Computational cost of the calculation of centrality for different kinds of networks

Centrality	Unweighted general	Weighted general	Unweighted sparse	Weighted sparse
Degree	$O(L)$	$O(L)$	$O(N)$	$O(N)$
Closeness	$O((N+L)N)$	$O(NL+N^2\log N)$	$O(N^2)$	$O(N^2\log N)$
Betweenness	$O((N+L)N)$	$O(NL+N^2\log N)$	$O(N^2)$	$O(N^2\log N)$
Eigenvector	$O(Lr)$	$O(Lr)$	$O(Nr)$	$O(Nr)$
Katz	$O(Lr)$	$O(Lr)$	$O(Nr)$	$O(Nr)$
Hub and authority	$O(Lr)$	$O(Lr)$	$O(Nr)$	$O(Nr)$
PageRank	$O(Lr)$	$O(Lr)$	$O(Nr)$	$O(Nr)$
RW betweenness	$O((N+L)N^2)$	$O((N+L)N^2)$	$O(N^3)$	$O(N^3)$
RW closeness	$O(N^3)$	$O(N^3)$	$O(N^3)$	$O(N^3)$

- graph-tool[12]: Efficient Python module for manipulation and statistical analysis of graphs [50]. Calculates PageRank, betweenness, closeness, eigenvector, Katz, HITS, and other centralities.

The integration of some tools with Python (igraph, NetworkX, graph-tool) and R (igraph, MuxViz) allows a high-level implementation of the missing centralities without too much effort.

We summarize the computational cost of the computation of the centralities in Table 8.4. They have been described in their respective sections, so we just remember now that N stands for the number of nodes, L for the number of edges, and r for the number of iterations until convergence, which depends on the particular network and is impossible to estimate beforehand.

8.8 Conclusions

We have seen how it is possible to find the most important items in a dataset, provided we transform this data into a complex network. The definition of "most important" is not unique, several complementary ways exist, each one concentrated in one structural characteristic of the network. Degree centrality allows to find the most connected nodes. Closeness centrality finds the nodes which are in the "middle" of the network, i.e., at a shortest average distance to the rest of the nodes. Betweenness centrality is specialized in the nodes which are "bridges" between separated parts of the network. Eigenvector centrality looks for nodes whose importance is given by the sum of the centralities of the nodes which send links to it, thus becoming a recursive definition which is expressed as an eigenvector

[12]graph-tool: https://graph-tool.skewed.de.

and eigenvalue problem. Katz centrality represents a balance between closeness and eigenvector centralities. Finally, the dynamics of random walkers in the network is the basis for several centralities, standing out PageRank, the well-known measure originally used to rank web pages by the Google search engine. We have also considered how centralities must be adapted for the different kinds of network, e.g., by taking into account the directionality of the links, their weights, or the multilayer structure. In summary, centrality integrates a large set of definitions and tools to analyse the relevance of the nodes in networks, being able to identify the most important ones, which may constitute the first step in many marketing and business applications, where targeted actions increase their success rate and reduce the overall cost.

Acknowledgements S.G. acknowledges funding from the Spanish Ministerio de Economía y Competitividad (grant number FIS2015-71582-C2-1).

References

1. Ahnert, S., Garlaschelli, D., Fink, T., Caldarelli, G.: Ensemble approach to the analysis of weighted networks. Physical Review E **76**(1), 016,101 (2007)
2. Anthonisse, J.M.: The rush in a directed graph. Stichting Mathematisch Centrum. Mathematische Besliskunde (BN 9/71), 1–10 (1971)
3. Arenas, A., Fernandez, A., Gomez, S.: Analysis of the structure of complex networks at different resolution levels. New Journal of Physics **10**(5), 053,039 (2008)
4. Barabási, A.L., Albert, R.: Emergence of scaling in random networks. Science **286**(5439), 509–512 (1999)
5. Barrat, A., Barthelemy, M., Pastor-Satorras, R., Vespignani, A.: The architecture of complex weighted networks. Proceedings of the National Academy of Sciences USA **101**(11), 3747–3752 (2004)
6. Bastian, M., Heymann, S., Jacomy, M., et al.: Gephi: an open source software for exploring and manipulating networks. ICWSM **8**, 361–362 (2009)
7. Batagelj, V., Mrvar, A.: Pajek – program for large network analysis. Connections **21**(2), 47–57 (1998)
8. Battiston, F., Nicosia, V., Latora, V.: Structural measures for multiplex networks. Physical Review E **89**(3), 032,804 (2014)
9. Bavelas, A.: Communication patterns in task-oriented groups. The Journal of the Acoustical Society of America **22**(6), 725–730 (1950)
10. Beauchamp, M.A.: An improved index of centrality. Behavioral Science **10**(2), 161–163 (1965)
11. Beveridge, A., Shan, J.: Network of thrones. Math Horizons **23**(4), 18–22 (2016)
12. Boccaletti, S., Bianconi, G., Criado, R., Del Genio, C.I., Gómez-Gardenes, J., Romance, M., Sendina-Nadal, I., Wang, Z., Zanin, M.: The structure and dynamics of multilayer networks. Physics Reports **544**(1), 1–122 (2014)
13. Bonacich, P.: Factoring and weighting approaches to status scores and clique identification. Journal of Mathematical Sociology **2**(1), 113–120 (1972)
14. Bonacich, P.: Technique for analyzing overlapping memberships. Sociological Methodology **4**, 176–185 (1972)
15. Bonacich, P.: Power and centrality: A family of measures. American Journal of Sociology **92**(5), 1170–1182 (1987)

16. Brandes, U.: A faster algorithm for betweenness centrality. Journal of Mathematical Sociology **25**(2), 163–177 (2001)
17. Brandes, U., Erlebach, T.: Network Analysis: Methodological Foundations, *Lecture Notes in Computer Science*, vol. 3418. Springer (2005)
18. Brandes, U., Wagner, D.: Analysis and visualization of social networks. Graph drawing software pp. 321–340 (2004)
19. Brin, S., Page, L.: The anatomy of a large-scale hypertextual web search engine. Computer Networks and ISDN Systems **30**(1), 107–117 (1998)
20. Csardi, G., Nepusz, T.: The igraph software package for complex network research. InterJournal, Complex Systems **1695**(5), 1–9 (2006)
21. Danon, L., Diaz-Guilera, A., Duch, J., Arenas, A.: Comparing community structure identification. Journal of Statistical Mechanics: Theory and Experiment **2005**(09), P09,008 (2005)
22. De Domenico, M., Porter, M.A., Arenas, A.: Muxviz: a tool for multilayer analysis and visualization of networks. Journal of Complex Networks **3**(2), 159 (2015).
23. De Domenico, M., Solé-Ribalta, A., Cozzo, E., Kivelä, M., Moreno, Y., Porter, M.A., Gómez, S., Arenas, A.: Mathematical formulation of multilayer networks. Physical Review X **3**(4), 041,022 (2013)
24. De Domenico, M., Solé-Ribalta, A., Omodei, E., Gómez, S., Arenas, A.: Ranking in interconnected multilayer networks reveals versatile nodes. Nature Communications **6** (2015)
25. Dekker, A.: Conceptual distance in social network analysis. Journal of Social Structure (JOSS) **6** (2005)
26. Duch, J., Arenas, A.: Community detection in complex networks using extremal optimization. Physical review E **72**(2), 027,104 (2005)
27. Floyd, R.W.: Algorithm 97: shortest path. Communications of the ACM **5**(6), 345 (1962)
28. Fortunato, S.: Community detection in graphs. Physics reports **486**(3), 75–174 (2010)
29. Freeman, L.C.: A set of measures of centrality based on betweenness. Sociometry pp. 35–41 (1977)
30. Freeman, L.C.: Centrality in social networks conceptual clarification. Social Networks **1**(3), 215–239 (1979)
31. Frobenius, G.: Über matrizen aus nicht negativen elementen. Sitzungsber. Königl. Preuss. Akad. Wiss. pp. 456–477 (1912)
32. Garrison, W.L.: Connectivity of the interstate highway system. Papers and Proceedings of the Regional Science Association **6**, 121–137 (1960)
33. Gómez, S., Fernández, A.: Radatools software, communities detection in complex networks and other tools (2011)
34. Guimerà, R., Diaz-Guilera, A., Vega-Redondo, F., Cabrales, A., Arenas, A.: Optimal network topologies for local search with congestion. Physical Review Letters **89**(24), 248,701 (2002)
35. Halu, A., Mondragón, R.J., Panzarasa, P., Bianconi, G.: Multiplex pagerank. PLOS ONE **8**(10), e78,293 (2013)
36. Jacob, R., Koschützki, D., Lehmann, K., Peeters, L., Tenfelde-Podehl, D.: Algorithms for centrality indices. Network Analysis pp. 62–82 (2005)
37. Katz, L.: A new status index derived from sociometric analysis. Psychometrika **18**(1), 39–43 (1953)
38. Kivelä, M., Arenas, A., Barthelemy, M., Gleeson, J.P., Moreno, Y., Porter, M.A.: Multilayer networks. Journal of Complex Networks **2**(3), 203–271 (2014)
39. Kleinberg, J.M.: Authoritative sources in a hyperlinked environment. Journal of the ACM (JACM) **46**(5), 604–632 (1999)
40. Koschützki, D., Lehmann, K., Peeters, L., Richter, S., Tenfelde-Podehl, D., Zlotowski, O.: Centrality indices. Network Analysis pp. 16–61 (2005)
41. Koschützki, D., Lehmann, K., Tenfelde-Podehl, D., Zlotowski, O.: Advanced centrality concepts. Network Analysis pp. 83–111 (2005)
42. Leskovec, J., Sosič, R.: Snap: Stanford network analysis platform (2013)
43. Lovász, L.: Random walks on graphs: A survey. Combinatorics, Paul Erdos is Eighty **2**(1), 1–46 (1993)

44. Newman, M.: Networks: An Introduction. Oxford University Press, Inc., New York, NY, USA (2010)
45. Newman, M.E.: Fast algorithm for detecting community structure in networks. Physical review E **69**(6), 066,133 (2004)
46. Newman, M.E.: A measure of betweenness centrality based on random walks. Social Networks **27**(1), 39–54 (2005)
47. Newman, M.E., Girvan, M.: Finding and evaluating community structure in networks. Physical review E **69**(2), 026,113 (2004)
48. Nieminen, J.: On the centrality in a directed graph. Social Science Research **2**, 371–378 (1973)
49. Noh, J.D., Rieger, H.: Random walks on complex networks. Physical Review Letters. **92**(11), 118,701 (2004)
50. Peixoto, T.P.: The graph-tool python library. figshare (2014)
51. Perron, O.: Zur theorie der matrices. Mathematische Annalen **64**(2), 248–263 (1907)
52. Pitts, F.R.: A graph theoretic approach to historical geography. The Professional Geographer **17**, 15–20 (1965)
53. Rosvall, M., Bergstrom, C.T.: Maps of random walks on complex networks reveal community structure. Proceedings of the National Academy of Sciences **105**(4), 1118–1123 (2008)
54. Sabidussi, G.: The centrality index of a graph. Psychometrika **31**, 581–603 (1966)
55. Schult, D.A., Swart, P.: Exploring network structure, dynamics, and function using networkx. In: Proceedings of the 7th Python in Science Conferences (SciPy 2008), vol. 2008, pp. 11–16 (2008)
56. Shannon, P., Markiel, A., Ozier, O., Baliga, N.S., Wang, J.T., Ramage, D., Amin, N., Schwikowski, B., Ideker, T.: Cytoscape: a software environment for integrated models of biomolecular interaction networks. Genome research **13**(11), 2498–2504 (2003)
57. Shaw, M.E.: Group structure and the behavior of individuals in small groups. Journal of Psychology **38**, 139–149 (1954)
58. Solá, L., Romance, M., Criado, R., Flores, J., del Amo, A.G., Boccaletti, S.: Eigenvector centrality of nodes in multiplex networks. Chaos **3**, 033,131 (2013)
59. Solé-Ribalta, A., De Domenico, M., Gómez, S., Arenas, A.: Random walk centrality in interconnected multilayer networks. Physica D: Nonlinear Phenomena **323**, 73–79 (2016)
60. Solé-Ribalta, A., Gómez, S., Arenas, A.: A model to identify urban traffic congestion hotspots in complex networks. Royal Society Open Science **3**(10), 160,098 (2016)
61. Yang, S.J.: Exploring complex networks by walking on them. Physical Review E **71**(1), 016,107 (2005)
62. Zhang, Z., Julaiti, A., Hou, B., Zhang, H., Chen, G.: Mean first-passage time for random walks on undirected networks. The European Physical Journal B-Condensed Matter and Complex Systems **84**(4), 691–697 (2011)

Chapter 9
Overlapping Communities in Co-purchasing and Social Interaction Graphs: A Memetic Approach

Ademir Gabardo, Regina Berretta, and Pablo Moscato

Abstract Simple undirected graphs can be employed to represent numerous complex systems including those arising in social networks, biological networks, communication networks, transportation routes and several others. An important feature of complex networks is the presence of communities, groups of elements densely connected among them but sparsely linked to the rest of the network. In many cases these communities can be overlapping, with nodes participating in more than one community. In this chapter, we present a memetic algorithm for overlapping community detection. Our approach uses the communities of links to depict the overlapping community structure in a simple undirected graph. The approach uses the line of the original graph interest. We perform modularity optimization to discover the communities of the vertices of the line graph to unveil the overlapping community structure of the network. To assess the quality of our method, we present results in synthetically generated benchmark networks and to exemplify the usefulness of our approach we present two case studies. In the first case study we use a network of characters of the novel "A Storm of Swords" book series "A Song of Ice and Fire", written by George R. R. Martin; and a second one using a co-purchasing network of luxury items from a brand-centric point of view.

Keywords Community detection · Eigenvector centrality · Overlapping community detection · Memetic algorithm

A. Gabardo · R. Berretta (✉) · P. Moscato
School of Electrical Engineering and Computing, The University of Newcastle, Callaghan, NSW, Australia
e-mail: ademir.gabardo@uon.edu.au; regina.berretta@newcastle.edu.au; Pablo.Moscato@newcastle.edu.au

P. Moscato, N. J. de Vries (eds.), *Business and Consumer Analytics: New Ideas*,
https://doi.org/10.1007/978-3-030-06222-4_9

9.1 Introduction

Networks are collections of interconnected items, frequently represented by means of a graph, a mathematical abstraction formed by nodes joined by edges. Networks are suitable for representing relationships in several complex systems. Simple undirected graphs have a wide range of applications as they appear in the study of social networks, biological networks, gene regulatory networks, transportation, communication and several other applications pervading many scientific fields [9, 41].

Complex networks differ in a number of properties. They may have a highly heterogeneous number of links connecting nodes, or highly connected nodes (hubs), and they may widely vary in the presence of the so-called communities. These are subsets of the nodes which are densely connected and sparsely linked to the rest of the network/graph. Moreover, these groups of nodes frequently show a high degree of similarity.

Detecting communities in complex networks is fundamental to understand complex systems. In many networks it is possible, thanks to mathematical models and algorithms, to identify communities having diverse hierarchical patterns, distinct cluster sizes and distinct resolution levels.

In real-world networks the communities can often be *overlapping* [1, 4, 7, 9, 24, 43, 47]. From a modelling perspective, and for some problem domains, this is actually the most "natural" approach to address some important issue under investigation. Overlapping communities are basically groups of nodes that are highly connected but that they can share a fraction of its nodes with other communities. Therefore, in this case individual nodes may belong to two or more communities.

An intricate level of overlapping complexity and nested communities are often found in real-world complex systems [1, 7, 43], such as social network members participating in distinct social groups [1, 24], proteins participating in different groups of biological functions [1, 4, 47] and researchers publishing in distinct scientific fields [7]. Figure 9.1 shows an example of a complex network with three levels of overlapping, with disjoint communities, partial overlap and the complete hierarchical overlapping.

Recent surveys about community detection methods emphasize that the majority of community detection algorithms are developed to detect non-overlapping communities [2, 54]. Furthermore, methods derived from non-overlapping community detection may present drawbacks when they are somehow adapted to work for an overlapping community detection problem.

As the reader may expect, the problems associated with community detection may vary, but detecting overlapping communities in complex networks is, in some cases, an NP-hard computational problem [9]; therefore, efficient algorithms and metaheuristics must be employed to tackle this problem for large scale instances.

This chapter starts describing in detail a memetic algorithm designed specifically for detecting overlapping communities (called "MADOC"), which was briefly introduced in [11]. Several computational experiment results are presented. Initially, to evaluate the performance of MADOC, we employ a group of synthetically generated networks. Next, two real world networks from different areas are used to illustrate the usefulness of our approach. A network of characters of the novel "A Storm of Swords" book series "A Song of Ice and Fire", written by George R. R. Martin; and a co-purchasing network of luxury items from a brand-centric point of view.

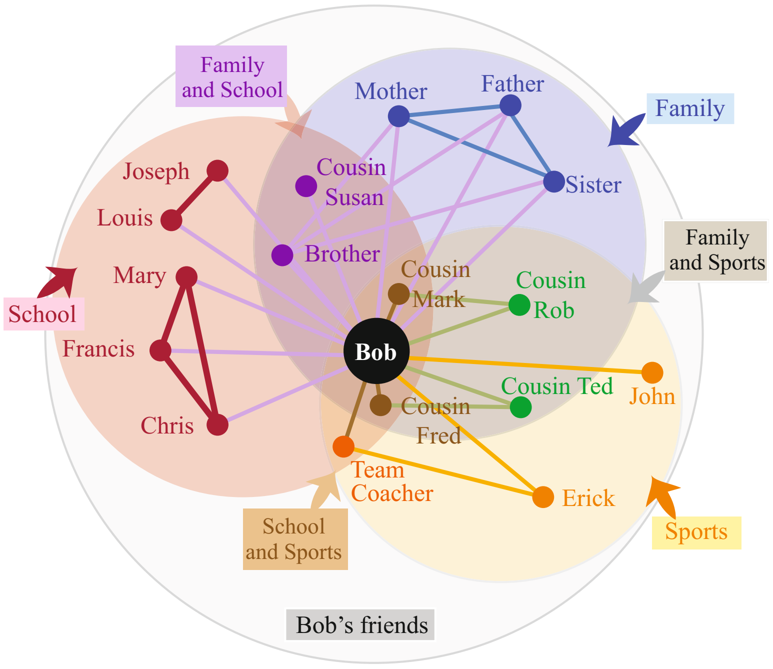


Fig. 9.1 The social network of the fictional character Bob with three overlapping communities in Bob's friends network as an example of hierarchical nested overlapping communities

9.2 Approaches to Detect Overlapping Communities

The goal of community detection algorithms is to identify a meaningful partition of the network nodes. Overlapping communities are communities that share a fraction of its membership, where given two sets of nodes A and B, the intersection $A \cap B \neq \emptyset$. Therefore, the goal of overlapping community detection algorithms is to give a non-empty set of labels to each node, i.e. to identify the set of communities that each node belongs.

Currently popular approaches to tackle the overlapping community detection problem are clique percolation, label propagation, node seed expansion and link communities. This section briefly introduces these approaches including some strengths and drawbacks to provide a basic introduction to existing methods.

9.2.1 Clique Percolation

Introduced by Palla et al. [43], the clique percolation method builds overlapping communities by searching for adjacent cliques. Cliques are fully connected subgraphs, in which every pair of nodes in the subgraph is connected. The algorithm merges cliques into a community if they share $(k - 1)$ nodes, where k is the size of the cliques. Figure 9.2 shows an example of two cliques of size four merged to form a community.

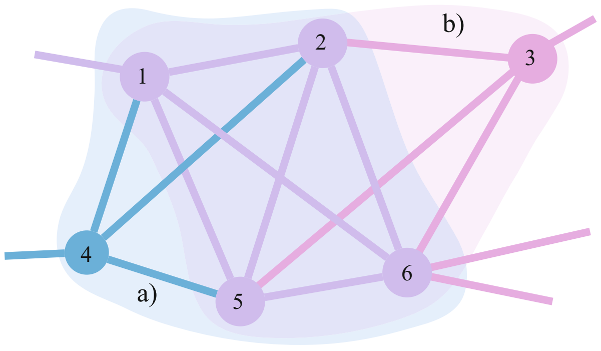


Fig. 9.2 An example of two overlapping communities highlighted by distinct colours. In (**a**) two cliques of size four [1, 2, 4, 5] and [1, 2, 5, 6] have three nodes [1, 2, 5] in common; therefore, these two cliques are merged to form a community. Similarly, in (**b**) two cliques of size four [1, 2, 5, 6] and [2, 3, 5, 6] with three nodes [2, 5, 6] are joined to form a community. Communities (**a**) and (**b**) have nodes [1, 2, 5, 6] in common, these are the overlapping nodes

Despite its popularity, clique percolation methods have the drawback of presenting inconsistent results when applied to a network absent of large cliques. In addition, finding cliques in large networks is computationally expensive. An example of implementation of the clique percolation approach is the algorithm CFinder[1] [43]. Recently published works using cliques to detect overlapping communities are the maximal clique network label propagation algorithm (MCNLPA) [51], Greedy clique expansion (GCE) [44] and optimization over maximal cliques (OMC) [21].

[1]http://www.cfinder.org/.

9.2.2 Label Propagation

Some methods attribute labels to nodes based on the labels of its neighbours. In the first interaction, each node receives a distinct label. Therefore, the number of communities is equal to the number of nodes in the graph. At each iteration, each node assumes the label used by most of its neighbours. If more than one label has the same maximum number of neighbours, one is chosen uniformly at randomly. As the process evolves, the same label tends to become associated with all nodes of a community. Figure 9.3 shows a network with ten nodes organized into three communities labelled A, B and C at iteration n. In the next iteration ($n + 1$), nodes will receive the most frequent label among their neighbours. Therefore, the node in community B will receive the label A, because it shares three edges with other nodes in community A, and a single edge with the nodes in community C. The algorithm stops when there are no more changes in labels.

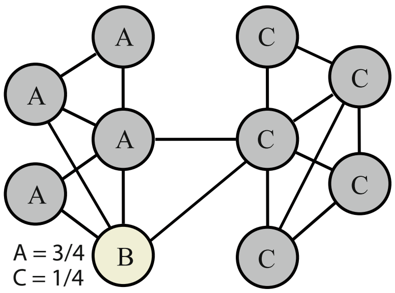


Fig. 9.3 A network with ten nodes organized into three communities identified by the labels A, B and C. In iteration n, the single node in community B connects to three nodes in community A and one node in community C. Hence, after the label update (in iteration $n + 1$) the node in community B will receive the label A

Advantages of this method are the high time efficiency of the algorithm and the ease of implementation. A drawback is the inconsistency in results across distinct runs caused by the randomness nature of the algorithm.

During the past year we have seen an increased interest in this type of approach. Gregory [17] extended the creation by Raghavan et al. [48] (later named RAK due to its authors' names) allowing nodes to participate in multiple communities, therefore, supporting overlapping communities. In the model proposed by Gregory, each node receives a belonging coefficient b for each community C, such that the sum of all belonging coefficients for the node is equal to 1. Other examples of algorithms using label propagation to tackle the overlapping community detection problem are the label propagation algorithm (LPA) [48], the community overlap propagation algorithm (COPRA)[17], the community formation game [5], the balanced multi-label propagation algorithm (BMLPA) [53] and the strength driven label propagation algorithm [55].

9.2.3 *Node Seed Expansion*

In node seed expansion methods, communities are built around a node (seed), or around the *natural community* of a node, also referred as *egonet*. The natural community of a node comprises the node, its neighbours and the edges connecting these nodes. Figure 9.4 shows a node and its natural community.

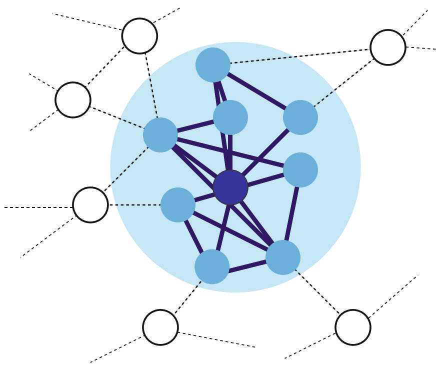


Fig. 9.4 A small network with a node and its natural community highlighted in the centre

Lancichinetti et al. [24] proposed a *fitness* to compute the ratio between the number of edges with both endpoints inside the community (internal edges) and the number of edges with only one endpoint inside the community (external edges) to evaluate the quality of communities. The method evolves by randomly adding nodes to a community if the community fitness is improved by the addition of the node, and pruning nodes that deteriorate the community fitness. The process is repeated until no positive moves can be made.

Examples of algorithms based on egonets are the algorithm of Lancichinetti et al. [24], the model-based overlapping seed expansion (MOSES) [35], the order statistics local optimization method (OSLOM) [26] and the egomunities algorithm [10].

9.2.4 *Link Communities*

Link communities methods use the similarity of the edges in order to detect overlapping communities in a network. Link communities methods also known as *link clustering* or *edge clustering* can employ hierarchical clustering [1] or line graphs [8] to identify the communities formed by the edges of a complex network. A line graph is a representation $L(G)$ that corresponds to the adjacency between the edges of a graph $G = (V, E)$, in such way that each edge in G corresponds to a node in $L(G)$, and two nodes in $L(G)$ are connected if they share a node.

Since nodes are adjacent to several edges, communities built using line graphs are naturally overlapping, since the edges adjacent to a node can be attributed to diverse communities and consequently nodes are assigned to multiple communities.

Figure 9.5 shows a graph G, its corresponding line graph $L(G)$ divided into two non-overlapping communities and the graph G with the overlapping communities.

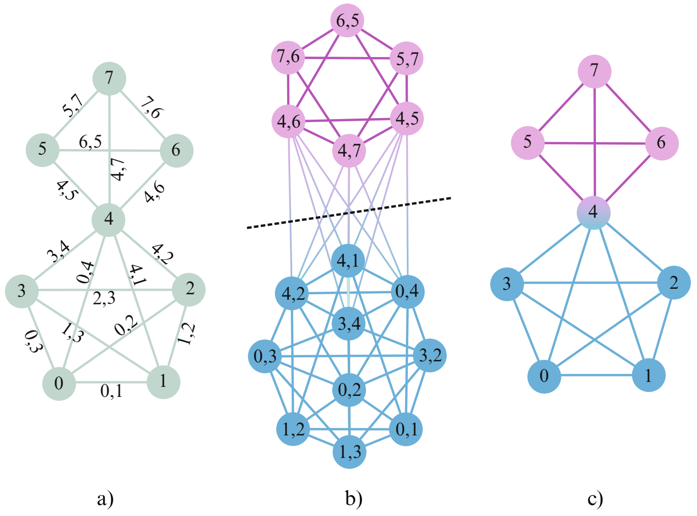


Fig. 9.5 (**a**) A graph $G = (V, E)$, (**b**) the line graph $L(G)$ of G divided into two non-overlapping communities and (**c**) the overlapping communities recovered from $L(G)$

In Fig. 9.5a, a small network with 8 nodes and 16 edges is shown, in Fig. 9.5b each edge in Fig. 9.5a is turned into a node, for instance, the edge connecting nodes 4 and 5 in Fig. 9.5a becomes the node (4,5) in the line graph, consequently, the line graph in Fig. 9.5b has 16 nodes (corresponding to the 16 edges from Fig. 9.5a). In Fig. 9.5b, an edge connects two nodes if they share a source node, for instance, nodes (4,5) and (4,6) are joined because both share the connection with node 4 in the original graph. The dashed line in Fig. 9.5b represents the non-overlapping partition of the line graph into two communities. In Fig. 9.5c the communities are recovered from the line graph to the original graph, node 4 appears in both communities in Fig. 9.5b; therefore, it is the overlapping node in this network. We can easily notice that in larger networks many nodes will have similar connections to the one presented by node 4 in Fig. 9.5, consequently, leading to numerous overlapping communities.

A drawback of these approaches is the increased dimension of the search space. Since networks usually have more edges than nodes, the line graph is larger than the original graph, demanding more computational resources to detect the overlapping communities. Nevertheless, there is a strong interest in the approach. Examples of methods for overlapping community detection using line graphs include: the link partition method of Evans and Lambiotte [8], genetic algorithm (GANET+) [47], the link communities method by Ahn et al. [1], the genetic algorithm for overlapping community detection (GaoCD) [50], the node-or-link communities method (NLC) [19] and the link-based memetic algorithm [18].

9.3 Memetic Algorithm to Detect Overlapping Communities (MADOC)

Memetic algorithms (MAs) are a population-based search approach in which a set of computational agents compete and cooperate to find solutions to a problem generally posed as an optimization one. MAs employ periods in which the agents used feasible (and sometimes even unfeasible) solutions of the problem are refined by algorithms and heuristics that do not need to communicate their doings with other agents or algorithmic procedures. An initial "population" is improved (according to some ad hoc criteria which depend on the problem). The feasible and unfeasible solutions are subsequently "recombined" (with other algorithms) and these processes are reiterated until they reach a termination criterion. A large variety of this basic metaheuristic template usually exists. The incorporation of problem domain knowledge is then a fundamental feature that characterizes MAs, and it has proved to be a practical solution for NP-hard optimization problems such as the overlapping community detection problem as well as many others [37, 39, 40]. We refer to the section dedicated to MAs (Part IV) in this collection for a survey in recent applications in business and consumer analytics.

Understandably, Memetic algorithms have been employed to detect disjoint partitions in complex networks [12, 15, 16, 27, 39, 52]; however, to date, aside the algorithm presented initially in [11] and described in this chapter, the only other MA for detecting overlapping communities is the algorithm proposed by Havemann et al. [18], a memetic algorithm following the work of Evans and Lambiotte [8] that combines probabilistic evolutionary approaches with deterministic local search to find communities over line graphs.

Our memetic algorithm to detect overlapping communities (MADOC), initially introduced in [11], performs modularity maximization to find the communities in a line graph, later recovering the information of the link communities to the nodes in the graph. Since nodes are adjacent to several edges, the communities are naturally overlapping. Our algorithm differs from Havemann's algorithm [18] in three major aspects: the initialization uses the 'natural communities of a graph in order to reduce the number of generations required to evolve the population and converge to a solution; the cost function used to evaluate the fitness of each individual is Girvan and Newman's modularity Q and the local search follows a strategy inspired in label propagation methods. Figure 9.6 provides a general outline of the operations performed by MADOC.

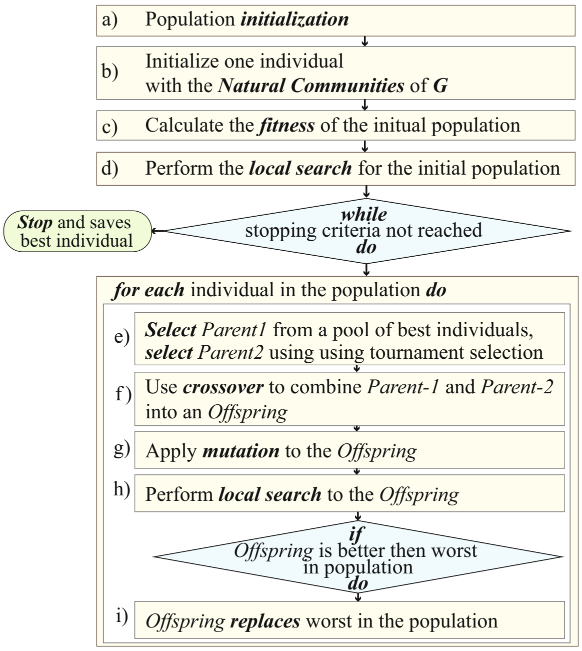

a) Population *initialization*
b) Initialize one individual with the *Natural Communities* of *G*
c) Calculate the *fitness* of the initual population
d) Perform the *local search* for the initial population
while stopping criteria not reached *do*
Stop and saves best individual
for each individual in the population *do*
e) *Select* *Parent1* from a pool of best individuals, *select* *Parent2* using using tournament selection
f) Use *crossover* to combine *Parent-1* and *Parent-2* into an *Offspring*
g) Apply *mutation* to the *Offspring*
h) Perform *local search* to the *Offspring*
if *Offspring* is better then worst in population *do*
i) *Offspring* *replaces* worst in the population

Fig. 9.6 The main scheme of processes performed by MADOC. Steps *a* to *d* show the initialization of the population and steps *e* to *i* show the evolutionary mechanism

Next sections detail the encoding used for representing an individual, the population and initialization criteria, mechanisms used for selection, recombination (sometimes called crossover in the literature) and mutation operators, fitness function and the local search used by our algorithm. The local search method provides the individual improvement phase of the memetic algorithm and the fitness function is part of the criteria used to guide the population to search the most promising regions of the configuration space when aiming to maximize the modularity of the partition of the set of nodes.

9.3.1 Individual Representation

A fundamental step of problem solving using population-based approaches and evolutionary computation techniques is representing solutions of the problem in a configuration space [38]. Over the years, the terms genotype, chromosome or individual are sometimes employed to describe a candidate solution encoding a phenotype, that is, something that encodes for one feasible solution for a specific problem [37, 38, 40]. The selection of a proper encoding is perhaps one of the most important elements and, generally, cannot be done without a proper consideration of the other algorithmic components of an MA [38].

A popular encoding approach used for community detection problem is the *string-coding* representation [39]. Given a graph $G = (V, E)$, where V is a set of nodes and E is a set of edges, an individual Ind can be encoded as a string of n "genes", with $n = |V|$, where Ind_i corresponds to the community of node i. Figure 9.7 shows an example of an individual used to represent a graph of eleven nodes divided into two communities.

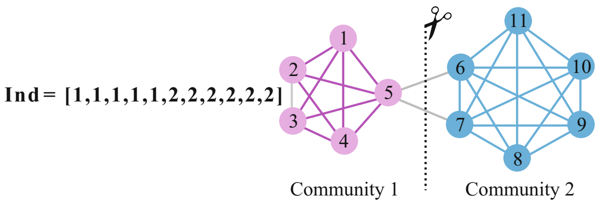


Fig. 9.7 The string-coding representation. In this example a network of eleven nodes is partitioned into two communities. Each position of the string (Ind) contains the community which the node belongs

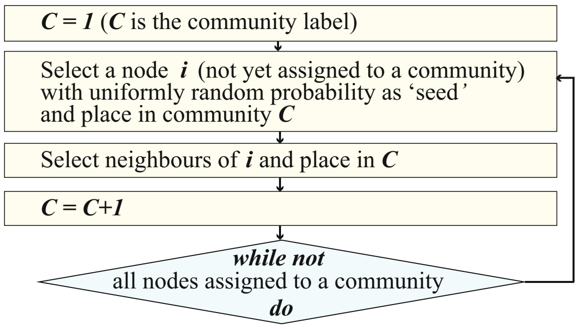
In this representation, the problem of community detection is translated to an optimization combinatorial problem, where the objective is to attribute nodes to communities in such a way to achieve the best partition for a given graph.

9.3.2 *Population and Initialization*

The basic principle of evolutionary algorithms is to evolve a population of individuals with prospective solutions for a problem. This process encompasses the evolution of several "generations" in which the best-fit individuals have a higher probability to pass its characteristics to the populations present in future generations. In order to be a useful search procedure, such a population must be diverse enough to explore the configuration space and mechanisms should unsure that there is no premature convergence (i.e. by avoiding diversity loss). Likewise, if possible, building an initial population of individuals with good characteristics is deemed necessary for obtaining a diverse and good start to the search process.

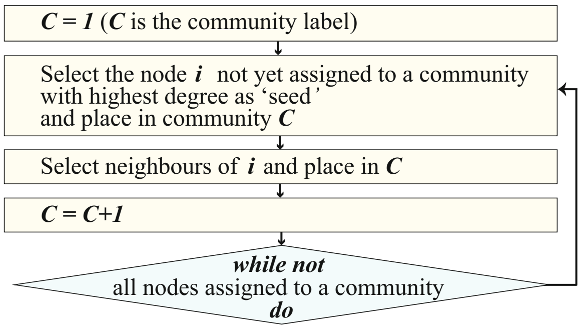
The initialization procedure adopted by our algorithm generates a population of S individuals (i.e. feasible solutions), in which $|S|$ is an algorithm input parameter. Figure 9.8 shows the algorithm for creating a new individual.

Additionally, a single individual is created, starting with the highest degree node as *seed* and expanding its *natural communities* until all vertices have been assigned to a community. Figure 9.9 details the procedure to create the additional individual.



C = 1 (*C* is the community label)
Select a node *i* (not yet assigned to a community) with uniformly random probability as 'seed' and place in community *C*
Select neighbours of *i* and place in *C*
C = C+1
while not all nodes assigned to a community *do*

Fig. 9.8 Steps performed to generate a new individual. This method ensures that nodes in the same community share at least one link, therefore, all individuals are valid showing communities with a single connected component



C = 1 (*C* is the community label)
Select the node *i* not yet assigned to a community with highest degree as 'seed' and place in community *C*
Select neighbours of *i* and place in *C*
C = C+1
while not all nodes assigned to a community *do*

Fig. 9.9 Steps performed to detect natural communities in a graph. Differently from algorithm shown in Fig. 9.8, rather of randomly chosen nodes, the highest degree nodes serve as seeds

The output of the algorithm from Fig. 9.9 encodes one individual, placed randomly in the population. Figure 9.10 shows the result of this procedure for a small network.

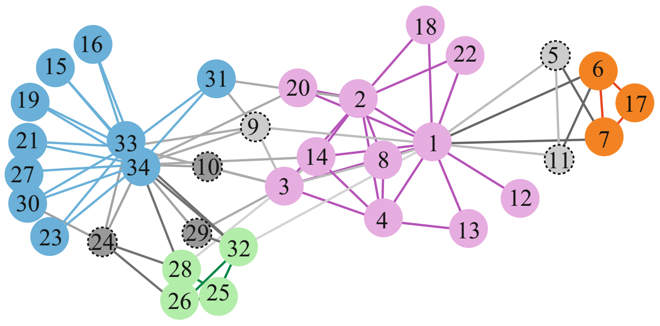


Fig. 9.10 Network partition found by the procedure of algorithm shown in Fig. 9.9 for a small network. Nodes circled with dashed lines are the remaining that would need to have different assignments to reach optimum modularity score for this network

While this initialization procedure is a heuristic (i.e. it does not produce the optimal modularity partition), it nevertheless is far better than a random assignment. It helps to drastically reduce the number of generations required to converge to an optimal solution. After the initial population is built this way, the algorithm progresses executing the selection, crossover, mutation, local search-based iterative improvement and replacement mechanisms to evolve the individuals in the population.

9.3.3 Selection and Replacement

At each generation MADOC generates S new individuals, where $|S|$ is the size of the population. The crossover/recombination mechanism selects two individuals as "parents" and recombines them to generate a new "offspring" solution. Along with the crossover and mutation operators, the selection mechanism is responsible for improving the quality of the solutions. This motivates our choice, one parent is randomly selected among the pool of best individuals in the current population (this pool is equal to 10% of the population size). The second parent is selected using tournament selection, by selecting a sub-population of size $|S'|$, that is the tournament size parameter, from the current population, the best individual of this sub-population is selected as the second parent. Our choice creates a selection pressure that can be adjusted by increasing or decreasing the tournament size [36].

Replacement also plays an important role in the evolutionary process. After the crossover of two parents create an offspring new solution, mutation and local search are applied to the offspring and its fitness is updated. The offspring replaces the less-fit individual in the population, only if, the quality of the offspring is better than the less-fit individual in the current population. This combination of selection and replacement is an elitist evolutionary strategy where the fittest individuals thrive.

9.3.4 Recombination

Recombination is the process of merging or combining two individuals, the parents, to produce an offspring solution with characteristics inherited from both parents as an essential mechanism of reorganization [20]. The crossover mechanism employed by MADOC is the *modularity-based crossover* [12, 39]. However, our crossover differs from the previous algorithms in [12, 39] by constructing two priority lists, one for each parent, instead of a global priority list. Figure 9.11 shows the crossover of two parents into an offspring.

The modularity Q of a community defines its priority, communities with higher modularity have higher priority. To generate the offspring we select the nodes from the community with the highest priority in the first parent, next we select the nodes in the community with the highest priority in the second parent, repeating the process until set all nodes in the offspring. Consequently, we ensure that the offspring will receive characteristics present in the solution structure from both parents.

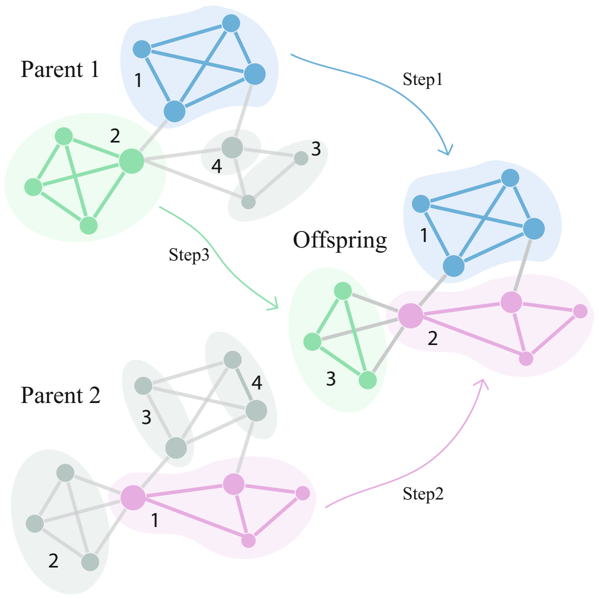


Fig. 9.11 An example of the modularity-based crossover operation. Each parent has its own priority list based on the community's fitness. The first step is to copy the nodes in the community with priority 1 from parent 1. Next, copy the nodes in the community with priority 1 from parent 2, then community 2 from parent 1, and so on, until all nodes have been placed in the offspring. Notice that in the third interaction, one node in the community 2 of parent 1 is already assigned to the community 2 in the offspring. In this case, there is no need to change this node in the offspring

9.3.5 Mutation

The mutation operator modifies characteristics of an individual according to a certain probability, this adds a level of perturbation to the solution. The mutation operator helps to preserve diversity in the population [13, 20, 28, 38]. As such, it is a necessary but "background operator" in memetic algorithms [37, 38, 40].

MADOC employs a *mutation probability* as a threshold to control the amount of perturbation introduced in a solution. The mutation probability *mp* ranges from 0.5 (5% of probability for a string to be modified by mutation) to 0.15 (15% of probability for a string to be modified by mutation), it is increased in amounts of 0.1 if there is no gain in the average fitness for more than 15 generations until reaches the maximum. Figure 9.12 shows the steps performed by the mutation operator.

Our mutation operator traverses the graph starting from node 1. If the uniformly random value generated is below than the mutation rate, the node receives a new random community in the range of 0 to n, where n is the number of nodes in the graph, and all its neighbours are assigned to the same community. Next, the algorithm moves for the next node that has not been modified yet.

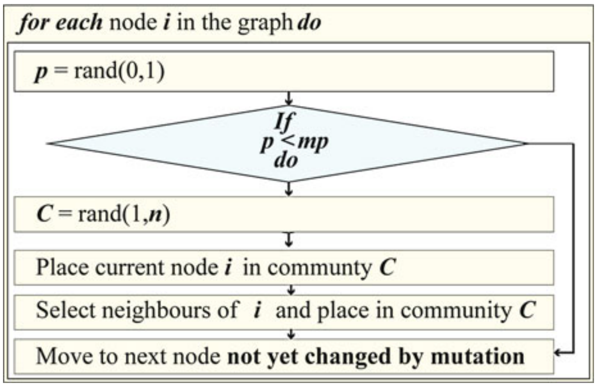

for each node i in the graph do
p = rand(0,1)
If p < mp do
C = rand(1,n)
Place current node i in communty C
Select neighbours of i and place in community C
Move to next node not yet changed by mutation

Fig. 9.12 The schematic steps performed by the mutation operator. Usually, the mutation procedure changes one gene at the time. Our mutation, however, changes a gene (node) and the genes which correspond to the neighbours of this node, therefore, avoid creating infeasible solutions during the mutation process

9.3.6 Fitness Function

For population-based search methods, the fitness function is the mechanism used to evaluate the quality of a solution encoded by an individual due to the given problem and the particular problem instance being addressed [40]. The fitness function sometimes equals the objective function we seek to maximize (or minimize) but in other cases it may different. The fitness function then becomes the fundamental score used for the selection and replacement mechanisms; it also represents the ultimate goal to be optimized. Here we employ the modularity optimization [42], a highly accepted score generally used to evaluate the quality of network partitions. A higher modularity value indicates good quality communities. Equation (9.1) shows how to compute modularity in a graph $G = (V, E)$ with k communities.

$$Q = \sum_{c=1}^{k} \left[\frac{l_c}{m} - \left(\frac{d_c}{2m} \right)^2 \right], \tag{9.1}$$

where c represents a community, k is the number of communities, $m = |E|$, l_c is the number of edges that has both nodes connected by the edge in the same community and d_c is the sum of the degrees of the nodes inside a community.

The objective of MADOC is then to obtain the partition that has the maximum modularity of a line graph $L(G)$ constructed from a graph G.

9.3.7 Local Search

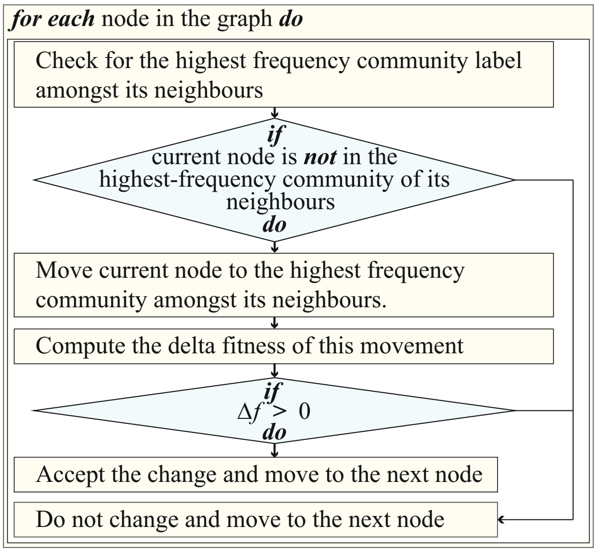
An important characteristic of MAs is the incorporation of the domain knowledge to the problem, preserving this knowledge during the search [37, 40]. Memetic algorithms combine a global search on a population of individuals with mechanisms of local improvement (local search). Inspired by the label propagation methods we named MADOC's local search mechanism as *'local consensus search'*, since it attempts to change a node label to the label attributed to the largest number of its neighbours [17].

The local search mechanism considers the gain in fitness caused by the acceptance of the proposed change. Due to the numerous permutations employed to improve each individual during the local search, computing this gain (or loss, if it would lead to a worse solutions) is often the most expensive task performed by an MA. The difference between the fitness of an individual Ind and the fitness of a neighbouring individual Ind' is called delta fitness. Equation (9.2) shows the delta fitness

$$\Delta f(Ind, Ind') = f(Ind') - f(Ind), \tag{9.2}$$

in which the term $\Delta f(Ind, Ind')$ is the result of the perturbation from the solution Ind to the new solution Ind'. Figure 9.13 shows the steps performed by the local search algorithm.

For MADOC's iterative improvement basic strategy, the local search is based on changing a node's membership one at a time. A change of membership of a node is only considered to a new community that a node already shares at least one connection. The local search procedure traverses the graph each node at a time



for each node in the graph *do*
Check for the highest frequency community label amongst its neighbours
if current node is *not* in the highest-frequency community of its neighbours *do*
Move current node to the highest frequency community amongst its neighbours.
Compute the delta fitness of this movement
if Δf > 0 *do*
Accept the change and move to the next node
Do not change and move to the next node

Fig. 9.13 Steps performed by our local search algorithm, the *local consensus search*. Similarly to the label propagation methods, the community with the highest frequency in the neighbourhood of a node has a higher probability as the candidate community to move a node. If the node is already in the highest frequency community among its neighbours, the node is skipped to ease the number of required permutations to traverse the graph

and a movement is accepted only if the fitness after the perturbation outperforms the fitness before the perturbation. To reduce the number of operations, it accepts the first movement that increases the fitness of the individual. When a movement is made, or all possible changes are tested, the local search moves to the next node.

9.3.8 Parameters Used by MADOC

MADOC uses three main parameters to control the algorithm's behaviour: $|S|$ (i.e. the population size), mp (mutation probability) and T_{max} (maximum number of generations). We performed a Wilcoxon signed-rank test to compare two sets of parameters, **Set A** and **Set B**. The first, **Set A**, with the same values used by algorithms MLCD [31], MOGA-Net [46] and MODPSO [14], and **Set B**, following the algorithm MA-Net [39]. Table 9.1 shows the values used by each algorithm.

Table 9.1 Parameters used by the community detection MAs we tested

	Set A	Set B
Parameter	MLCD, MOGA-Net, MODPSO	MA-Net
S (Population size)	300	40
mp (Mutation probability)	0.15	0.05–0.15[a]
T_{max} (Maximum generations)	200	30[b]

[a]The mutation rate is adjusted according to the fitness
[b]The number of generations without fitness improvement

Three benchmark networks were used to test the parameters sets, Zachary Karate Club, American College Football and a LFR synthetically generated network with 128 nodes and 1024 edges and mixing parameter $\mu = 0.1$. Tests were performed over 50 independent runs for each benchmark network with the two sets of parameters summarizing 300 test runs. Results showed no statistically significant difference between the two sets of parameters for the Zachary Karate Club network and the LFR benchmark. However, the **Set B** performed better for the American College Football benchmark network with a marginal statistically significant difference, with *p-value = 0.04837*, also, producing higher average fitness and smaller standard deviation. The experimental results presented in the subsequent sections were obtained with parameters following the values of **Set B**.

9.4 Computation Experiments in Synthetically Generated Networks

To assess the performance of MADOC, we conducted computational experiments using synthetically generated and real-world benchmark networks. In this section we present the computational results in a well-known benchmark of synthetically generated networks where the communities are known in advance. In the subsequent sections, we present two case studies using real-world networks.

Lancichinetti et al. [25] proposed an algorithm to synthetically generate complex networks (for testing purposes) such that the ground truth partition of the nodes is known. They employ the following parameters to generate these networks: degree distribution, minimum and maximum degree, minimum and maximum size of communities and a mixing parameter μ, used to define the fraction of connections between a node and its community and the number of connections that it shares with the rest of the network. Equation (9.3) defines the mixing parameter μ.

$$\mu = \frac{Z_{out}}{Z_{in} + Z_{out}}, \tag{9.3}$$

where Z_{out} is the number of connections of a node connected with others outside its community, and Z_{in} is the number of connections of a node connected with nodes in its community. If $\mu \leqslant 0.5$, the benchmark will produce networks with well-defined communities.

Lancichinetti et al. [23] later updated the benchmark procedure with the addition of overlapping structure to the communities, where nodes can participate in more than one community. The overlapping fraction of the network is set with the parameters: *on* (the number of overlapping nodes) and *om* (the number of memberships of the overlapping nodes). Since then, the LFR benchmark has become the reference method for evaluating overlapping community detection algorithms and several authors have used it [5, 26, 31, 45, 54]. We used five synthetically generated networks, LFR-1 to LFR-5, to test the performance of MADOC. Table 9.2 depicts the details.

Table 9.2 Benchmark networks used to assess the performance of MADOC to detect overlapping communities in several graphs $G(V, E)$, where $|V|$ is the number of nodes, $|E|$ is the number of edges, $avg(k)$ is the average degree, μ is the mixing parameter, *on* is the number of overlapping nodes and *om* is the number of communities for the overlapping nodes

		$G(V, E)$					$L(G)$		
Network	$\|V\|$	$\|E\|$	$avg(k)$	μ	*on*	*om*	$\|V\|$	$\|E\|$	$avg(k)$
LFR-1	500	2490	9.96	0.098	0%	2	2490	29,101	23.37
LFR-2	500	2309	9.24	0.102	10%	2	2309	24,720	21.42
LFR-3	500	2403	9.61	0.097	20%	2	2403	27,069	11.26
LFR-4	500	2387	9.54	0.097	30%	2	2387	26,610	11.14
LFR-5	500	2457	9.82	0.097	40%	2	2457	28,305	11.52

To compare the solutions obtained by MADOC with the known structure, we employed the *normalized mutual information* (NMI) score to evaluate the agreement with the ground truth [6, 24, 54] (as it is normally used to compare the results of clustering algorithms against one assumed to be the ground truth). Table 9.3 shows the results obtained by MADOC in terms of similarity with the known benchmark network communities.

Table 9.3 Results of 50 independent runs on five test networks, where k_e is the number of expected communities to be observed (due to the generating procedure used), sd is the standard deviation of the NMI score used to compare the results of MADOC on five benchmark networks

Network	k_e	sd	Max Q on $L(G)$	NMI
LFR-1	20	0.0038	0.8329	0.882
LFR-2	20	0.0112	0.8329	0.948
LFR-3	23	0.0242	0.8284	0.888
LFR-4	29	0.0317	0.7231	0.756
LFR-5	24	0.0876	0.6953	0.767

MADOC obtained good and stable results across multiple runs with low standard deviation. The overlapping communities found show high similarity with the ground truth community structure in the networks used on these computational tests. The next sections show the detailed computational tests and results using two real-world networks.

9.5 Overlapping Communities of the Characters of the Novel "A Storm of Swords": A Case Study

This section presents our first case study, the overlapping communities of the characters of the novel "A Storm of Swords" [32] of the book series "A Song of Ice and Fire" written by George R. R. Martin and broadcasted as by HBO as the famous series *Game of Thrones*.[2] The network[3] is comprised of 107 nodes (characters) and 353 weighted edges. Two characters are connected if they appear in the book separated by at maximum 15 words. The edge weight is the total of interactions between two characters. This network was also discussed in Chap. 8 in the context of a centrality analysis of its characters. Further details about the dataset are given in Chap. 26 that collects the information of most datasets used in it.

To detect the overlapping communities of characters in the "Storm of Swords" network we followed the same approach previously shown in this chapter. We first transformed the network into a line graph and computed the communities of edges by finding the partition that maximizes the modularity on the line graph. Later, we recover the information from the line graph to finally obtain eight overlapping communities with size ranging from 23 to 38 characters.

[2]http://www.hbo.com/game-of-thrones.

[3]https://www.macalester.edu/~abeverid/thrones.html.

While a few characters participate in more than three communities, the majority of characters (76%) are assigned to one, two or three communities. Figure 9.14 shows on the left a table with the number of characters assigned to multiple communities and on the right a bar chart with distribution of the number of communities per character.

Number of communities	Count	Percentage
8	0	0%
7	2	2%
6	4	4%
5	3	3%
4	17	16%
3	8	7%
2	19	18%
1	54	50%

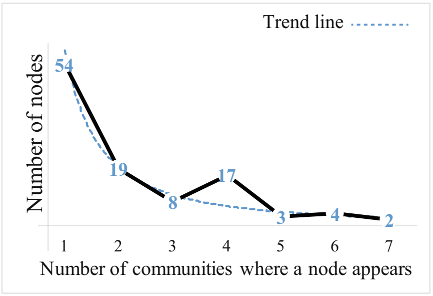


Fig. 9.14 Left, the count and percentage of characters according to the number of communities they belong to. Right, the chart with the distribution of the number of communities per character and a trend-line of this distribution

The novel consists of an intricate plot with involving several families, Kings and their courts. Located in Westeros, its Seven Kingdoms are still involved in the War of the Five Kings. Then, it is illustrative to mention the five factions: *The King on the Iron Throne*, **Joffrey** Baratheon, heir of King **Robert** I Baratheon; *The King in Highgarden*, **Renly** Baratheon, the youngest brother of King Robert; *The King in the North and the Trident*, **Robb** Stark, the heir of Lord **Eddard** Stark of Winterfell; *The King in the Narrow Sea*, **Stannis** Baratheon, the elder of King Robert's younger brothers; and *The King of the Isles and the North*, **Balon** Greyjoy, Lord of the Iron Islands.

As perhaps expected, these individuals appear in many communities. This is due to the fact that they would interact not only with members of their kingdoms but with many other characters that do not belong to them. Accordingly, these factions are related to the communities we observe: Balon is present in five communities, Stannis in seven, Robb in four, Renly in five and Joffrey in six.

Other characters with that are members of several communities are **Robert** (a label for King Robert I Baratheon, in seven communities), **Arya** (for Arya Stark, sister of Robb Stark), **Cersei** (for Cersei Lannister), and **Eddard** (in six communities), and **Lysa** (in five). Cersei married King Robert I Baratheon and became Queen of the Seven Kingdoms; she is the mother of Prince Joffrey. The author of *"A Storm of Swords"* employs a point-of-view (POV) approach, with each chapter presenting the thoughts of one particular character. Interestingly, Arya's

POV is present in 13 chapters of *"A Storm of Swords"* indicating her central role in the book's plot. Figure 9.15 shows the network and the eight overlapping communities highlighted by distinct colours.

Figure 9.16 shows each community highlighted by a distinct colour in the network. In this network, we could observe that the number of communities in which a character participates is correlated to its centrality. As shown in Chap. 8 which also includes the network "Storm of Swords" as an example, centrality is a measure of influence and importance in the network. Besides few exceptions in which major characters appear in a single community, the majority of influential characters those that appear as members of communities.

The author of Chap. 8 gently provided a list of metrics of strength and centrality for each character; we used these metrics to compute the correlation between the number of communities in which a character appears compared to its strength and centrality of the characters. Table 9.4 shows the Pearson's, Spearman's and Kendall's correlation with six centrality measures.

Table 9.4 The Pearson's, Spearman's and Kendall's correlation between the number of communities to which a character is associated with and metrics of strength and centrality

Correlation	Degree	Strength	Weighted closeness	Weighted betweenness	Weighted eigenvector	Weighted pagerank
Pearson's	0.5145	0.3385	0.5870	0.2645	0.5168	0.2865
Spearman's	0.6508	0.4919	0.6000	0.3771	0.7375	0.4496
Kendall's	0.5516	0.3942	0.4688	0.3297	0.6006	0.3577

This analysis shows that the eigenvector centrality has the strongest correlation with the number of communities in which a character appears. Distinctively from other metrics, the eigenvector centrality takes into account topological position of nodes in the network. The PageRank algorithm and the Katz centrality can be seen as variants of this methodology and in turn as variants of the concept of eigenvector centrality. If a node has a high value of its eigenvector score it means, in turn, that it is connected to many nodes who themselves have high scores.

Table 9.5 shows the characters with the highest centrality score listed in Chap. 8 ordered by the number of communities in which they participate.

In Table 9.5, along with characters with multiple community assignments, there are four important and central characters which are associated with a single community, **Daenerys**, **Jon**, **Samwell** and **Tyrion**. These four characters play important roles in the plot. However, they interact mostly with their group or alliance, therefore, appearing in a single network community. In terms of their participation as narrators from a POV perspective, Jon Snow is in 12 chapters and

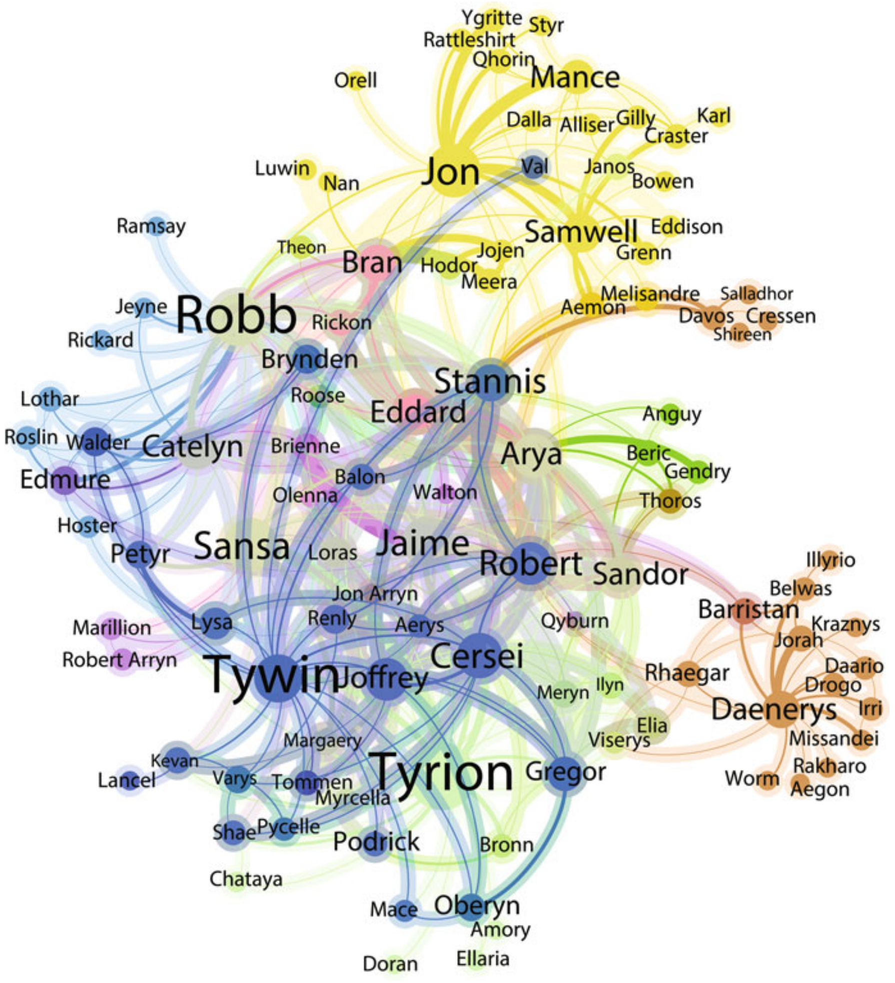


Fig. 9.15 The eight overlapping communities in the "Storm of Swords" network. The font size of each node is proportional to their score in terms of eigenvector centrality

Tyrion Lannister in 11 in the book, confirming their central role.[4] Both, however, seem to be more restricted to their communities, while others seem to be linking many communities. In conclusion, we can observe that centrality measures offer an important contribution to network analysis complementing the information provided by the overlapping community structure.

[4]http://awoiaf.westeros.org/index.php/POV_character.

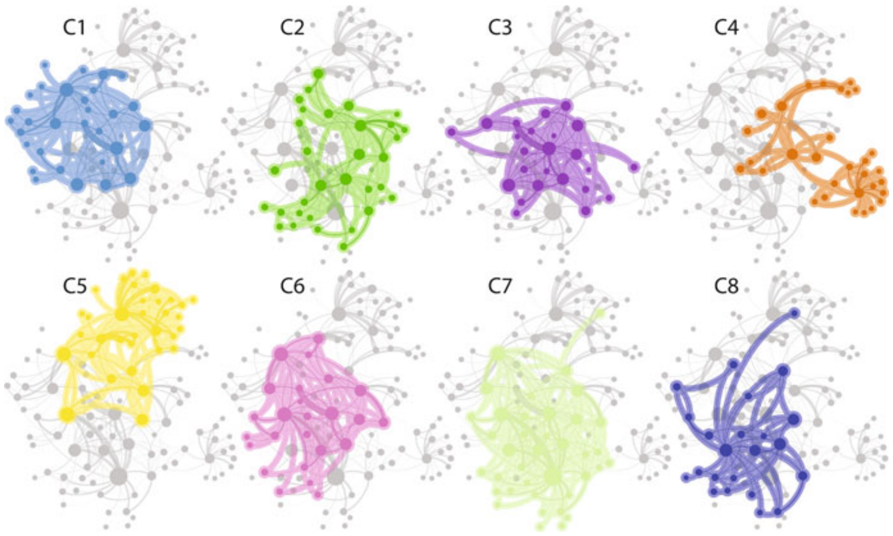


Fig. 9.16 Communities C1 to C8 highlighted by distinct colours in the "Storm of Swords" network

9.6 Overlapping Communities in Product Co-purchasing Networks: A Case Study

Overlapping community detection can provide an extremely useful tool for consumer behaviour (co-purchasing) analysis in an e-commerce setting. This has recently been shown in other papers (see, for instance, [22, 56]) where other co-purchasing networks have been analysed using a community detection approach.

More specifically, for the second case study of this chapter, we look at the co-purchasing network of products sold through the Amazon website. The analysis of co-purchasing data can reveal relationships between products, brands and the associations between categories of products. Good recommendation systems can enhance e-commerce sales by suggesting to potential customers the correct segment of products [29]. Furthermore, the analysis of networks of brands can be used for business and marketing decision making purposes [30, 49] and advertisement planning and recommendation systems [3] among other uses.

To show the usefulness of MADOC in a business and consumer analytics context, we investigate a group of 570 co-purchased items from Amazon.com to depict the overlapping communities of products, categories, brands and the distribution of targeted gender, rising from a seed brand. The dataset used in this experiment is a subset of a larger dataset containing 191,000 products crawled from the Amazon.com website [33, 34]. Products' metadata includes the description, price, sales rank, brand and co-purchasing links. The co-purchasing links include *also bought*,

Table 9.5 The most central characters of the network “Storm of Swords” ordered according to the number of communities in which they appear

Node	C1	C2	C3	C4	C5	C6	C7	C8	Number of communities
Robert		1	1	1	1	1	1	1	7
Stannis	1	1	1	1	1		1	1	7
Arya	1	1	1		1	1	1		6
Cersei	1	1	1			1	1	1	6
Joffrey	1	1	1			1	1	1	6
Robb	1				1	1	1		4
Bran	1	1			1	1			4
Brienne	1	1	1			1			4
Catelyn	1		1			1	1		4
Jaime	1		1			1	1		4
Margaery		1	1			1	1		4
Tywin	1		1				1	1	4
Sansa					1	1	1		3
Daenerys				1					1
Jon					1				1
Samwell					1				1
Tyrion							1		1

also viewed, *bought together* and *buy after viewing*. In our experiment, we used the *also bought* information to create a complex network of co-purchasing items.

To construct our subset network, we chose the brand *Versace* as the seed, we then selected the items of this brand with the “also bought” information available in the metadata (nb: not available for all products) as the first group of nodes. We used the ASIN[5] as nodes’ IDs and the description and brand as nodes label. Next, we added the products in the “also bought” list to the graph with the links between these products and the Versace products, summarizing 570 products of 176 distinct brands. Finally, we connected each product in the graph to the “also bought” items present in the graph. The resulting network consists of 570 nodes and 5933 edges. Figure 9.17 shows the distribution of the products according to category and gender. Further information on this dataset and the full Amazon dataset are provided in Chap. 26. To compute the overlapping communities, the graph is transformed into a line graph where MADOC performs modularity optimization to identify the communities in the line graph. The line graph created from this graph comprises of 5933 nodes and 264,008 edges. By performing modularity optimization seven communities are found. Subsequently, the information of the communities in the line graph is recovered to the original graph, resulting in seven overlapping communities. Figure 9.18 shows the distribution of the communities

[5] Amazon Standard Identification Number.

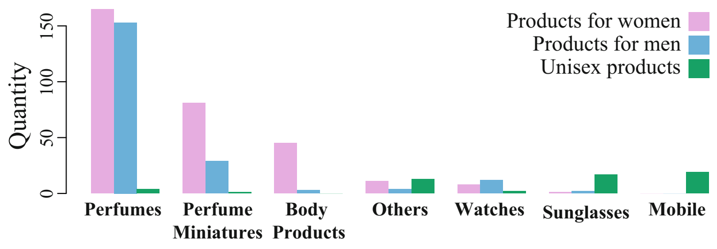


Fig. 9.17 Distribution of the products according to category and targeted gender

according to size and targeted gender of products. Figure 9.19 shows the areas in the network covered by each community highlighted by different colours and Fig. 9.20 shows the word clouds[6] of brands for each community. The brands that appear more frequently have a larger font size than the less frequent. At a first glance, the community size distribution may seem odd, with the community C3 covering 71% of the graph, even though, there is no product that appears in all communities, only 11 products (1.9%) appear in six communities and 185 products (32%) appear in a single community. We have also 136 products in two communities, 101 in 3, 72 in 4 and 65 in 5 communities.

Table 9.6 details the categories of products according to the number of communities they appear in, as well as the number of brands and targeted gender. This shows that of those few (11) products that are members of all communities, all of

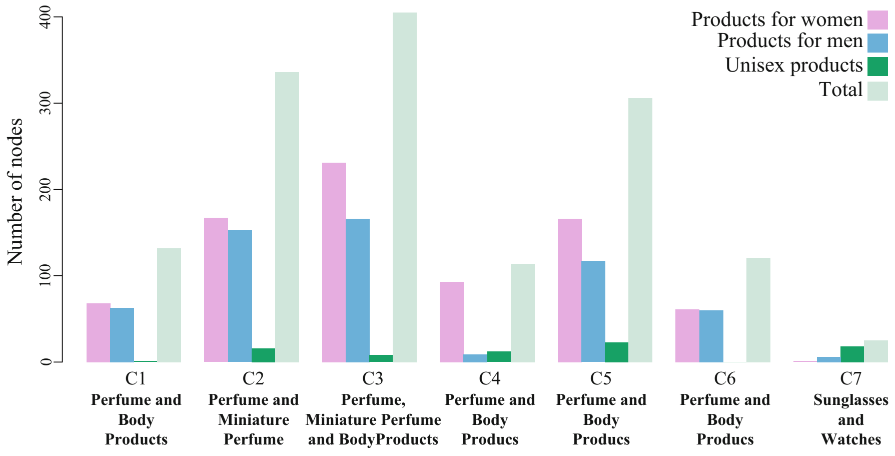


Fig. 9.18 Distribution of size for communities C1 to C7 recovered from the line graph to the original graph with the distribution of targeted gender and dominant categories of products for each community

[6]http://www.wordle.net.

them are perfumes. When we decrease the number of communities that products appear in, we start to see more variety with still 89% perfumes being members of 5 communities, 7% body products (e.g. body lotion) and 4% perfume miniatures which we keep as a separate category to "perfumes" as they can be bought with very differing intentions and they differ highly in price. From the information in this table we can see that the non-perfume products (in this network including mostly perfumes) are more likely to be in only 1, 2 or 3 communities. This means that these products have lower levels of heterogeneity in terms of being bought together with perfumes. This shows MADOC's successful separation of products into separate communities that "make sense" in terms of co-purchasing behaviours. It is more likely that perfumes are bought together with other perfumes or body products than, for instance, watches, sunglasses or smartphones. However, it is interesting to see that these cross-category co-purchasing behaviours are still present as it provides brand managers an image of where their brand reputation stands among other brands and what else their consumers are interested in. Finding out more about your consumers is usually a key questions for business and marketing managers. As part of the analysis, we built the network of brands, in the same manner as for the products but each node represents one brand that was present in the dataset. A connection is placed between brands *A* and *B* when a product from brand *B* appears in the "also bought" list of product *A*, the number of co-purchases corresponds to the weight of the edges. The complete network of brands has 177 nodes and 1606 edges. To simplify the visualization we filtered the network of brands, keeping only the brands with five or more co-purchases, the result of this filtering is a graph with 52 nodes (brands) and 100 edges. Figure 9.21 shows the brand's network. The dashed lines show the separate communities found. The brand Versace naturally appears in the centre of the graph, as it is the brand chosen as the seed. The four communities in this graph show four distinct categories of products (a) mainly perfumes and products targeted more at a female audience, (b) men's and women's perfumes, (c) more men's perfumes/colognes brands and (d) mobile phones and watch brands.

The connections between these communities allow brand managers to obtain new insights from their "brand network", its structure and decomposition. As we only retained those connections between products that were "also bought" by a consumer, it means that a real record of co-purchasing exists between products of the brands that are connected. Brands with a higher weighted connection are likely targeting the same (or similar) target market and those with a weak or no connection can assume that their consumer base is, in principle, quite different. For example, the "heavier" edges between Versace and some brands (as opposed to the thinner edges to others) provide certain insights for the brand managers of Versace. Using this information, and this visualization, we can see that Versace's customers are also likely to be customers of Dolce & Gabbana, Bvlgari, Gucci, Givenchy and/or

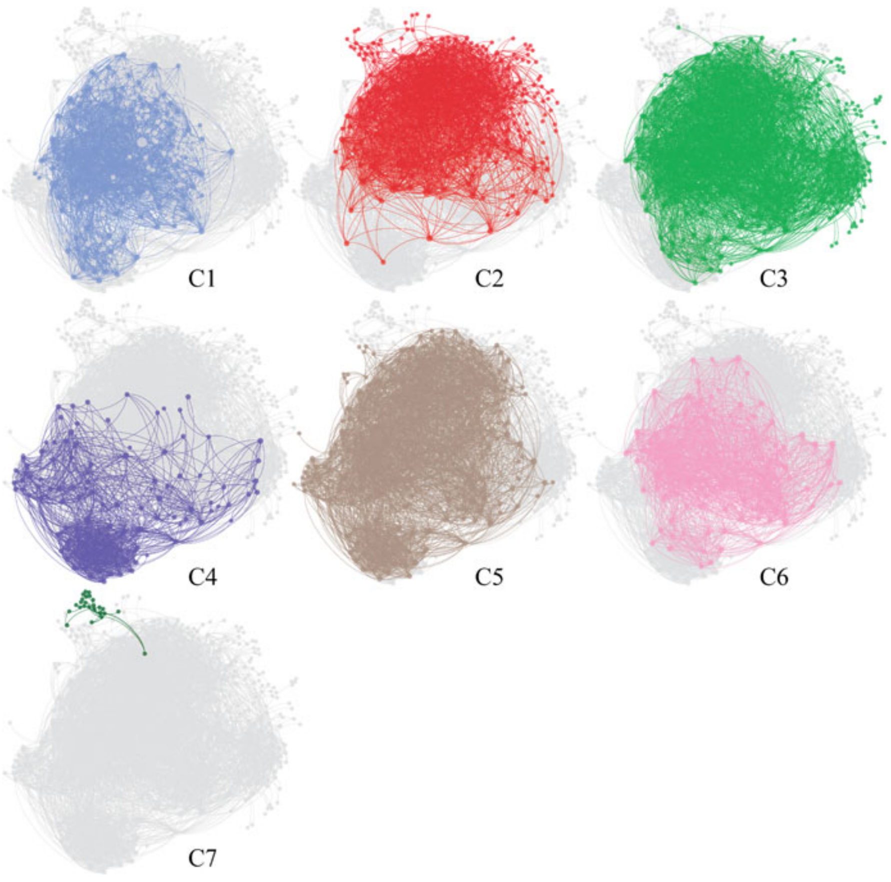


Fig. 9.19 The areas in the graph covered by communities C1 to C7 highlighted by different colours

Calvin Klein. What is interesting is that some of these other brands also have fairly "heavy" connections to the brand Paco Rabanne. However, this brand has a weaker connection to Versace than the other brands we just mentioned. Considering that some of Versace's customers co-purchase products from quite a few brands that are connected to Paco Rabanne, this could be an attractive new target market for Versace.

Furthermore, another example we can highlight from this figure is the strong connection between Versace and Victoria's Secret, the strong connection between Victoria's Secret and the Paris Hilton brand but the much weaker connection

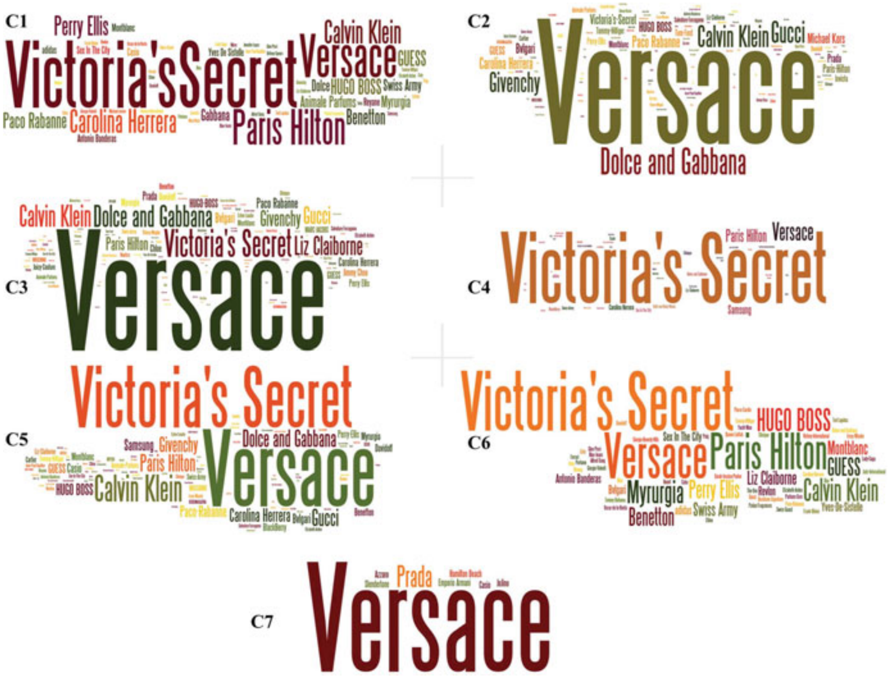


Fig. 9.20 Word clouds of brands for communities C1 to C7 in the Versace's co-purchase network. Brands with larger font size are the most frequent brands in the community, brands with smaller fonts are less frequent in the community

showing between Versace and Paris Hilton. This shows that Versace indeed "shares" customers with Victoria's Secret but not so much with the Paris Hilton brand. There could be some interesting reasons for these purchasing patterns. A complete understanding may also need to include the type of target market and demographics that these brands are trying to target. It could be said that out of the three brands mentioned, Versace is the most "up market" brand, Victoria's Secret the medium one and the Paris Hilton brand the slightly lower one (in terms of price, target market, etc.). Although sometimes these relationships and differences between brands could be obvious to brand managers, other times they may not be so obvious. For instance, Versace may not have expected to share as many customers with other brands that would not normally be considered to be at the top of the "luxury brand scale" which means there could be room to grow and new possible target markets for Versace (in this online setting).

Table 9.6 Distribution of product categories according to the number of communities where these products are present, number of brands and targeted gender

Number of communities	Items count	Number of brands	Product categories	Women's product	Men's product	Unisex
6	11	10	Perfumes 100%	73%	27%	0%
5	65	37	Perfumes 89%, Body products 7%, Perfume miniature 4%	48%	52%	0%
4	72	42	Perfumes 80%, Body products 15%, Perfume miniature 5%	56%	44%	0%
3	101	48	Perfumes 81%, Body products 8%, Perfume miniature 6%, Others 5%	58%	41%	1%
2	136	56	Perfume 50%, Perfume miniature 18%, Mobile 10%,Body products 8%, Watches 8%, Sunglasses 3%,Others 3%	52%	35%	13%
1	185	93	Perfume miniature 41%, Others 20%, Perfume 19%, Sunglasses 8%, Body products 7%, Watches 5%	57%	21%	22%

9.7 Conclusions

This chapter has presented a memetic algorithm for the problem of identifying overlapping communities in simple undirected graphs. We have seen, using two real-world networks, that the method allows to provide good solutions for a mathematical model in which the nodes of the graph can belong to different sets. Moving away from "just considering" partitions of a set of nodes of a graph has also modelling advantages. The results presented, for still relatively small real-world networks and synthetically generated graphs, show that the method has some promise to be part of the set of tools required by data scientists to investigate how to extract relevant structure from these networks. We speculate that more research will be conducted in the area of overlapping communities and that probably the use of multi-objective optimization since other score functions, aside of modularity, are being proposed in the literature. We hope that the results presented would encourage our readers to try the approach and work on the scalability of the basic memetic algorithm components which need to be improved to address networks involving millions of nodes.

Acknowledgements The authors of this chapter acknowledge Dr. Sergio Gómez for his contribution providing the centrality data of the network "Storm of Swords" and for his valuable comments. We also thank Ceci Brie, for her help interpreting some aspects of the obtained communities

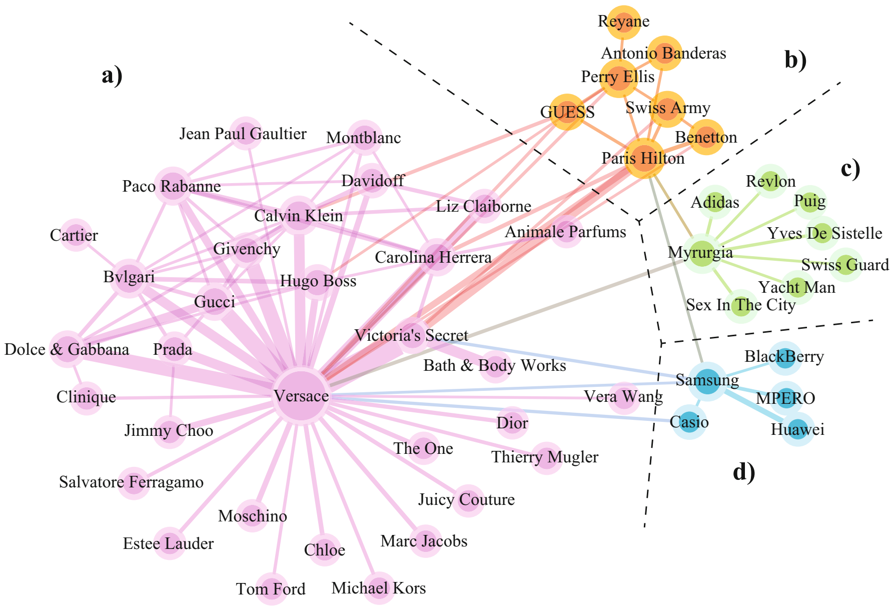


Fig. 9.21 A simplified network of brands of co-purchases with Versace on Amazon.com. We only present nodes corresponding to brands with five or more co-purchases, the result is a graph with 52 nodes (brands) and 100 edges. The weights of the edges (proportional to their width in the illustration) represent the number of co-purchases of products of two different brands. The network is divided into four communities (**a**)–(**d**) highlighted by dotted lines and different colours

and their structure. Ademir Gabardo is supported by CNPQ Brasil (http://cnpq.br), grant number 204978/2014-9. P.M. acknowledges this research by The University of Newcastle, and previous founding from the Australian Research Council grants Future Fellowship FT120100060 and with R.B. they acknowledge support via Discovery Project DP140104183.

References

1. Ahn, Y.Y., Bagrow, J.P., Lehmann, S.: Link communities reveal multiscale complexity in networks. Nature **466**(7307), 761–764 (2010)
2. melio, A., Pizzuti, C.: Overlapping community discovery methods: A survey. In: Social Networks: Analysis and Case Studies, pp. 105–125. Springer (2014)
3. Ansari, A., Essegaier, S., Kohli, R.: Internet recommendation systems. Journal of Marketing Research **37**(3), 363–375 (2000)
4. Barabási, A.L., Frangos, J.: Linked: the new science of networks science of networks. Basic Books (2014)
5. Chen, W., Liu, Z., Sun, X., Wang, Y.: A game-theoretic framework to identify overlapping communities in social networks. Data Mining and Knowledge Discovery **21**(2), 224–240 (2010)
6. Danon, L., Diaz-Guilera, A., Duch, J., Arenas, A.: Comparing community structure identification. Journal of Statistical Mechanics: Theory and Experiment **2005**(09), P09,008 (2005)

7. Derényi, I., Palla, G., Vicsek, T.: Clique percolation in random networks. Physical review letters **94**(16), 160,202 (2005)
8. Evans, T., Lambiotte, R.: Line graphs, link partitions, and overlapping communities. Physical Review E **80**(1), 016,105 (2009)
9. Fortunato, S.: Community detection in graphs. Physics Reports **486**(3), 75–174 (2010)
10. Friggeri, A., Chelius, G., Fleury, E.: Egomunities, exploring socially cohesive person-based communities. arXiv preprint arXiv:1102.2623 (2011)
11. Gabardo, A.C., Berretta, R., de Vries, N.J., Moscato, P.: Where does my brand end? An overlapping community approach. In: Intelligent and Evolutionary Systems: The 20th Asia Pacific Symposium, IES 2016, Canberra, Australia, November 2016, Proceedings, pp. 133–148. Springer (2017)
12. Gach, O., Hao, J.K.: A memetic algorithm for community detection in complex networks. In: Parallel Problem Solving from Nature-PPSN XII, pp. 327–336. Springer (2012)
13. Goldberg, D.E., Holland, J.H.: Genetic algorithms and machine learning. Machine Learning **3**(2), 95–99 (1988)
14. Gong, M., Cai, Q., Chen, X., Ma, L.: Complex network clustering by multiobjective discrete particle swarm optimization based on decomposition. IEEE Transactions on Evolutionary Computation **18**(1), 82–97 (2014)
15. Gong, M., Cai, Q., Li, Y., Ma, J.: An improved memetic algorithm for community detection in complex networks. In: Evolutionary Computation (CEC), 2012 IEEE Congress on, pp. 1–8. IEEE (2012)
16. Gong, M., Fu, B., Jiao, L., Du, H.: Memetic algorithm for community detection in networks. Physical Review E **84**(5), 056,101 (2011)
17. Gregory, S.: Finding overlapping communities in networks by label propagation. New Journal of Physics **12**(10), 103,018 (2010)
18. Havemann, F., Gläser, J., Heinz, M.: A link-based memetic algorithm for reconstructing overlapping topics from networks of papers and their cited sources. In: 15th International Conference on Scientometrics and Informetrics (2015)
19. He, D., Jin, D., Chen, Z., Zhang, W.: Identification of hybrid node and link communities in complex networks. Scientific Reports **5** (2015)
20. Holland, J.H.: Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence. MIT Press (1992)
21. Huang, Z., Wang, Z., Zhang, Z.: Detecting overlapping and hierarchical communities in complex network based on maximal cliques. In: Social Media Processing, pp. 184–191. Springer (2015)
22. Jebabli, M., Cherifi, H., Cherifi, C., Hamouda, A.: Overlapping community detection versus ground-truth in Amazon co-purchasing network. In: 2015 11th International Conference on Signal-Image Technology & Internet-Based Systems (SITIS), pp. 328–336. IEEE (2015). http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=7400584&tag=1
23. Lancichinetti, A., Fortunato, S.: Benchmarks for testing community detection algorithms on directed and weighted graphs with overlapping communities. Physical Review E **80**(1), 016,118 (2009)
24. Lancichinetti, A., Fortunato, S., Kertész, J.: Detecting the overlapping and hierarchical community structure in complex networks. New Journal of Physics **11**(3), 033,015 (2009)
25. Lancichinetti, A., Fortunato, S., Radicchi, F.: Benchmark graphs for testing community detection algorithms. Physical Review E **78**(4), 046,110 (2008)
26. Lancichinetti, A., Radicchi, F., Ramasco, J.J., Fortunato, S.: Finding statistically significant communities in networks. PLoS One **6**(4), e18,961 (2011)
27. Li, Y., Liu, J., Liu, C.: A comparative analysis of evolutionary and memetic algorithms for community detection from signed social networks. Soft Computing **18**(2), 329–348 (2014)
28. Lima, C.F., Sastry, K., Goldberg, D.E., Lobo, F.G.: Combining competent crossover and mutation operators: a probabilistic model building approach. In: Proceedings of the 7th annual conference on Genetic and evolutionary computation, pp. 735–742. ACM (2005)

29. Linden, G., Smith, B., York, J.: Amazon. com recommendations: Item-to-item collaborative filtering. Internet Computing, IEEE **7**(1), 76–80 (2003)
30. Lucas, B., Arefin, A.S., De Vries, N.J., Berretta, R., Carlson, J., Moscato, P.: Engagement in motion: Exploring short term dynamics in page-level social media metrics. In: Big Data and Cloud Computing (BdCloud), 2014 IEEE Fourth International Conference on, pp. 334–341. IEEE (2014)
31. Ma, L., Gong, M., Liu, J., Cai, Q., Jiao, L.: Multi-level learning based memetic algorithm for community detection. Applied Soft Computing **19**, 121–133 (2014)
32. Martin, G.R.: A storm of swords. Bantam (2000)
33. McAuley, J., Pandey, R., Leskovec, J.: Inferring networks of substitutable and complementary products. In: Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 785–794. ACM (2015)
34. McAuley, J., Yang, A.: Addressing complex and subjective product-related queries with customer reviews. arXiv preprint arXiv:1512.06863 (2015)
35. McDaid, A., Hurley, N.: Detecting highly overlapping communities with model-based overlapping seed expansion. In: Advances in Social Networks Analysis and Mining (ASONAM), 2010 International Conference on, pp. 112–119. IEEE (2010)
36. Miller, B.L., Goldberg, D.E.: Genetic algorithms, tournament selection, and the effects of noise. Complex Systems **9**(3), 193–212 (1995)
37. Moscato, P., Cotta, C.: A modern introduction to memetic algorithms. In: Handbook of Metaheuristics, pp. 141–183. Springer (2010)
38. Moscato, P., et al.: On evolution, search, optimization, genetic algorithms and martial arts: Towards memetic algorithms. Caltech concurrent computation program, C3P Report **826**, 1989 (1989)
39. Naeni, L.M., Berretta, R., Moscato, P.: MA-Net: A reliable memetic algorithm for community detection by modularity optimization. In: Proceedings of the 18th Asia Pacific Symposium on Intelligent and Evolutionary Systems, Volume 1, pp. 311–323. Springer (2015)
40. Neri, F., Cotta, C., Moscato, P.: Handbook of memetic algorithms, vol. 379. Springer (2012)
41. Newman, M.E.: The structure and function of complex networks. SIAM review **45**(2), 167–256 (2003)
42. Newman, M.E.: Modularity and community structure in networks. Proceedings of the National Academy of Sciences **103**(23), 8577–8582 (2006)
43. Palla, G., Derényi, I., Farkas, I., Vicsek, T.: Uncovering the overlapping community structure of complex networks in nature and society. Nature **435**(7043), 814–818 (2005)
44. Paul, M., Anand, R., Anand, A.: Detection of highly overlapping communities in complex networks. Journal of Medical Imaging and Health Informatics **5**(5), 1099–1103 (2015)
45. Pizzuti, C.: GA-Net: A genetic algorithm for community detection in social networks. In: Parallel Problem Solving from Nature–PPSN X, pp. 1081–1090. Springer (2008)
46. Pizzuti, C.: A multi-objective genetic algorithm for community detection in networks. In: Tools with Artificial Intelligence, 2009. ICTAI'09. 21st International Conference on, pp. 379–386. IEEE (2009)
47. Pizzuti, C.: Overlapped community detection in complex networks. In: Proceedings of the 11th Annual conference on Genetic and evolutionary computation, pp. 859–866. ACM (2009)
48. Raghavan, U.N., Albert, R., Kumara, S.: Near linear time algorithm to detect community structures in large-scale networks. Physical Review E **76**(3), 036,106 (2007)
49. Rust, R.T., Huang, M.H.: The service revolution and the transformation of marketing science. Marketing Science **33**(2), 206–221 (2014)
50. Shi, C., Cai, Y., Fu, D., Dong, Y., Wu, B.: A link clustering based overlapping community detection algorithm. Data & Knowledge Engineering **87**, 394–404 (2013)
51. Wu, P., Pan, L.: Detecting highly overlapping community structure based on maximal clique networks. In: Advances in Social Networks Analysis and Mining (ASONAM), 2014 IEEE/ACM International Conference on, pp. 196–199. IEEE (2014)
52. Wu, P., Pan, L.: Multi-objective community detection based on memetic algorithm. PLoS ONE **10**(5), e0126,845 (2015)

53. Wu, Z.H., Lin, Y.F., Gregory, S., Wan, H.Y., Tian, S.F.: Balanced multi-label propagation for overlapping community detection in social networks. Journal of Computer Science and Technology **27**(3), 468–479 (2012)
54. Xie, J., Kelley, S., Szymanski, B.K.: Overlapping community detection in networks: The state-of-the-art and comparative study. ACM Computing Surveys (CSUR) **45**(4), 43 (2013)
55. Xie, J., Szymanski, B.K.: Community detection using a neighborhood strength driven label propagation algorithm. In: Network Science Workshop (NSW), 2011 IEEE, pp. 188–195. IEEE (2011)
56. Yamazaki, T., Shimizu, N., Kobayashi, H., Yamauchi, S.: Weighted micro-clustering: Application to community detection in large-scale co-purchasing networks with user attributes. In: Proceedings of the 25th International Conference Companion on World Wide Web, WWW '16 Companion, pp. 131–132. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland (2016). http://dx.doi.org/10.1145/2872518.2889406

Chapter 10
Taming a Graph Hairball: Local Exploration in a Global Context

James Abello, Daniel Mawhirter, and Kevin Sun

Abstract A variety of data sets can be modelled as an underlying reference topology where every vertex has an associated set of attributes or key words that are not necessarily derived from the topology. Blind application of most graph drawing algorithms produces a "hairball" even for modest graph sizes. In this chapter, we explore several intuitive mechanisms to decompose a weighted graph into vertex clusters and edge layers that facilitate user exploration. Our approach minimizes the screen bottleneck by enhancing the graph topology with weights on the given edges that encode their pairwise vertex similarity (avoiding the quadratic pairwise computation of vertex similarity if the given graph topology is sparse). The decompositions are based on graph distance topology clustering coupled with iterative peeling. Information from these decompositions is used to landmark vertices that drive egonet user exploration. These landmarks are vertices of "high diversity" in the decomposition that "dominate" the vertex set in terms of Shannon entropy. We illustrate the usefulness of this decomposition on the On-Line Encyclopaedia of Integer Sequences. Our method can be used as a first step for community detection, graph visualization, and data summarization. These techniques can be applied to situations in business analytics, for example, when studying a network of products frequently purchased together.

Keywords Core decomposition · Egonets · Graph exploration · Landmarks · Peeling · Shannon entropy · Shortest paths · Vector majorization · User engagement

J. Abello (✉)
DIMACS and Computer Science Department, Rutgers University, New Brunswick, NJ, USA
e-mail: abello@dimacs.rutgers.edu

D. Mawhirter
Colorado School of Mines, Golden, CO, USA
e-mail: dmawhirt@mines.edu

K. Sun
Rutgers University, New Brunswick, NJ, USA
e-mail: kevin.sun@rutgers.edu

P. Moscato, N. J. de Vries (eds.), *Business and Consumer Analytics: New Ideas*,
https://doi.org/10.1007/978-3-030-06222-4_10

10.1 Introduction

A variety of data collections can be modelled as graph topologies where each member of the collection is represented by a vertex with associated metadata and edges encode references among members of the collection. These references can be the result of human annotation or could be bibliographical references in a field of study or scientific discipline. The edges can be weighted via some “semantic” similarity measure between the vertex metadata. In some applications the goal is to discover “interesting” relations among the items in the collection that can be inferred from a combination of topological connectivity and “semantic” similarity. For example, we can use this work to untangle large, complicated graphs that resemble “hairballs”, such as a co-purchasing network of items. This work aims to provide some basic support for exploratory data mining where a computer user may wander around a topology and discover interesting structures while observing the computer’s feedback about his/her exploration routes. We base our approach on the notion of “Independent Data Landmarks”. They can be thought of as approximations to traditional “cluster centres” that are “independent” in the topology but that could be close in the “data semantics”. These independent landmarks can be overlaid on a partition of the vertex set. Their resilience to disappear from consideration when equivalence classes in the partition are iteratively removed gives us a measure of their diversity with respect to the partition (see Sect. 10.4). We choose to drive the data set exploration according to vertex partitions derived from the graph peel decomposition introduced in Alvarez et al. [2]. We provide a user interface driven by egonets and shortest path exploration. Most of our algorithms are approximate and suitable for the interactive navigation of large semi-external graphs (see Goodrich and Pszona [15]).

10.1.1 Independent Data Landmarks

In this work we exploit the inherent locality of “Independent Data Landmarks” and the vertex peel numbers to efficiently obtain “meaningful landmarks” and to explore a data collection in an intuitive manner via vertex egonets, where the egonet of a vertex is the subgraph induced by it and its neighbour vertices. Our aim is to provide a pairing of global and local exploration options for the data collection. Our interface can be used to examine a network topology at different levels of granularity without losing sight of the underlying vertex partition determined by the independent landmarks and their peel values. Since a vertex can be shared among different peel layers, we use this information to record a vertex’s peeling profile. This profile is an indicator of the vertex’s role in the graph data set. The vertex’s Shannon entropy measures its degree of involvement in the peeling community structure. We exemplify our findings on the On-Line Encyclopaedia of Integer Sequences (OEIS).

The rest of the chapter is organized as follows. Notational conventions and the basic concepts used are presented in Sect. 10.2. It also illustrates some of the main characteristics of subgraphs that are obtained by our decompositions . In Sect. 10.3, we introduce the distance based independent landmarks algorithms, its main properties, and an efficient algorithm to compute it. Sect. 10.4 indicates how to use these landmark decomposition jointly with peeling to filter and analyse a graph data set network at different scales and it proposes a measure of vertex diversity based on Shannon's entropy. Applications of the proposed edge decomposition to the OEIS is the subject of Sect. 10.7.1. We close with a discussion of possible future research directions in Sect. 10.8.

10.1.2 Related Work

Our work can be viewed as an attempt to identify large graph exploration mechanisms that may help amplify a user's understanding of the structure of a large graph data set. Current work in this area can be classified as topological, hierarchical, or hybrid methods.

10.1.2.1 Topological Methods

Bikakis et al. [9] proposed an approach which includes partitioning very large graphs via their position in a Euclidean space. The technique is promising for very large graphs and allows for efficient memory usage if the graph must be indexed and stored. User operations can then be translated into spatial operations on the graph. Their work uses a straightforward approach in order to make the visualization less ambiguous and to preserve the adaptability since some hierarchical clustering methods are very dependent on the input set. While this approach does provide the user with an easy navigable graph, it does not address the semantics which may be gained by users viewing the graph.

10.1.2.2 Hierarchical Methods

Nachmanson et al. [18] aim to navigate a graph much like a geographical map with GraphMap. This technique involves clustering the node and edges of a graph into zoom layers. Then via a top-down approach, as the user zooms through layers, a finer view is shown displaying the intersections of all layers currently in view. The zoom layer method allows users to view large graphs without exceeding a predefined threshold. While viewing, the graph is stable in the sense that all nodes and edges stay in the same relative position, allowing the geometry of the graph to be preserved. GraphMaps zooming and stability features aim to prevent users from becoming disoriented, while still being able to gain semantics. There are three main drawbacks to GraphMap: slow speed, loss of graph direction, and a lack of effective labelling.

10.1.2.3 Hybrid Methods

These are methods that incorporate a graph's topology as well as the attributes of its entities. Shi et al. [21] develop OnionGraph, which allows users to analyse heterogeneous graphs by aggregating nodes using their topology and attributes. Heterogeneous graphs are those which have many entities, and many ways to define edges between them. OnionGraph aggregates nodes into clusters represented by a multiple concentric circles indicating the level of the view. Users are also able to filter nodes and edges to gain a better view. This hybrid approach allows scalability up to 1M nodes while still preserving interactivity for users to analyse large data sets.

10.2 Peeling Values and Fixed Points

We provide here the notations and definitions used in this paper following the exposition given in Abello and Quelroy [1]. In particular, we introduce the concept of *fixed points of degree peeling* graphs. We use the term *network* interchangeably with *graph*. We concentrate on undirected graphs even though peeling based concepts are generalizable. We will discuss the case of weighted graphs later on.

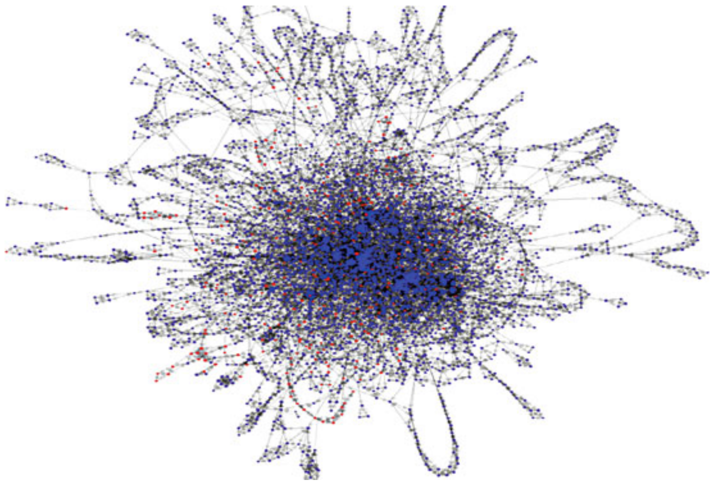


Fig. 10.1 An example of a "hairball"; each node represents an integer sequence in the OEIS and the edges correspond to cross-references given by the OEIS. The red vertices correspond to sequences that have the "hard" keyword, and the blue vertices correspond to sequences that have the "easy" keyword, as determined by the editors of the OEIS

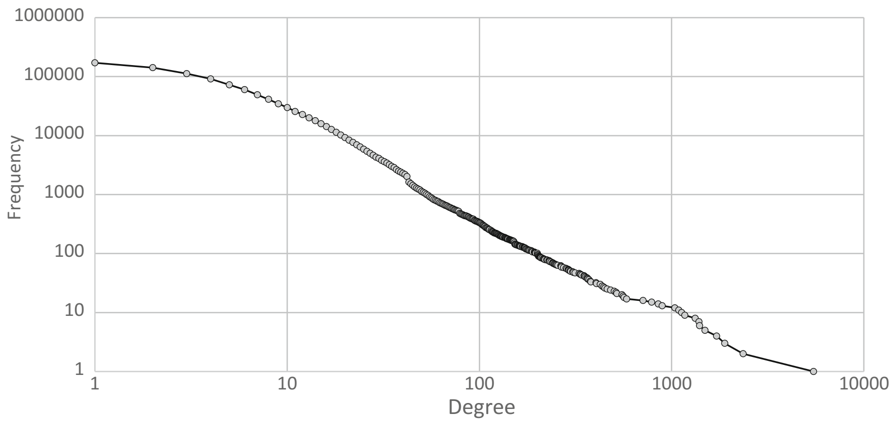


Fig. 10.2 Degree distribution of the OEIS

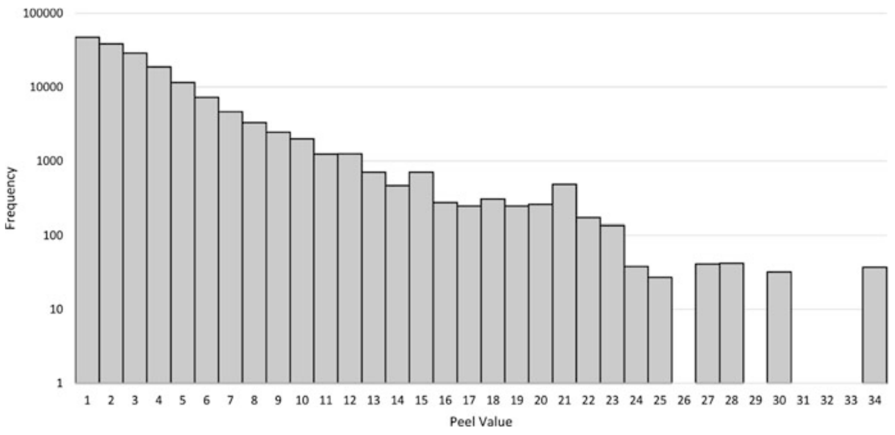


Fig. 10.3 Peel distribution of the OEIS

In this section, we use a weighted version of the reference network of the On-Line Encyclopaedia of Integer Sequences (OEIS, Sloane [22]) to illustrate the different concepts used (see Figs. 10.1, 10.2, and 10.3). The vertices correspond to sequences appearing in the OEIS database and an edge connects two sequences if one references the other and the edge weight encodes a "semantic" similarity between the endpoints. Different versions of vertex similarity can be used depending on the application. In the case of the OEIS, the edge weight corresponds to the weighted Jaccard coefficient of the words on the webpages of two sequences.

10.2.1 Definitions

We define G as an undirected graph with vertex set $V(G)$ and edge set $E(G)$. A partition of $V(G)$ is called a *vertex decomposition*. Similarly a partition of $E(G)$ is called an *edge decomposition*. The degree of a vertex u in G and the minimum degree are denoted by $d_G(u)$ and $d^-(G)$, respectively. The subgraph induced by a subset of vertices S is $G[S]$. For a given subset of edges $L \subseteq E$, the *layer* of G determined by L is the subgraph $G(L) = (V', L)$ where $V' = \{u \in V(G), \exists(u, v) \in L\}$.

Definition 10.1 (Peeling Value) The *peeling value* of a vertex $u \in V(G)$ denoted $peel_G(u)$ is the largest $i \in [1, d_G(u)]$ such that u belongs to a subgraph of G of minimum degree i. The *peeling value* of an edge $e \in E(G)$ denoted $peel_G(e)$ is the minimum peeling value of its endpoints.

In social networks, maximal induced subgraphs with peeling value at least k may be interpreted as some form of equilibrium for "a model of user engagement". In this scenario, "each player incurs a cost to remain engaged but derives a benefit proportional to the number of engaged neighbors" (see Bhawalkar et al. [8]). The peeling value was studied for random graphs by Pittel et al. [19] generated with the Erdős-Rényi model (see Erdős and Rényi [12]). The maximum peeling value of a graph (also called *degeneracy*) relates to other graph theoretical measures such as the colouring number (see Szekeres and Wilf [23]) and arboricity. Eppstein et al. [11] show that a peeling ordering of the vertices can be used to improve the running time of an algorithm for the maximal cliques problem.

Degree peeling and concepts related to it are useful in the network analysis. It has been used to evaluate the relevance of communities in co-authorship networks (see Giatsidis et al. [14]). The authors proposed a reformulation of peeling that takes into account edge weights. Some graph decompositions based on degree peeling have been used by Alvarez et al. [2] and Baur et al. [7] as an aid to provide *layered* visualizations of graphs. Some aspects of internet topology (see Carmi et al. [10]) have been addressed also in this context.

10.2.2 A Network Hierarchy

One of the interesting aspects of degree peeling is the unravelling of a network hierarchy. This hierarchy is obtained by partitioning the vertices of the network into groups according to their peeling value (in the increasing order). The group with highest peeling value is called the core of the graph. The unique group that a vertex belongs to depends not only on the number of connections it has to vertices in its group but also on its connections to vertices in upper groups.

For RAM resident graphs, Batagelj and Zaversnik [5] show that the peeling value of all vertices can be computed efficiently in $\mathcal{O}(|E(G)|)$. For graphs that do not fit in RAM, an I/O efficient external-memory algorithm that computes an approximation to the peeling values has been recently proposed by Goodrich and Pszona [15]. The *peeling value of* G, denoted $peel(G)$, is the maximum peel value of all its vertices. The peeling value of G is also called the *degeneracy* of G (see Lick and White [16]). For a graph of peeling value k, its vertices can be ordered in a sequence $(v_1, \ldots, v_n)$ called the Erdős-Hajnal sequence (see Erdő̋s and Hajnal [13]) such that there are at most k edges going from v_i to $(v_{i+1}, \ldots, v_n)$. A simple but fundamental property of peel values is that they are a local manifestation of a global graph connectivity phenomenon. The following result states this precisely.

Theorem 10.1 (Peeling Value Locality, see Montresor et al. [17]) *A vertex $u \in V$ has at least $peel_G(u)$ neighbours with a peeling value at least $peel_G(u)$ and at most $peel_G(u)$ neighbours with a peeling value at least $peel_G(u) + 1$.*

The authors of the previous theorem exploit these local relations between the peeling value of a vertex and the peeling values of its neighbours to compute peeling values by a distributed algorithm.

Definition 10.2 (Peel Decomposition) The *vertex peel decomposition* of a graph G is the partition induced by the peeling values of the vertices of G.

Definition 10.3 (Graph Core) The core of G, $core(G)$, is the maximal subset of vertices of G whose peeling value is maximum, i.e. equal to the peeling value of G.

Definition 10.4 (Local Peeling Values) Let $\mathcal{P}$ be a partition of $V(G)$. The *local peeling value* of a vertex $u \in G$ is $peel_{\mathcal{P}}(u) = peel_{G[P(u)]}(u)$ where $P(u)$ is the set in $\mathcal{P}$ that contains u. Similarly, if $\mathcal{L}$ is a partition of $E(G)$, the *local peeling value* of an edge $e \in E(G)$ is $peel_{\mathcal{P}}(e) = peel_{G(L(e))}(e)$ where $L(e)$ is the set in $\mathcal{L}$ that contains e.

Definition 10.5 (Fixed Point) A graph F is a fixed point of degree peeling k if $core(F) = V(F)$ and the peeling value of F is k. Equivalently, a graph F is a *fixed point of degree peeling* if the vertex peel decomposition of F has only one class and its peeling value is equal to its minimum degree.

Note that if F is a fixed point of degree peeling, the local peeling values of elements in F do not depend on elements with higher local peeling values. Our quest is to identify "distance based landmark" vertices that are "persistent" under "non decreasing" iterative removal of fixed points of degree peeling. Those vertices that persist as landmarks across the entire network are suggested as the most "diverse" vertices in the network. They are potential cluster centres across the peel decomposition. Their influence is spread across all the subgraphs of varying density levels to which it belongs (i.e. across fixed points of degree peeling). Among all possible edge partitions of G into fixed points of degree peeling, the one proposed in Abello-Quelroy is maximal. We denote by FP_k the class of graphs that are fixed points of degree peeling k. They are also called *strongly k-degenerate graphs* by Bauer et al. [6]. FP_k includes well-known classes of graphs. For example, the class FP_1 corresponds to forests (without isolated vertices), cliques of size n are in FP_{n-1}, k-regular graphs are in FP_k, and one can easily exhibit less obvious graph classes. For fixed points $F \in FP_k$, the peel value locality property captured by Theorem 10.1 can be re-stated as: "a vertex $u \in V(F)$ has at least k neighbours of peeling value k". The size of the maximum clique in $F \in FP_k$ is bounded above by $k + 1$. For more details about edge maximal FP_k graphs the reader is referred to Abello and Quelroy [1] and Bauer et al. [6]. Graphs generated using the Barabási-Albert model (see Barabási and Albert [4]) model with a clique of size k as seed are examples of edge maximal FP_k graphs. More generally, the construction of any "edge-maximal" FP_k graph goes as follows: from a clique of size k iteratively add $(n - k)$ vertices linked to exactly k vertices. This property indicates that the average degree of an FP_k graph with n vertices is αk with $1 \leq \alpha \leq 2$. Any k-connected subgraph or connected component of an FP_k graph is a fixed point with peeling value k. A natural question is to edge decompose efficiently any graph that does not fit in RAM into a union of fixed points of degree peeling. At the time of this writing we do not know such an efficient I/O algorithm. Moreover, we suspect that there is no such an algorithm with a constant number of passes over the data. Due to this state of affairs we concentrate in using "the easier" to compute core vertex decomposition and amplify its applicability to "large" graph exploration by introducing the notion of independent persistent landmarks across vertex graph decompositions.

10.2.3 Iterative Vertex Core Decomposition

The following Algorithm 2, from Abello and Quelroy [1], computes a vertex decomposition of G into fixed points of degree peeling. It relies on the fact that, for any graph G, $core(G)$ is a fixed point of degree peeling. After the removal of $core(G)$, the peeling value of the remaining vertices will directly drop if they were connected to the core. This operation can affect other vertices due to the iterative computation of peeling values. This means that in each obtained fixed point F of peeling value k, all vertices in F have a local peeling value lower or equal to their global peeling value in G.

Algorithm 2: Iterative vertex core decomposition of G

```
  Input: $G = (V, E)$
  Output: $\mathcal{C} = (C_1, \ldots, C_l)$, each $C_i$ is a fixed point.
1 $G' \leftarrow G$; $\mathcal{C} \leftarrow \emptyset$;
2 while $V(G') > 0$ do
3   $\mathcal{C} \leftarrow \mathcal{C} \cup \{core(G')\}$; $G' \leftarrow G'[V(G') - core(G')]$;
4 end while
5 return $\mathcal{C}$;
```

Notice that this iterative vertex core decomposition discards the connections between the different groups in the graph. This is one of the main reasons we introduce the notion of "independent vertex landmarks" next in the entire network by tracing their existence across the core decomposition. This notion provide us a tool to explore the data at different levels of granularity maintaining an overall "landmark reference set". This landmark reference set could be useful as a "network mental map" since it can be used to keep an overall view of the space (based on "far" distances) with direct access to the data "close" to each of the landmarks.

10.2.4 Iterative Peel Independent Landmarks

For our purposes, a subset of vertices L in a connected graph G is considered an l-landmark set for G if L has l vertices, L is independent and if V can be partitioned into l subsets $C_1, \ldots, C_l$ such that each C_i contains exactly an element of L and the diameter of the subgraphs induced by the C_i's is minimized.

Even though the task of computing such a landmark set, for an arbitrary graph, is NP-Complete we can adapt a classical approximation algorithm that guarantees that the obtained diameter is not more than twice the optimum. Since the bottleneck for the approximation algorithm, when the desired size of L is known a priori is the iterative identification of vertices that lie far apart from each other we opt for the following heuristic approach described next.

10.3 Independent Landmarks

When obtaining the landmarks of a graph data set, the first one selected is the centre. The centre of a graph is the vertex with minimum eccentricity, that is, it minimizes the greatest distance to the other vertices.

10.3.1 Distance Based Landmarks

A Breadth-First-Search (BFS) tree provides us with an approximation for the diameter and corresponding centre of a graph, as well as a way to cluster the vertices of the graph via "landmark" vertices. We build the BFS tree as follows: an arbitrary vertex of the graph is chosen, we construct the BFS tree from this node to all the nodes in its component, we peel the resulting tree, and reconstruct a BFS tree using the vertex or last two vertexes that were last peeled as the root. The height of this second BFS tree gives us a 2-approximation for the diameter of the original graph and one of the last peeled vertices from this second tree is our approximation to the centre. This method is well-known and finding interactive approximations for graph diameter is an interesting area of research.

10.3.2 Landmark Selection via BFS

The centre of the second described tree above is selected as the first landmark. Landmarks are then iteratively selected as the most distant leaf in a BFS tree, which is obtained starting from the set of previously selected landmarks. The distance to the most distant leaf monotonically decreases as more landmarks are added. Let the last landmark added have a distance r from the previous landmarks. The number of landmarks can be chosen as a function of the number of vertices in the data set or the density of the graph. Alternatively, the user may specify the integer value r that acts as a stopping point for the landmark-obtaining procedure. After all landmarks are selected, no vertex in the graph will be further than r hops from its nearest landmark. We can then obtain a clustering defined such that every vertex is in a cluster with its nearest landmark by BFS distance.

10.3.3 Shortest Path to Fixed Centre

Having a precomputed set of landmarks enables us to provide a quick heuristic for obtaining the shortest path from any vertex to the centre of the graph. First, we precompute the shortest path from each landmark to the centre of the graph. Then, for any selected vertex, we perform a BFS until the nearest landmark is found, and then we union the discovered path with the precomputed landmark-to-centre path.

10.4 Graph Filtering and Vertex Diversity

In this section, we indicate how to use our distance landmarks to explore network data at different scales. We also propose a measure of landmark diversity by adapting a measure of vertex diversity based on Shannon entropy proposed by Abello and Quelroy [1].

10.4.1 Graph Filtering

Peeling values can be used to filter out vertices with few connections. By associating with each vertex the first time that it appears as a landmark in the iterative core decomposition, we can keep track of the proportion of recently added vertices to any layer. Sudden proportion changes between consecutive levels are an indicator of a possible community structure.

10.4.2 Assessing Vertices Diversity

For a landmark vertex u, we can associate a *landmark profile vector* indicating those peel values for which u is selected as a landmark vertex when we iteratively delete all vertices with peel value less than that of vertex u.

Definition 10.6 For a graph G and its iterative core decomposition $\mathcal{L} = (L_1, \ldots, L_p)$, the *profile* of a vertex $u \in V(G)$ denoted $profile(u)$ is a sequence of binary variables $(l_1, \ldots, l_p)$ where each $l_i = 1$ if and only if u was selected as a landmark vertex after peeling the graph up to degree i.

Notice that the number of times a vertex u is selected as a landmark, in the iterative core decomposition, corresponds to the number of ones in the vector $profile(u)$.

Landmark vertex profiles are used next to assess its *landmark diversity*, which measures how its participation as a landmark is distributed across the different peel layers.

Definition 10.7 (Vector Majorization, Shannon Entropy, and Landmark Diversity) For a vector u in R^k, let $p(u) = (p_1(u), p_2(u), \ldots, p_k(u))$ denote the vector obtained by sorting the entries of u from largest to smallest. A vector v in R^k is said to be *majorized* by a vector u in R^k iff for $1 \le l < k$, $\sum_{j=1}^{l} p_j(v) \le \sum_{j=1}^{l} p_j(u)$ and $\sum_{j=1}^{k} p_j(v) = \sum_{j=1}^{k} p_j(u)$ (see Arnold [3]). Let $H(profile(u))$ be the Shannon entropy of the profiles normalized by the maximum peeling value T (see Shannon [20]).

$$H(profile(u)) = -\sum_{i=1}^{p} \frac{l_i}{T} \log_2 \left(\frac{l_i}{T} \right) \tag{10.1}$$

For two vertices u and v, if $profile(v)$ is majorized by $profile(u)$, then $H(profile(v)) \ge H(profile(u))$ since $H(.)$ is a Schur-concave function. Therefore, we can rank landmark vertices using the entropy of their profiles. We call this measure landmark diversity. Namely, the *diversity* of a landmark vertex u is $H(profile(u))$. The diversity of a landmark does not solely depend on its peeling value, but also on how it is positioned with respect to its distance to other landmarks across the peel decomposition. A vertex that is not part of the core of the graph can still have a bigger diversity than vertices from the core. This connection between vertex diversity, vector majorization, and Shannon Entropy was first introduced in [1].

10.5 User-Driven Exploration Mechanisms

In this section, we describe the exploration mechanisms implemented in our interface that are primarily driven by the user's actions. Any feature of our interface can be applied to any new tabs that appear as a result of a user-driven exploration. In other words, the interface can be used in a recursive fashion.

10.5.1 Egonets

The *egonet* of a vertex u is the subgraph induced by u and its neighbours. When the user clicks on the "Show Egonet" menu option for any vertex u, the egonet of u will appear in the display. In the menu option, the number in parentheses indicates the size of that vertex's neighbourhood. When a large number of vertices is added to the display, the force-directed layout takes a bit of time to settle. Therefore, when the user wishes to view an egonet with more than 128 vertices, a pop-up box will appear, confirming that the user wishes to view the egonet in the current window. Alternatively, the user can click "Open Egonet in New Window", which displays the subgraph induced by the neighbours of the vertex, excluding the vertex itself, in a new window.

10.5.2 Shortest Path Net

The user can provide an ordered subset A of vertices, and using A, we can compute a "shortest path net". Intuitively, the shortest path net H of A is an approximation to a Steiner tree for A. Initially, H only contains the first vertex of A. Then, for each remaining vertex a in A, we add a to H as well as the subgraph induced by the vertices on the shortest path from a to any existing vertex in H. This shortest path can be obtained via BFS. Once all of the vertices of A have been added, the displayed graph H is our desired approximation to a Steiner tree for A.

In our interface, the user can provide the ordered set A in two ways. One way is for the user to individually mark vertices by selecting the appropriate menu option for each desired vertex. A small red (grey) star appears next to each marked vertex. The other way is to type the identifiers of the desired vertices, separated by commas, into the textbox labelled "List to Visualize (csv)". After the user specifies A, either through individually marking or typing in the textbox, the user can click "Visualize Vertices" and the shortest path net on these vertices, in the order that they were marked or typed, is displayed in a new tab.

10.5.3 Peeling to a Shortest Path

Another way to explore large graphs is by computing the shortest path between two vertices, and then peeling the graph while preserving vertices on the path. Pages of the OEIS are tagged with different keywords to indicate some simple properties about the sequence like: it is easy to obtain more terms (easy), it is hard to obtain more terms (hard), more terms wanted for the site (more), etc. The graph is initially filtered down to those sequences having keywords of easy, hard, none, or base. A weighted shortest path is then computed between an interesting pair of vertices. The graph is peeled to level 3, with the vertices on the path (including endpoints) protected from the peeling process (Figs. 10.4, and 10.5). The resulting graph is then rendered using a force directed layout. Figures 10.6, 10.7, and 10.8 of Sect. 10.7.1 each show one end of such path layout (Figs. 10.9, 10.10, 10.11, 10.12, 10.13, 10.14, 10.15, and 10.16).

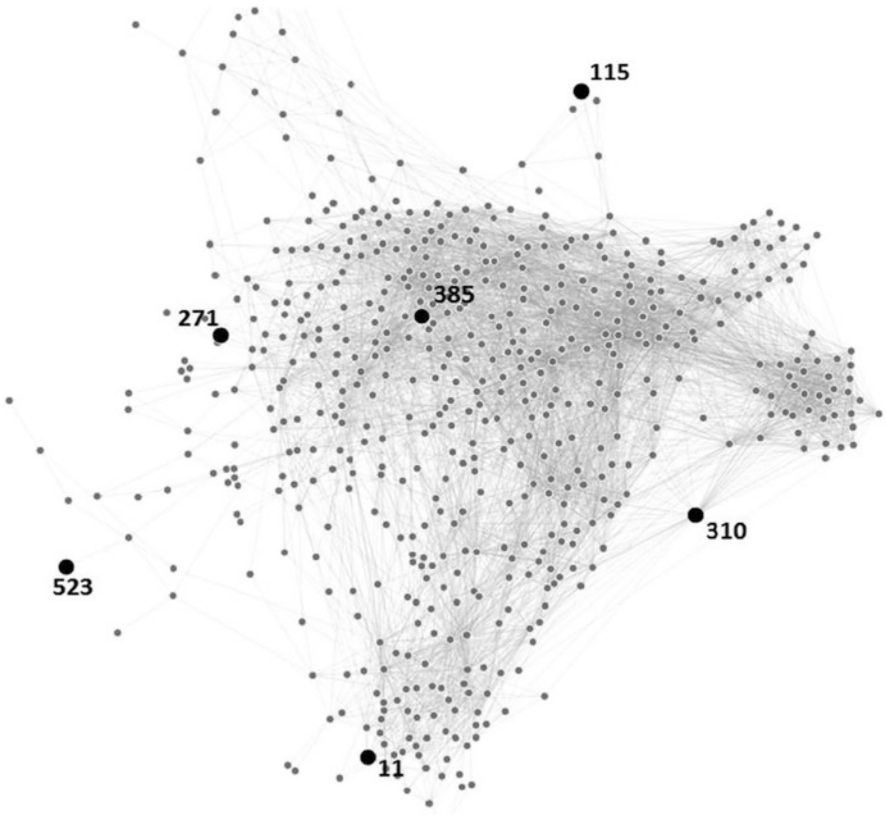


Fig. 10.4 A co-purchasing network of items on Amazon, with a few vertices highlighted

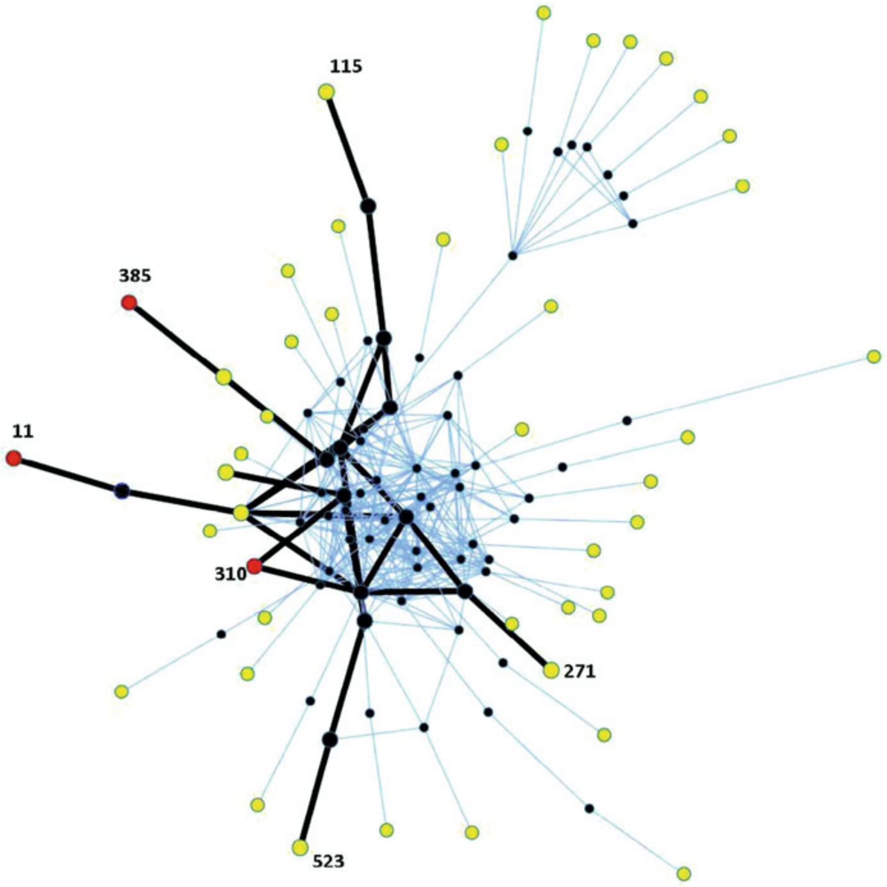
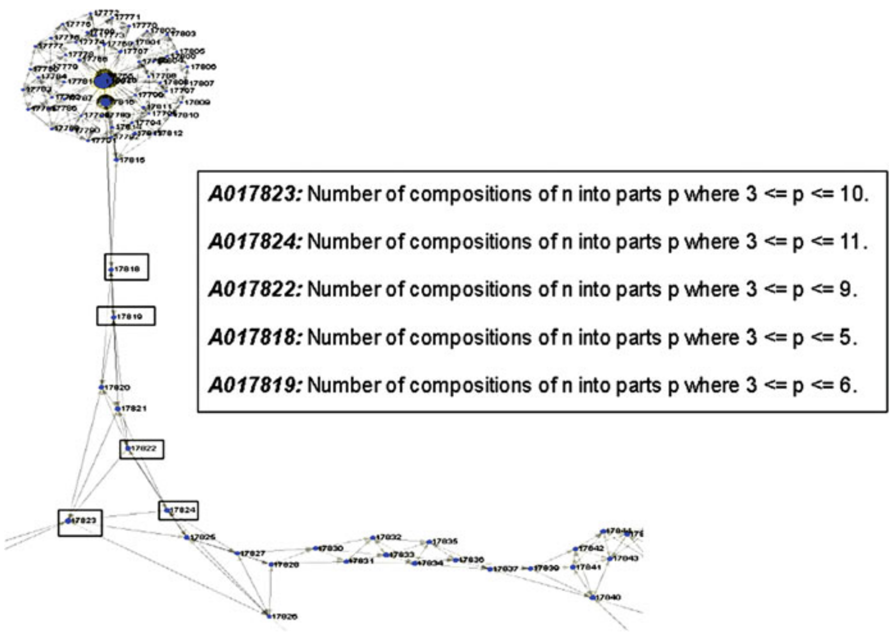


Fig. 10.5 Our computed landmark-based subgraph of the Amazon network, with the same vertices highlighted as in Fig. 10.4

10.6 Software Architecture

The software architecture supporting these exploration techniques consists of three major components: a relational database, server-side Java code, and client-side Javascript code. The relational database is provided by SQLite and its Java Database Connection (JDBC) driver. The Java code runs within the Apache Tomcat web server and provides resources via a Representative State Transfer (RESTful) API. The Javascript code leverages the Data Driven Documents (D3) library to provide the Force Directed Layout based visual results.

The SQLite database consists of two tables. The $Sequences$ table holds vertex properties and the $Cross\_Refs$ table contains edges among the vertex set. Queries against this database occur from within the Java code via the JDBC driver. Queries



A017823: Number of compositions of n into parts p where 3 <= p <= 10.

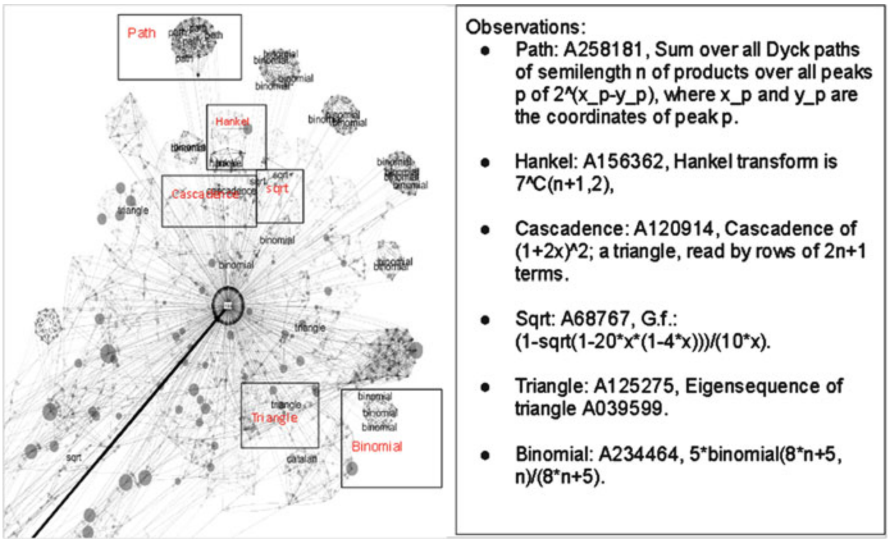
A017824: Number of compositions of n into parts p where 3 <= p <= 11.

A017822: Number of compositions of n into parts p where 3 <= p <= 9.

A017818: Number of compositions of n into parts p where 3 <= p <= 5.

A017819: Number of compositions of n into parts p where 3 <= p <= 6.

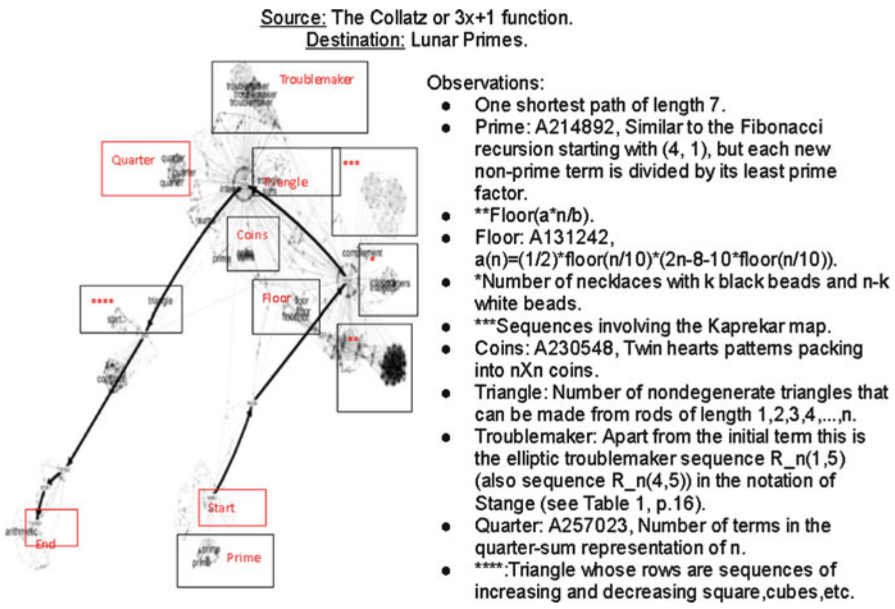
Fig. 10.6 This Figure shows a portion of the OEIS graph; we can see that vertices near each other have similar descriptions



Observations:

- Path: A258181, Sum over all Dyck paths of semilength n of products over all peaks p of 2^(x_p-y_p), where x_p and y_p are the coordinates of peak p.
- Hankel: A156362, Hankel transform is 7^C(n+1,2),
- Cascadence: A120914, Cascadence of (1+2x)^2; a triangle, read by rows of 2n+1 terms.
- Sqrt: A68767, G.f.: (1-sqrt(1-20*x*(1-4*x)))/(10*x).
- Triangle: A125275, Eigensequence of triangle A039599.
- Binomial: A234464, 5*binomial(8*n+5, n)/(8*n+5).

Fig. 10.7 The central vertex corresponds to the Catalan numbers; the vertices nearby represent integer sequences that can be naturally formed into clusters


Source: The Collatz or 3x+1 function.
Destination: Lunar Primes.

Observations:

- One shortest path of length 7.
- Prime: A214892, Similar to the Fibonacci recursion starting with (4, 1), but each new non-prime term is divided by its least prime factor.
- **Floor(a*n/b).
- Floor: A131242, a(n)=(1/2)*floor(n/10)*(2n-8-10*floor(n/10)).
- *Number of necklaces with k black beads and n-k white beads.
- ***Sequences involving the Kaprekar map.
- Coins: A230548, Twin hearts patterns packing into nXn coins.
- Triangle: Number of nondegenerate triangles that can be made from rods of length 1,2,3,4,...,n.
- Troublemaker: Apart from the initial term this is the elliptic troublemaker sequence R_n(1,5) (also sequence R_n(4,5)) in the notation of Stange (see Table 1, p.16).
- Quarter: A257023, Number of terms in the quarter-sum representation of n.
- ****:Triangle whose rows are sequences of increasing and decreasing square,cubes,etc.

Fig. 10.8 The source of this shortest path corresponds to the Collatz sequence; the destination corresponds to the lunar primes. We can see that the vertices near those on the shortest path can be naturally clustered by their descriptions

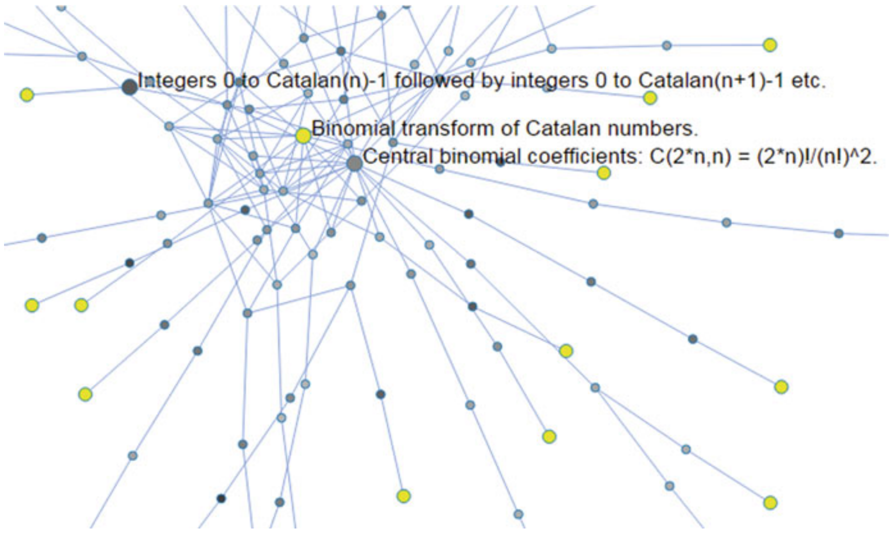

Integers 0 to Catalan(n)-1 followed by integers 0 to Catalan(n+1)-1 etc.
Binomial transform of Catalan numbers.
Central binomial coefficients: C(2*n,n) = (2*n)!/(n!)^2.

Fig. 10.9 A portion of the landmark spanning tree of the neighbourhood of sequence A000108, which are the Catalan numbers. The vertex corresponding to the sequence itself is not shown in this view, since the user knows that in the underlying space, it is adjacent to each vertex displayed here

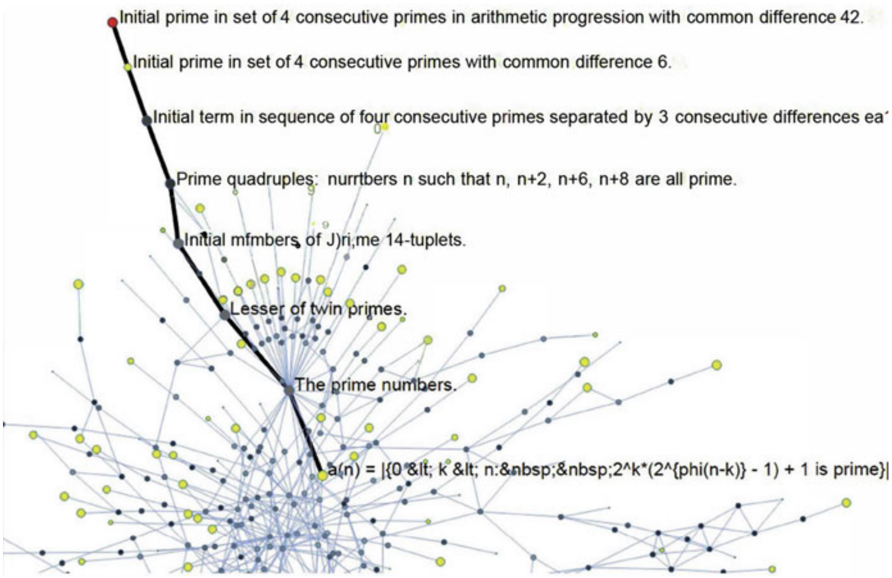

Initial prime in set of 4 consecutive primes in arithmetic progression with common difference 42.
Initial prime in set of 4 consecutive primes with common difference 6.
Initial term in sequence of four consecutive primes separated by 3 consecutive differences ea
Prime quadruples: nurrtbers n such that n, n+2, n+6, n+8 are all prime.
Initial mfmbers of J)ri,me 14-tuplets.
Lesser of twin primes.
The prime numbers.
a(n) = |{0 &lt; k &lt; n: 2^k*(2^{phi(n-k)} - 1) + 1 is prime}|

Fig. 10.10 The user can add a vertex, which will appear coloured red (grey), along with the shortest path from the added vertex to its nearest landmark, as well as the shortest path from that landmark to the centre vertex. Edges within nodes on a shortest path are highlighted black, while nodes on a shortest path are slightly larger

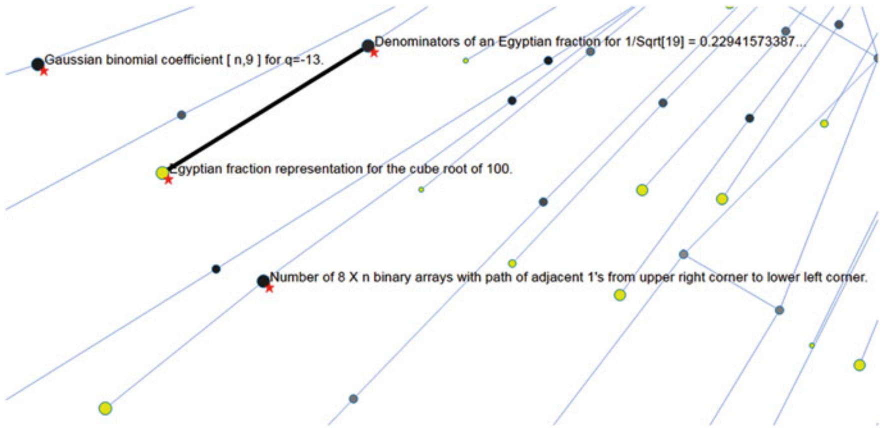

Gaussian binomial coefficient [n,9] for q=-13.
Denominators of an Egyptian fraction for 1/Sqrt[19] = 0.22941573387...
Egyptian fraction representation for the cube root of 100.
Number of 8 X n binary arrays with path of adjacent 1's from upper right corner to lower left corner.

Fig. 10.11 The user can mark a vertex by clicking on it and selecting the appropriate item from the drop-down menu. A red (grey) star is displayed next to each marked vertex

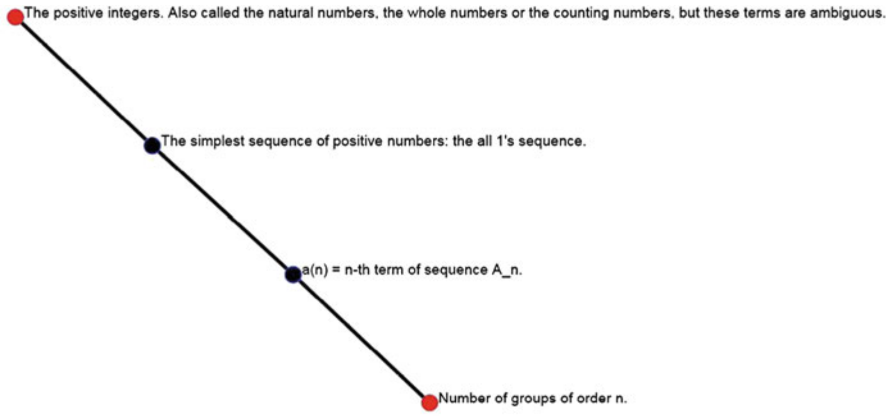

The positive integers. Also called the natural numbers, the whole numbers or the counting numbers, but these terms are ambiguous.
The simplest sequence of positive numbers: the all 1's sequence.
a(n) = n-th term of sequence A_n.
Number of groups of order n.

Fig. 10.12 The user can request to view the shortest path between any two displayed vertices. The edges in the shortest path are highlighted in black

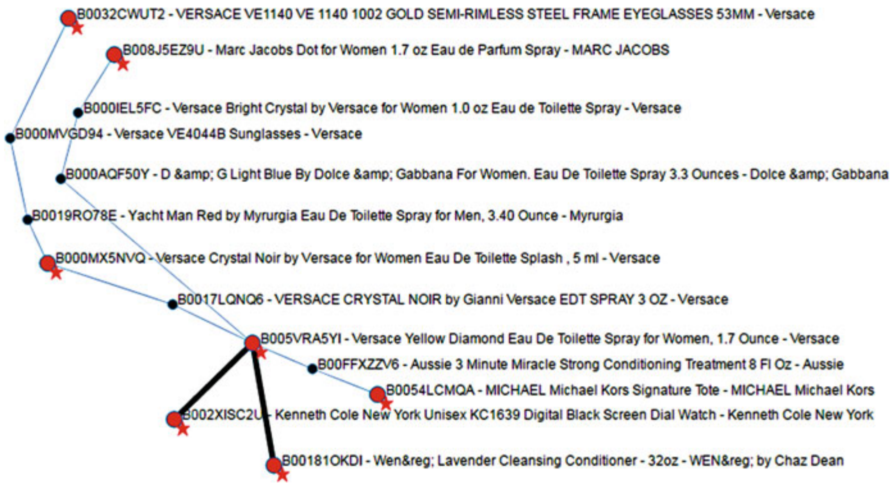

B0032CWUT2 - VERSACE VE1140 VE 1140 1002 GOLD SEMI-RIMLESS STEEL FRAME EYEGLASSES 53MM - Versace
B008J5EZ9U - Marc Jacobs Dot for Women 1.7 oz Eau de Parfum Spray - MARC JACOBS
B000IEL5FC - Versace Bright Crystal by Versace for Women 1.0 oz Eau de Toilette Spray - Versace
B000MVGD94 - Versace VE4044B Sunglasses - Versace
B000AQF50Y - D &amp; G Light Blue By Dolce &amp; Gabbana For Women. Eau De Toilette Spray 3.3 Ounces - Dolce &amp; Gabbana
B0019RO78E - Yacht Man Red by Myrurgia Eau De Toilette Spray for Men, 3.40 Ounce - Myrurgia
B000MX5NVQ - Versace Crystal Noir by Versace for Women Eau De Toilette Splash , 5 ml - Versace
B0017LQNQ6 - VERSACE CRYSTAL NOIR by Gianni Versace EDT SPRAY 3 OZ - Versace
B005VRA5YI - Versace Yellow Diamond Eau De Toilette Spray for Women, 1.7 Ounce - Versace
B00FFXZZV6 - Aussie 3 Minute Miracle Strong Conditioning Treatment 8 Fl Oz - Aussie
B0054LCMQA - MICHAEL Michael Kors Signature Tote - MICHAEL Michael Kors
B002XISC2U - Kenneth Cole New York Unisex KC1639 Digital Black Screen Dial Watch - Kenneth Cole New York
B00181OKDI - Wen&reg; Lavender Cleansing Conditioner - 32oz - WEN&reg; by Chaz Dean

Fig. 10.13 The user can select vertices to view their shortest path net, described in Sect. 10.5. A red (grey) star appears next to the selected vertices

for adjacency lists are cached using a Google Guava LoadingCache to retain graph data up to the size of main memory.

Computationally intensive tasks are handled on the server side in the Java component. It provides access to the precomputed information as well as on-the-fly results. The shortest path between two vertices is found using a BFS on the server-side graph and the path is returned to the display component. When adding a single vertex, the code will BFS from there to the nearest landmark, combine it with the path from the landmark to the centre, and return. When acting on a set of marked vertices, the code will find the shortest path between the first pair and add

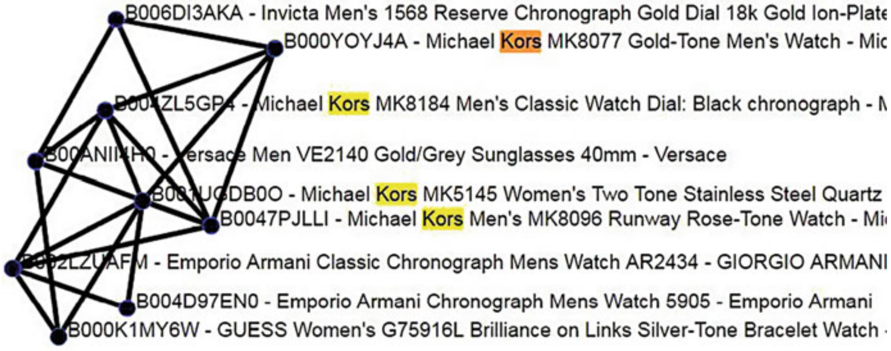

B006DI3AKA - Invicta Men's 1568 Reserve Chronograph Gold Dial 18k Gold Ion-Plate
B000YOYJ4A - Michael Kors MK8077 Gold-Tone Men's Watch - Mic
B004ZL5GP4 - Michael Kors MK8184 Men's Classic Watch Dial: Black chronograph - M
B00ANII4H0 - Versace Men VE2140 Gold/Grey Sunglasses 40mm - Versace
B003UGDB0O - Michael Kors MK5145 Women's Two Tone Stainless Steel Quartz
B0047PJLLI - Michael Kors Men's MK8096 Runway Rose-Tone Watch - Mi
B002LZUAFM - Emporio Armani Classic Chronograph Mens Watch AR2434 - GIORGIO ARMANI
B004D97EN0 - Emporio Armani Chronograph Mens Watch 5905 - Emporio Armani
B000K1MY6W - GUESS Women's G75916L Brilliance on Links Silver-Tone Bracelet Watch

Fig. 10.14 A portion of the subgraph induced by the vertices with peel value 5. We can see that the Michael Kors products form a clique, and the two Emporio Armani products are connected as well

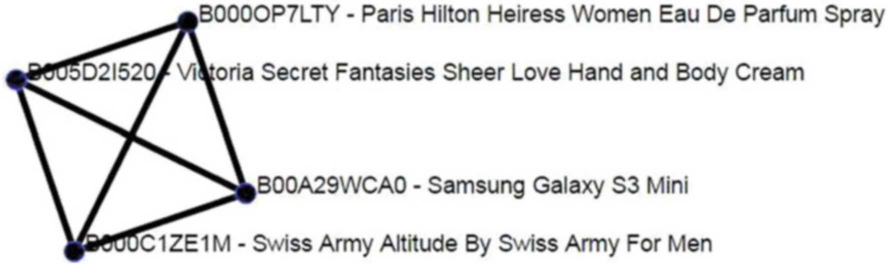

B000OP7LTY - Paris Hilton Heiress Women Eau De Parfum Spray
B005D2I520 - Victoria Secret Fantasies Sheer Love Hand and Body Cream
B00A29WCA0 - Samsung Galaxy S3 Mini
B000C1ZE1M - Swiss Army Altitude By Swiss Army For Men

Fig. 10.15 The subgraph induced by vertices with peel value 25. These four items include two perfume sprays, a body cream, and a Samsung Galaxy cell phone

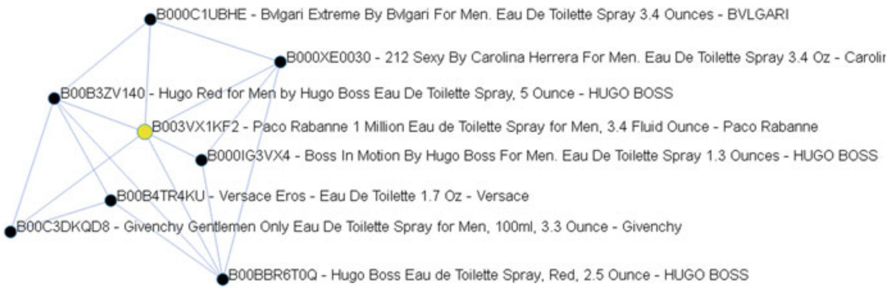

B000C1UBHE - Bvlgari Extreme By Bvlgari For Men. Eau De Toilette Spray 3.4 Ounces - BVLGARI
B000XE0030 - 212 Sexy By Carolina Herrera For Men. Eau De Toilette Spray 3.4 Oz - Carolir
B00B3ZV140 - Hugo Red for Men by Hugo Boss Eau De Toilette Spray, 5 Ounce - HUGO BOSS
B003VX1KF2 - Paco Rabanne 1 Million Eau de Toilette Spray for Men, 3.4 Fluid Ounce - Paco Rabanne
B000IG3VX4 - Boss In Motion By Hugo Boss For Men. Eau De Toilette Spray 1.3 Ounces - HUGO BOSS
B00B4TR4KU - Versace Eros - Eau De Toilette 1.7 Oz - Versace
B00C3DKQD8 - Givenchy Gentlemen Only Eau De Toilette Spray for Men, 100ml, 3.3 Ounce - Givenchy
B00BBR6T0Q - Hugo Boss Eau de Toilette Spray, Red, 2.5 Ounce - HUGO BOSS

Fig. 10.16 The egonet of product B00AKFFF2I, Davidoff The Game Eau de Toilette Spray for Men, from an Amazon co-purchasing network. We can see that the neighbouring products are very similar to each other and the original product

the path vertices to a set. For each additional vertex, a shortest path will be found to a vertex already in the set and the path will be added. The code also supports simpler operations like accessing the egonet of a vertex and getting vertex descriptions from the database. This enables the user to quickly and conveniently retrieve information about any vertex.

The Javascript-based browser application performs the iterative force directed layout computations and provides the various modes of user interactivity previously described. The code merges arbitrary graphs on the same vertex set together to support the continual addition of user-requested graph components. The vertices and edges are then matched up with Scalable Vector Graphics (SVG) circles and paths which to be displayed on the screen. The coordinates of these components are managed by the Force Directed Layout in order to make the graph visually appealing.

10.7 Tour of the Interface

In this section, we provide a tour of the exploration mechanisms (Fig. 10.17).

The exploration begins by hovering over the blue box labelled "Menu" in the top-left of the display. The user can select a peel value and click "Show Tree". This brings up the shortest-path tree of the largest connected component, rooted

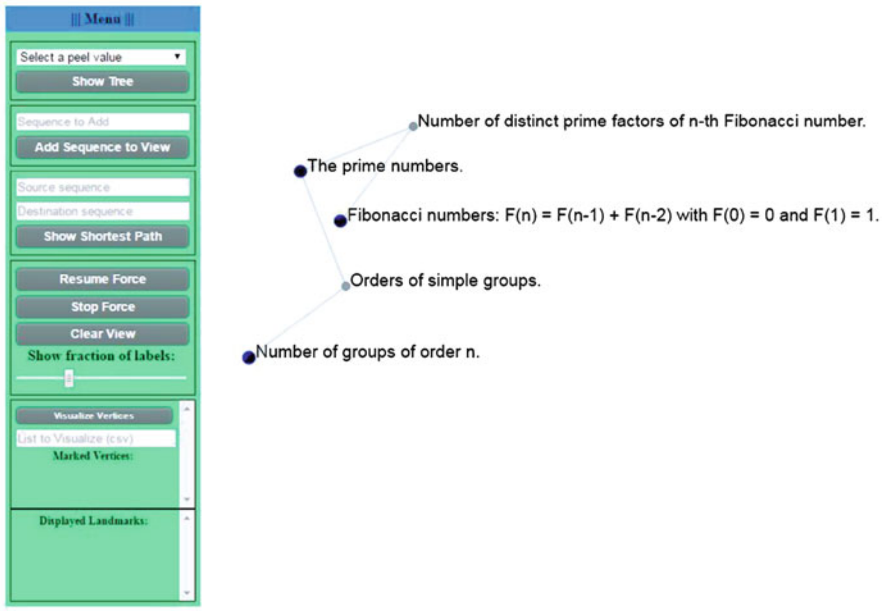


Fig. 10.17 An overview of the exploration tool interface

at the centre of the component, whose leaves are the pre-computed landmarks. The landmarks, which include the centre, are coloured yellow (light grey) while the path-vertices are grey. The font size of the labels of the vertices, as well as the size of the vertices, can be adjusted using the controls at the top-right of the display. By adjusting the sliding bar in the main menu, the user can control the number of labels being displayed.

Now, in the "Sequence to Add" box, the user may enter the sequence number of a sequence that interests them. By clicking "Add Sequence to View", the user can see the entered vertex appear in red (grey), along with the shortest path to the nearest landmark, joined with the shortest path from the nearest landmark to the root of the tree. The edges of this path are highlighted in black.

The user may click on any vertex and a drop-down menu will appear. Clicking "Open OEIS Page" will open the OEIS page corresponding to the sequence that vertex represents. Clicking "Show Egonet" will add the subgraph induced by the neighbourhood of the vertex to the display. Clicking "Open Egonet in New Window" will show the subgraph induced by the neighbourhood of the vertex in a new tab. In this case, the vertex that was clicked on will not be shown; it is understood that each neighbour is adjacent to the clicked vertex. The user can set any vertex as a source or destination by clicking the respective menu options, and then click "Show Shortest Path" in the menu option. (Alternatively, the user may manually enter the sequence numbers in their respective boxes in the main menu). Finally, the user can click "Mark Vertex". This will be recorded under the "Marked Vertices" section of the main menu, which the user can consult for further exploration.

Furthermore, the user can click the "Visualize Vertices" button, which is located above the list of marked vertices. This opens, in a new tab, the iterative-shortest-path view of the marked vertices. The description of how this graph is obtained is provided in Sect. 10.3.3. Since the order of the vertices affects the topology of the graph, we perform the iterative shortest path algorithm on the vertices as they are listed in the list of marked vertices. Alternatively, the user may enter the list of vertices to visualize in the textbox labelled "List to Visualize (csv)" as a comma-separated string.

10.7.1 Sample Results

In this section, we provide some images of sample results from our exploration mechanism applied to the On-Line Encyclopaedia of Integer Sequences (OEIS).

Due to the approximate nature of our centre-finding heuristic, multiple vertices could act as the centre of the graph. The results are shown below (Table 10.1).

The subgraph induced by the vertices of the Amazon network with peel value 1 consists of 54 vertices and 1 edge. This may be an artefact of the underlying graph structure. The Amazon network we work with only represents a small portion of the

Table 10.1 Each sequence has a frequency corresponding to the number of times it is selected as a centre, using our BFS-based centre-finding heuristic

Sequence A-number	Frequency as centre
A142	167761
A115729	49181
A108	28441
A203	27031
A94216	25191
A10	21661
A59797	21611
A129178	21581
A7318	19651
A1045	16421
A239839	15891
A66121	13681
A114337	12291
A720	11331
A166469	10281
A2275	9691
A242248	8851
A2808	7641
A103678	6941
A174408	6931

The left column contains the A-number of those sequences with the highest frequencies, and the right column contains the corresponding frequencies

entire underlying network, so this may explain the sparsity of this graph. In general, the subgraph induced by the vertices with peel value 1 is a forest.

On the other hand, the subgraph induced by the vertices of the Amazon network with peel value 27 (the highest in the network) is very dense. It contains 30 vertices and 429 edges, which is only $\binom{30}{2} - 429 = 435 - 429 = 6$ edges away from being a complete subgraph. Furthermore, every node in this subgraph corresponds to a Victoria's Secret product.

10.8 Conclusions

This work offers some basic support for exploratory graph data mining. When a computer user wants to wander around a topology in order to discover interesting structures, the user is aided by receiving interactive feedback from the computer about his/her exploration routes. The suggested exploration mechanisms include a novel notion of shortest path nets interconnecting a set of iteratively computed global landmarks. Local exploration is driven by egonets centred around vertexes in the user derived shortest path nets. We illustrate the use of our mechanisms in the exploration of the On-Line Encyclopaedia of Integer Sequences (OEIS) and a network of co-purchased items from Amazon.

Acknowledgements We would like to thank the 2015 DIMACS REU program supported by the NSF. Continuation of this work has been partially supported by NSF IIS 1563816 and 1563971 grants titled Human-computer Graph Exploration and Tele-discovery, 2017–2020. As a result of this cooperation, the first author has incorporated the "Sparse Net Exploration" into a very large graph exploration system called "Atlas" whose description appears in the additional reference [24].

References

1. Abello, J., Quelroy, F.: Network decompositions into fixed points of degree peeling. Social Networks Analysis and Mining pp. 4–19 (2014)
2. Alvarez-Hamelin, J., Dall Asta, L., Barrat, A., Vespignani, A.: Large scale networks fingerprinting and visualization using the k-core decomposition. Advances in neural information processing systems **18**, 41 (2006)
3. Arnold, B.C.: Majorization and the Lorenz order: a brief introduction, vol. 43. Springer-Verlag Berlin (1987)
4. Barabási, A., Albert, R.: Emergence of scaling in random networks. science **286**(5439), 509–512 (1999)
5. Batagelj, V., Zaversnik, M.: An o(m) algorithm for cores decomposition of networks. arXiv preprint cs/0310049 (2003)
6. Bauer, R., Krug, M., Wagner, D.: Enumerating and generating labeled k-degenerate graphs. In: 7th Workshop on Analytic Algorithmics and Combinatorics (ANALCO), pp. 90–98 (2010)
7. Baur, M., Brandes, U., Gaertler, M., Wagner, D.: Drawing the as graph in 2.5 dimensions. In: Graph Drawing, pp. 43–48. Springer (2005)
8. Bhawalkar, K., Kleinberg, J., Lewi, K., Roughgarden, T., Sharma, A.: Preventing unraveling in social networks: the anchored k-core problem. Automata, Languages, and Programming pp. 440–451 (2012)
9. Bikakis, N., Liagouris, J., Kromida, M., Papastefanatos, G., Sellis, T.: Towards scalable visual exploration of very large rdf graphs. In: The Semantic Web: ESWC 2015 Satellite Events, pp. 9–13. Springer (2015)
10. Carmi, S., Havlin, S., Kirkpatrick, S., Shavitt, Y., Shir, E.: A model of internet topology using k-shell decomposition. Proceedings of the National Academy of Sciences **104**(27), 11,150–11,154 (2007)
11. Eppstein, D., Löffler, M., Strash, D.: Listing all maximal cliques in sparse graphs in near-optimal time. CoRR **abs/1006.5440** (2010)
12. Erdős, P., Rényi, A.: On the evolution of random graphs. Publ. Math. Inst. Hungar. Acad. Sci **5**, 17–61 (1960)
13. Erdős, P., Hajnal, A.: On chromatic number of graphs and set-systems. Acta Mathematica Hungarica **17**(1), 61–99 (1966)
14. Giatsidis, C., Thilikos, D., Vazirgiannis, M.: Evaluating cooperation in communities with the k-core structure. In: Advances in Social Networks Analysis and Mining (ASONAM), 2011 International Conference on, pp. 87–93. IEEE (2011)
15. Goodrich, M.T., Pszona, P.: External-memory network analysis algorithms for naturally sparse graphs. CoRR **abs/1106.6336** (2011)
16. Lick, D.R., White, A.T.: k-degenerate graphs. Canad. J. Math **22**, 1082–1096 (1970)
17. Montresor, A., Pellegrini, F.D., Miorandi, D.: Distributed k-core decomposition. CoRR **abs/1103.5320** (2011)

18. Nachmanson, L., Prutkin, R., Lee, B., Riche, N.H., Holroyd, A.E., Chen, X.: Graphmaps: Browsing large graphs as interactive maps. In: Graph Drawing and Network Visualization, pp. 3–15. Springer (2015)
19. Pittel, B., Spencer, J., Wormald, N., et al.: Sudden emergence of a giant k-core in a random graph. Journal of combinatorial theory. Series B **67**(1), 111–151 (1996)
20. Shannon, C.E.: A mathematical theory of communication. ACM SIGMOBILE Mobile Computing and Communications Review **5**(1), 3–55 (2001)
21. Shi, L., Liao, Q., Tong, H., Hu, Y., Zhao, Y., Lin, C.: Hierarchical focus+ context heterogeneous network visualization. In: Visualization Symposium (PacificVis), 2014 IEEE Pacific, pp. 89–96. IEEE (2014)
22. ed. Sloane, N.J.A.: The on-line encyclopedia of integer sequences. URL http://oeis.org
23. Szekeres, G., Wilf, H.S.: An inequality for the chromatic number of a graph. Journal of Combinatorial Theory **4**(1), 1–3 (1968)
24. Abello, J., Hohman F., Bezzam V., Chau D.H.: Atlas – Local Graph Exploration in a Global Context (2019), In: Proceedings of the ACM International Conference on Intelligent User Interfaces, Los Angeles, CA, ACM (2019)

Chapter 11
Network-Based Models for Social Recommender Systems

Antonia Godoy-Lorite, Roger Guimerà, and Marta Sales-Pardo

Abstract With the overwhelming online products available in recent years, there is an increasing need to filter and deliver relevant personalized advice for users. Recommender systems solve this problem by modelling and predicting individual preferences for a great variety of items such as movies, books or research articles. In this chapter, we explore rigorous network-based models that outperform leading approaches for recommendation. The network models we consider are based on the explicit assumption that there are groups of individuals and of items, and that the preferences of an individual for an item are determined only by their group memberships. The accurate prediction of individual user preferences over items can be accomplished by different methodologies, such as Monte Carlo sampling or Expectation-Maximization methods, the latter resulting in a scalable algorithm which is suitable for large datasets.

Keywords Human preferences · Prediction · Matrix factorization · Modelling · Social marketing

A. Godoy-Lorite (✉)
Department of Mathematics, Imperial College London, London, UK
e-mail: a.godoy-lorite@imperial.ac.uk

R. Guimerà
Institució Catalana de Recerca i Estudis Avançats (ICREA), Barcelona, Catalonia, Spain

Departament d'Enginyeria, Química, Universitat Rovira i Virgili, Tarragona, Catalonia, Spain
e-mail: roger.guimera@urv.cat

M. Sales-Pardo
Departament d'Enginyeria Química, Universitat Rovira i Virgili, Tarragona, Spain
e-mail: marta.sales@urv.cat

P. Moscato, N. J. de Vries (eds.), *Business and Consumer Analytics: New Ideas*,
https://doi.org/10.1007/978-3-030-06222-4_11

11.1 Introduction

The internet has changed the way business is done and how products are advertised, sold and distributed [1, 2]. Now we are a click away from an ever increasing array of products [3]. However, the availability of so many choices puts a stress on the customer who often has no time to browse over endless online product catalogues. As an illustration, Netflix has available around 5000 movies only in the United States, iTunes more than 37 million songs and Amazon up to 32 million books in different formats; if we were to spend 0.5 seconds per item, it would take us approximately 40 minutes to browse the whole catalogue in Netflix and over 200 full days to go through the whole catalogue of songs and books in iTunes and Amazon. The platforms that have best adapted to this situation are those that efficiently recommend items that fit personal preferences.

Recommender systems are algorithms precisely designed to predict user's preferences over a variable amount of items. A popular event that boosted research in recommender systems was the Netflix contest (2006–2009) [4–6]. Netflix sponsored a competition to improve the accuracy of their recommendation algorithm at the time, offering $1,000,000 to the best performing team. This competition captured the attention of researchers on the topic and improved significantly the state-of-the-art algorithms and even resulted in the creation of companies (for instance, Gravity R&D or 4-Tell Inc. [7]) that played a major role in boosting e-commerce. The increase in the volume of online business coupled to the availability of data on online purchases of products by users has in recent years enhanced the interest in recommender systems, both in the private and academic sectors.

Currently, the main strategies for making social recommendations are content-based approaches, collaborative filtering and hybrid approaches [8]. Content-based approaches use available metadata on users or items such as demographics, overall top selling items, past buying habit of users or item reviews to guess user preferences. On the other hand, collaborative filtering (CF) methods are based on the plausible expectation that similar users relate to similar objects in a similar manner, i.e., they purchase similar items and rate the same item similarly. Hybrid methods aim at combining both approaches.

Importantly, CF approaches are in general more accurate at predicting user preferences than content-based approaches. Typically datasets available for recommendation are *sparse*—most users rate just a few items (<10 items) and most of the items have been rated only by a few users, which makes it hard for content-based methods to make good predictions. In contrast, CF algorithms have successfully addressed this problem by exploiting known preferences of like-minded users to provide item recommendations or predictions.

A major problem recommender systems face is the need to provide recommendations in a reasonably short amount of time. Taking into account that available datasets comprise millions of user-item ratings (and that is a small fraction of the data in a real industrial setting), the scalability of the algorithm with the number of observations is critical. Specifically, if the run-time of an algorithm scales linearly in the number of observed ratings R, the time needed to obtain a recommendation if $R = 10{,}000$ is $\propto 10^5$, whereas if an algorithm scales quadratically with R, then the time needed to obtain a recommendation for $R = 10{,}000$ ratings will be $\propto 10^{10}$. As a good rule of thumb, linear (or sub-linear) CF algorithms will be good candidates to deal with the large and sparse datasets in an industrial set-up.

In this chapter, we focus on collaborative filtering models suitable for large sparse datasets. Specifically, we show how block model approaches to model a network of user-item ratings is superior to state-of-the-art approaches such as the Item-Item and Matrix Factorization approaches [9–12].

This chapter is organized as follows. In Sect. 11.2 we introduce the network framework for the recommendation problem, fundamental concepts of inference and the use of Stochastic Block Models to make inference on network data. In Sect. 11.3 we present the network-based approaches for recommender systems, the bipartite Stochastic Block Model recommender (SBM) and the Mixed-Membership Stochastic Block Model recommender (MMSBM). In Sect. 11.4 we give an overview of some of the most successful collaborative filtering approaches, one being the Item-Item model and two Matrix Factorization approaches: the "classical" Matrix Factorization (MF) and the Mixed-Membership Matrix Factorization(MMMF), which we use as benchmark algorithms. In Sect. 11.5 we analyse and compare the predictive power of those algorithms and provide a practical guide to run the network-based algorithms with real datasets. To conclude, in Sect. 11.6 we provide overall discussion of the chapter.

11.2 Network Approach to Recommender Systems

Formally, the recommendation problem is the following. We have an observation $R^{\mathcal{O}}$ consisting of a collection of ratings R_{ij} of users (i) to items (j) (e.g., ratings of users to movies or books). Ratings are on a fixed discrete scale S, so that each observed rating r_{ij} takes a value within this scale (e.g., in a 5 point scale $S = \{1, 2, 3, 4, 5\}$). This observation is sparse so that out of a group of N users and M items we only observe a small fraction of the $N \times M$ possible ratings. The goal is then to predict the values of a set of query/unobserved ratings R^{T}. In the recommender systems literature, the observation $R^{\mathcal{O}}$ is called the training set, while the query set R^{T} is called the test set. This is because recommendation algorithms use the training dataset to train the algorithm and obtain the model parameters and the test set to assess the accuracy of the trained algorithm at making predictions for unobserved ratings. Note that being able to make accurate predictions on unobserved ratings is the first and necessary step towards being able to make suggestions of new items to users based on the rating predictions over those items.

Formally, this problem can be mapped into a problem of predicting unobserved edge values in a network. Specifically, since ratings occur between two different types of nodes (users and items), we can represent $R^{\mathcal{O}}$ as a bipartite network (see Fig. 11.2a). In this network we have an edge connecting each user with all the items she has provided a rating for in the observation. Importantly, within this representation edges can take a value within the scale of ratings S. The problem of predicting ratings within the unobserved query set R^{T} then becomes that of predicting values of unobserved edges within this network.

Here, we focus on estimating the probability that a specific unobserved edge r_{ij} takes value r given the observed data $R^{\mathcal{O}}$, which formally is expressed as $p(r_{ij} = r|R^{\mathcal{O}})$. To do that we use a Bayesian approach to perform inference on network data. In what follows, we introduce the basic concepts of Bayesian inference and the Stochastic Block Model, a general class of generative models suitable for network data.

11.2.1 Inference on Complex Networks Based on Stochastic Block Models

Let us assume our observed data is $R^{\mathcal{O}}$ and we want to know the probability that a certain variable X (for instance, a rating) takes values x conditioned on the observed data, that is $p(X = x|R^{\mathcal{O}})$. If we consider an ensemble of generative models $\mathcal{M}$ for our observed data, we can express $p(X = x|R^{\mathcal{O}})$ as

$$P(X = x|R^{\mathcal{O}}) = \int_{\mathcal{M}} p(X = x|M)\, p(\mathcal{M}|R^{\mathcal{O}})\, dM, \tag{11.1}$$

where $p(X = x|M)$ is the probability of variable X being equal to x given model M (X is, for example, the rating that user u gives item i), and $p(M|R^{\mathcal{O}})$ is the plausibility of model M given the observation $R^{\mathcal{O}}$. Using Bayes' theorem we can rewrite Eq. (11.1) as,

$$P(X|R^{\mathcal{O}}) = \frac{\int_{\mathcal{M}} p(X|M)\, p(R^{\mathcal{O}}|M)\, p(M)\, dM}{p(R^{\mathcal{O}})}, \tag{11.2}$$

where $p(R^{\mathcal{O}}|M)$ is the probability that model M gives rise to the observed data $R^{\mathcal{O}}$, also called likelihood, and $p(M)$ is the prior probability that model M is the correct one, also called the prior. Importantly, the accuracy of the predictions depends strongly on the ability of some of the models in the family of models in $\mathcal{M}$ to describe the observed data.

In our case we consider the family of Stochastic Block Models (SBM) [13–15] as generative models. SBMs are based on the simple assumption that there are groups of nodes and that nodes within a group have similar connectivity patterns. Within this class of models, the probability of two nodes being connected only depends on the groups to which the nodes belong. Formally, a SBM $\mathcal{M} = (P, \mathbf{Q})$ is then completely determined by the partition P of nodes into groups and the matrix $\mathbf{Q}$ of connection probabilities between pairs of nodes belonging to pairs of groups, so that Eq. (11.2) can be rewritten as

$$p_{\mathrm{SBM}}(X = x | R^{\mathcal{O}}) = \frac{\sum_{P \in \mathcal{P}} \int_{[0,1]^G} \; p(X = x | P, \mathbf{Q}) \; p_{SBM}(R^{\mathcal{O}} | P, \mathbf{Q}) \; p(P, \mathbf{Q}) \; d\mathbf{Q}}{p(R^{\mathcal{O}})}, \tag{11.3}$$

where $\mathcal{P}$ is the space of all possible partitions of nodes into groups and G is the total number of pairs of groups of nodes.

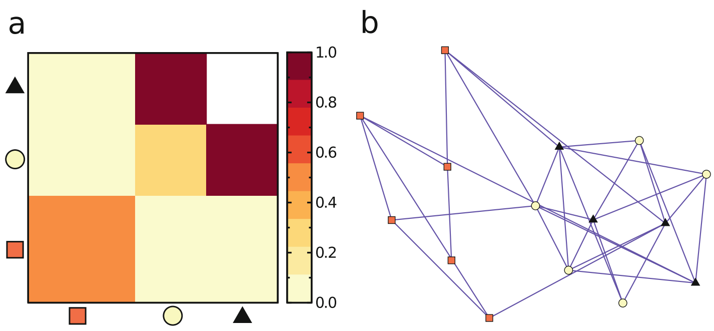


Fig. 11.1 Stochastic Block Models: A stochastic block model is fully specified by a partition of nodes into groups P and a connection probability matrix $\mathbf{Q}$. Each element $Q_{\alpha\beta}$ in the $\mathbf{Q}$ matrix represents the probability that a node in group α connects to a node in group β. (**a**) An example of a $\mathbf{Q}$ matrix of connection probabilities. We consider three groups of nodes comprising 4 (triangles), 5 (circles) and 6 (squares) nodes. We colour matrix elements according to their value following the colour bar on the right hand side. (**b**) A realization of the model in panel (**a**)

SBMs are suitable models to describe complex networks because they are versatile enough to capture the large variety of connectivity patterns observed in real networks (see Fig. 11.1 for an illustration). For instance, many real-world networks have been found to have a modular or assortative structure in which nodes within the same group (also called module or community) are more likely to be connected to nodes within the same group than to nodes in other groups [16–19]. A SBM with $Q_{\alpha\alpha} \gg Q_{\alpha\beta} \, \forall \alpha, \beta$ would generate networks with such structure. Interestingly, SBMs can also depict other connectivity patterns such as disassortative patterns in which nodes are more likely to connect to nodes in other groups or patterns that define distinct topological roles such as hubs and peripheral nodes in core-periphery structures [18, 20, 21]. Importantly, this family of models can be extended to directed [22], and weighted networks [23, 24].

11.3 Modelling Ratings Using Stochastic Block Models

In this section we will consider two network-based models for recommender systems: the simple bipartite Stochastic Block Model (SBM) [14] and the Mixed-Membership Bipartite Stochastic Block Model (MMSBM) [15]. In both models, there are different groups. The difference between these models is that while in the bipartite SBM each user and item belong solely to one group, in the MMSBM users (and items) have a certain probability of belonging to each group of users (items). Importantly, this fact allows us to describe the network of ratings using fewer groups of users and items and to implement more efficient inference algorithms.

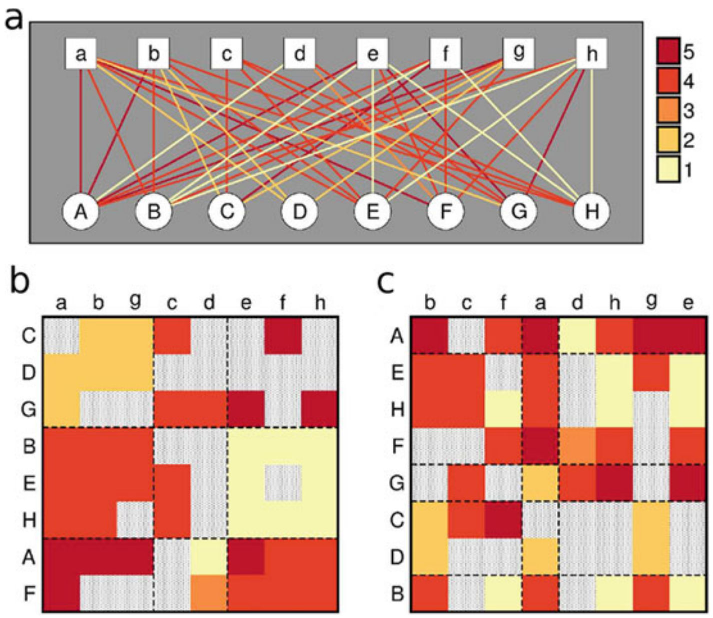


Fig. 11.2 Bipartite SBM for recommendation: (**a**) Eight users labelled A–H rate movies, labelled a–h, as indicated by the colours of the links. (**b**–**c**) Matrix representation of the ratings; grey elements represent unobserved ratings. Different partitions of the nodes into groups (indicated by the dashed lines) provide different explanations for the observed ratings. The partition in (**b**) has a high explanatory power because ratings in each pair of user-item groups are very homogeneous. For example, it seems plausible that user C would rate item a with a 2, given that all users in the same group as C rate 2 all items in the same group as a. Conversely, the partition in **c** has very little explanatory power. The predictions of partition (**b**) contribute much more than those of partition (**c**) to the inference of unobserved ratings. *Reprinted from Guimerà R. et al. Predicting Human Preferences Using the Block Structure of Complex Social Networks. PLOS ONE 7(9):e44620, under the Creative Commons Attribution (CC BY) license at https://creativecommons.org/licenses/by/4.0/*

11.3.1 Predictions Based on the Bipartite SBM Recommender System

Our inference problem is to estimate the probability $p(r_{ui} = r|R^{\mathcal{O}})$ that the unobserved rating of item i by user u is $r_{ui} = r$, given the observation $R^{\mathcal{O}}$. Hence, by setting the observable x in Eq. (11.2) to $X = r_{ui}$ we obtain,

$$p(r_{ui} = r|R^{\mathcal{O}}) = \frac{\int_M \; p(r_{ui} = r|M) \; P(R^{\mathcal{O}}|M) \; p(M) \, dM}{\int_{\mathcal{M}'} \; p(R^{\mathcal{O}}|M')p(M') \, dM'} \; . \tag{11.4}$$

where $p(r_{ui} = r|M)$ is the probability that $r_{ui} = r$ if the ratings were actually generated using model M, and $p(R^{\mathcal{O}}|M)$ is the probability of model M generating the observed ratings $R^{\mathcal{O}}$ or likelihood.

As previously mentioned, we will use the family of Stochastic Block Models $\mathcal{M}_{SBM}$ to describe the observed ratings [13, 21, 25]. In the bipartite SBM users and items are partitioned into two different and independent sets of groups. Therefore, $\mathbf{Q}$ is a $g_u \times g_i$ rectangular matrix, with g_u and g_i being the number of user and item groups, respectively. Additionally, because in our network (ratings) edges can take $|S|$ different values, we have one such matrix for each value of r. Therefore, the probability that the rating of user u to item i is equal to $r_{ui} = r$ depends exclusively on the groups σ_u and σ_i to which user u and item i, respectively, belong, so that

$$p(r_{ui} = r) = Q^r_{\sigma_u \sigma_i} \, . \tag{11.5}$$

Because in the recommender system the possible rating values for a given edge are exclusive, we have the following constraint $\sum_r Q^r_{\sigma_u \sigma_i} = 1$ for each (user, item) pair.

Note that in the bipartite SBM, ratings are considered as independent categories without assuming that the distance between ratings is linear (that is, that $r = 3$ is as far from $r = 4$, as $r = 4$ is from $r = 5$). This poses an advantage over other approaches which assume linearity in the distances between ratings, since users have been found not to perceive equal differences between adjacent ratings (i.e., $r = 4$ and $r = 5$ might be perceived as closer in rating space than $r = 3$ and $r = 4$ [26]).

Assuming a flat prior over models $p(M) = \text{const.}$, the integral in Eq. (11.4) over all possible values of $Q_{\alpha\beta}$ can be carried out analytically, so that we obtain

$$p_{SBM}(r_{ui} = r|R^{\mathcal{O}}) = \frac{1}{\mathcal{Z}} \sum_{P_U \in \mathcal{P}_U, P_I \in \mathcal{P}_I} \left(\frac{n^r_{\sigma_u \sigma_i} + 1}{n_{\sigma_u \sigma_i} + |S|} \right) e^{-\mathcal{H}(P_U, P_I)} \tag{11.6}$$

where the sum is over all possible partitions of users and items into groups, $n^r_{\sigma_u \sigma_i}$ is the number of ratings with value r observed from users in group σ_u to items in group σ_i, and $n_{\sigma_u \sigma_i}$ is the total number of observed ratings from users in group σ_u to items in group σ_i. The $\mathcal{H}(P_U, P_I)$ is understood as an energy function or Hamiltonian which weighs the contribution of each partition of users and items (P_U, P_I) to the sum over all pairs of partitions,

$$\mathcal{H}(P_U, P_I) = \sum_{\alpha,\beta} \left[ln\left(\left(n_{\alpha\beta} + |S| - 1\right)!\right) - \sum_{s=1}^{|S|} ln\left(\left(n^{s}_{\alpha\beta}\right)!\right) \right] \tag{11.7}$$

and $\mathcal{Z} = \sum_{(P_U,P_I)} e^{-\mathcal{H}(P_U,P_I)}$ is called the partition function. In order to estimate the sum over all partitions, [14] estimated $p_{SBM}(r_{ui} = r|R^{\mathcal{O}})$ using Metropolis-Hastings sampling [13, 27]. Note that within this approach, no prior assumptions are made on the grouping of the users and items, or in the desired shape of the connection probability matrix, so that the algorithm itself samples/selects those SBMs which provide the best description of the data.

An advantage of this approach is that we obtain the whole distribution for each rating $P_{SBM}(r_{ui} = r|R^{\mathcal{O}})$. Therefore, one can choose how to make predictions: using the most likely rating, the mean or the median, among others. In [14], the authors chose to select the most likely rating

$$r_{ui} = \arg\max_{r}\{p_{SBM}(r_{ui} = r|R^{\mathcal{O}})\}. \tag{11.8}$$

This probabilistic prediction is in contrast to most recommender systems like matrix factorization and Item-Item algorithms, where the prediction is expressed as a single real number.

The bipartite SBM recommender we just described has two main advantages: (1) it is based on plausible hypotheses about how individuals' preferences arise, and (2) it is mathematically rigorous since it is the result of the full Bayesian probabilistic treatment of the model. However, the correct probabilistic treatment of the model comes at the cost of producing a slow algorithm. The approach above relies on Markov chain Monte Carlo sampling over partitions to make rating predictions, therefore, its computational time does not scale well with the size of the dataset (see Fig. 11.4b). This fact makes it impractical for datasets with millions of ratings [14].

11.3.2 Predictions Based on Mixed-Membership Stochastic Block Model

In this section, the Mixed-Membership Bipartite Stochastic Block Model (MMSBM) approach for recommendation is considered. As previously mentioned, mixed membership models allow nodes to belong to all possible (latent) groups with a finite probability [28, 29]. In our case, we consider a bipartite MMSBM in which we have latent groups for it assumes that each node in the bipartite graph of users and items belongs to a mixture of groups.

In the recommendation problem our goal is to estimate the probability $p(r_{ui} = r|R^{\mathcal{O}})$. In order to do so, we need to compute the likelihood of the observed data given the model parameters. To that end, we define the model parameters as follows.

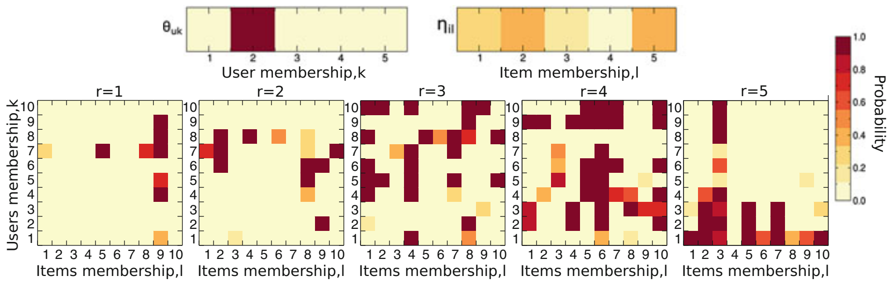


Fig. 11.3 Mixed-Membership Stochastic Block Model: We illustrate the parameters of a bipartite MMSBM for an equal number of latent groups of users and items $K = L = 5$ obtained for the MovieLens 100K dataset. In the top row, we show examples of mixed-membership vectors $\boldsymbol{\theta}$ and $\boldsymbol{\eta}$ for user u and item i, respectively. Each vector component θ_{uk} (η_{il}) is the probability that user u (item i) belongs to group k (group l). Probabilities are shown in colours following the colour bar on the right hand side. In this example, user u has a higher probability to belong to group $k = 2$, while item i has similar probabilities to belong to any group. In the bottom row, we show the inferred values for the probability matrices $\mathbf{Q}$. From left to right, the five matrices correspond to the ratings $r = 1, 2, 3, 4, 5$. For each one of these matrices, the rows and columns correspond to user and item groups, respectively. Each matrix element is the probability $Q^r_{k\ell}$ that a user in group k gives a rating r to an item in group ℓ. Notice that there is no ordering of the probability matrices that would make them diagonal

In the bipartite MMSBM, we consider that there are K groups of users and L groups of items. For each pair of user-item groups k, ℓ, there is a probability $Q^r_{k\ell}$ $\forall r \in S$ that users in group k give rating r to items in group ℓ. Note that because in $R^{\mathcal{O}}$ each user-item edge has only one rating r the probability matrices $Q^r_{k\ell}$ are normalized

$$\forall k, \ell : \sum_{r \in S} Q^r_{k\ell} = 1\,. \tag{11.9}$$

To model mixed group memberships, each user u has a vector $\theta_u \in R^K$, where θ_{uk} denotes the extent to which user u belongs to group k. Similarly, each item i has a vector $\eta_i \in R^L$ (see Fig. 11.3). These vectors are normalized as,

$$\sum_k \theta_{uk} = 1, \tag{11.10}$$

$$\sum_\ell \eta_{i\ell} = 1. \tag{11.11}$$

Given the membership vectors θ_u and η_i, and the probability matrices $Q^r_{k\ell}$, the probability distribution of each rating r_{ui} is a convex combination,

$$p(r_{ui} = r) = \sum_{k,\ell} \theta_{uk} \eta_{i\ell} Q^r_{k\ell}\,. \tag{11.12}$$

Abbreviating all these parameters as $\boldsymbol{\theta}, \boldsymbol{\eta}, \mathbf{Q}$, the likelihood of the observed ratings is thus

$$P(R^{\mathcal{O}}|\boldsymbol{\theta}, \boldsymbol{\eta}, \mathbf{Q}) = \prod_{(u,i)\in R^{\mathcal{O}}} \sum_{k,\ell} \theta_{uk}\eta_{i\ell} Q_{k\ell}^{r_{ui}} . \tag{11.13}$$

In order to perform a full Bayesian approach as for the simple bipartite SBM, we would have to compute the integral in Eq. (11.2) to obtain $p(r_{ui} = r|R^{\mathcal{O}})$. However, this is unfeasible for the current model. Therefore, in order to estimate $p(r_{ui} = r|R^{\mathcal{O}})$, we make a steepest descent approximation and evaluate the integral by considering the model parameters $\hat{\boldsymbol{\theta}}$, $\hat{\boldsymbol{\eta}}$ and $\hat{\boldsymbol{Q}}$ that maximize the likelihood in Eq. (11.13).

Note that while this approximation should in principle not perform as well as considering all possible model parameters, our results show that the maximum likelihood prediction for the bipartite MMSBM produces as accurate predictions as the full probabilistic treatment of the simple bipartite SBM (Fig. 11.5). Our results suggest that the introduction of mixed-membership vectors seems to already provide enough flexibility to the model to capture all the patterns covered by the model averaging in the simple bipartite SBM approach. The maximum likelihood parameters $\hat{\boldsymbol{\theta}}$, $\hat{\boldsymbol{\eta}}$, $\hat{\boldsymbol{Q}}$ are inferred using an efficient expectation-maximization algorithm (EM). We start with a standard variational trick that changes the log of a sum into a sum of logs, writing

$$\begin{aligned}
\log P(R^{\mathcal{O}}|\boldsymbol{\theta}, \boldsymbol{\eta}, \mathbf{Q}) &= \sum_{(u,i)\in R^{\mathcal{O}}} \log \sum_{k\ell} \theta_{uk}\eta_{i\ell} Q_{k\ell}^{r_{ui}} \\
&= \sum_{(u,i)\in R^{\mathcal{O}}} \log \sum_{k\ell} \omega_{ui}(k,\ell) \frac{\theta_{uk}\eta_{i\ell} Q_{k\ell}^{r_{ui}}}{\omega_{ui}(k,\ell)} \\
&\geq \sum_{(u,i)\in R^{\mathcal{O}}} \sum_{k\ell} \omega_{ui}(k,\ell) \log \frac{\theta_{uk}\eta_{i\ell} Q_{k\ell}^{r_{ui}}}{\omega_{ui}(k,\ell)} .
\end{aligned} \tag{11.14}$$

Here $\omega_{ui}(k, \ell)$ is the estimated probability that a given ranking r_{ui} is due to u and i belonging to groups k and ℓ, respectively, and the lower bound in the third line is Jensen's inequality $\log \bar{x} \geq \overline{\log x}$. The equality holds when

$$\omega_{ui}(k,\ell) = \frac{\theta_{uk}\eta_{i\ell} Q_{k\ell}^{r_{ui}}}{\sum_{k'\ell'} \theta_{uk'}\eta_{i\ell'} Q_{k'\ell'}^{r_{ui}}} , \tag{11.15}$$

giving us the update Eq. (11.15) for the expectation step. For the maximization step, we derive update equations for the parameters $\boldsymbol{\theta}, \boldsymbol{\eta}, \mathbf{Q}$ by taking derivatives of the log-likelihood (11.14). Including Lagrange multipliers for the normalization constraints (11.11), we obtain

$$\theta_{uk} = \frac{\sum_{i\in\partial u}\sum_{l}\omega_{ui}(k,\ell)}{\sum_{i\in\partial u}\sum_{k\ell}\omega_{ui}(k,\ell)} = \frac{\sum_{i\in\partial u}\sum_{l}\omega_{ui}(k,\ell)}{d_u}\,, \tag{11.16}$$

where d_u is the degree of the user u. Similarly,

$$\eta_{i\ell} = \frac{\sum_{u\in\partial i}\sum_{k}\omega_{ui}(k,\ell)}{\sum_{u\in\partial i}\sum_{k\ell}\omega_{ui}(k,\ell)} = \frac{\sum_{u\in\partial i}\sum_{k}\omega_{ui}(k,\ell)}{d_i}\,, \tag{11.17}$$

where d_i is the degree of item i. Finally, including a Lagrange multiplier for (11.9), we have

$$Q^r_{k\ell} = \frac{\sum_{(u,i)\in R^{\mathcal{O}}|r_{ui}=r}\omega_{ui}(k,\ell)}{\sum_{(u,i)\in R^{\mathcal{O}}}\omega_{ui}(k,\ell)}\,. \tag{11.18}$$

These equations can be solved iteratively with the following EM algorithm. Starting with an initial estimate of $\boldsymbol{\theta}$, $\boldsymbol{\eta}$ and $\mathbf{Q}$, we repeat the following steps until the parameters converge:

1. (Expectation step) use (11.15) to compute $\omega_{ui}(k,\ell)$ for $(u,i)\in R^{\mathcal{O}}$,
2. (Maximization step) use (11.16)–(11.18) to compute $\boldsymbol{\theta}$, $\boldsymbol{\eta}$ and $\mathbf{Q}$.

The number of parameters and terms in the sums in Eqs. (11.15)–(11.18) is $NK + ML + |R^{\mathcal{O}}|KL$. Assuming that K and L are constant, this is $O(N + M + |R^{\mathcal{O}}|)$, and hence linear in the size of the observed ratings (see Fig. 11.4a). As the set of observed ratings $R^{\mathcal{O}}$ is typically very sparse because only a small fraction of all possible user-item pairs have observed ratings, the expectation-maximization algorithm is feasible even for very large datasets.

In summary, the MMSBM approach has a double advantage: (i) it uses a model that is realistic and flexible, and (ii) the algorithm scales with the number of observed ratings, and is therefore suitable for very large datasets. In addition, it is consistent for sparse datasets, giving good results with few ratings per user (users in datasets in Sect. 11.5 rate typically less than 10 items, but they are enough to give good predictions).

11.4 State of the Art: Other Non-Network Based Collaborative Filtering Approaches

As already mentioned, collaborative filtering algorithms find similarities between users and items to make predictions, instead of focusing on the content or known external information regarding users or items other than user-item ratings. There are different strategies to identify these similarities or patterns in the recommender system. Two of the most representative approaches are neighbour-based models such as Item-Item or User-User approaches, and latent factor models such as Matrix

Factorization, commonly used also as benchmark algorithms to compare against novel recommendation models. While neighbour-based models are simple and intuitive, Matrix Factorization techniques are usually more effective because they allow us to discover the latent features underlying the interactions between users and items. Neighbour-based models are sometimes considered graph-based models, given that they use the structure of the bipartite network to compute similarities between users or between items, and they have also been called model-based algorithms [10]. In this chapter, we will consider as network-based models only those using network inference. Within this section, we explain the rationale for some of the most widely used CF algorithms and analyse some of the main theoretical differences between them and the network-based models SBM and MMSBM.

Item-Item

Neighbour-based CF models generate recommendations using only information about rating profiles for different users. There are two approaches, the User-User approach and the Item-Item approach. In the former, the algorithm finds users with a rating history similar to the query user (neighbours) and generates recommendations using this neighbourhood; for the Item-Item, the algorithm finds similar items to the query item based on their rating history, and generates recommendations using the query item's neighbourhood. For rating systems with much more users than items (as is the case in the datasets we analyse), the Item-Item approach gives better predictions than the User-User approach and is computationally more efficient, therefore from now on we will focus on the Item-Item algorithm, taking into account that the User-User model is computed analogously [10]. Let us assume that we have a list of users $U = \{u_1, \ldots, u_N\}$ and items $I = \{i_1, \ldots, i_M\}$, which the users have rated. The Item-Item approach assumes that the rating from user u to an item i should be similar to the rating she gives to similar items to i. Considering the vector $\vec{i} \in R^N$ of ones for users that have rated item i and zeros otherwise, we can obtain the similarity between item pairs (i, j) by computing the cosine similarity between $\vec{i}$ and $\vec{j}$ as,

$$\text{sim}(i, j) = \cos(i, j) = \frac{\vec{i} \cdot \vec{j}}{||\vec{i}||_2 ||\vec{j}||_2}, \tag{11.19}$$

where $|| \cdot ||_2$ denotes the Euclidean norm of a vector. For other adjusted versions of the similarity see [10]. Note that we can only establish similarities between items that have been rated by the same users.[1] According to the similarity measure, we define the neighbourhood of an item i, ∂i as those k items with highest similarity i.

[1]For the User-User model the reasoning is equivalent: each user is represented by a vector with all the items she has rated V_u. The similarity between users would be computed as in Eq. (11.19) as $\text{sim}(u, v)$.

Hence, the prediction of r_{ui} would be the average of the ratings that user u gave to the items in the neighbourhood of item i as,

$$r_{ui} = \frac{\sum_{j \in \partial i}(\text{sim}(i, j) \cdot r_{uj})}{\sum_{j \in \partial i}(|\text{sim}(i, j)|)}. \tag{11.20}$$

Note that if in the k-nearest neighbours there is no item rated by u, the algorithm cannot perform a prediction, which may happen for sparse datasets. Also, the algorithm assumes a linear psychological scale on the ratings, that is, a rating of 5 is seen as five times better than a rating of 1, but unfortunately this is not necessary in agreement with people's perception [26].

Matrix Factorization Approaches

The most widely used methods for recommendations are the Latent feature or Matrix Factorization methods [11, 30]. Latent feature models assume that there is a space of latent attributes of users and items that determine user-item ratings. Therefore, ratings are not independent from one another but set by the specific position in the latent feature space of users and items. Specifically, MF assumes that there exists a single latent feature space for both users and items, and that the rating of user u to item i is proportional to the closeness between the two in this space. The dimension of the latent space K is much smaller than the number of users and items, such that the problem is dimensionally reduced. Formally, this is equivalent to assuming that the matrix of observed ratings $R^{\mathcal{O}}$ (with a number of rows equal to the number of users N, and a number of columns equal to the number of items M) can be decomposed into

$$R^{\mathcal{O}} = P\, Q, \tag{11.21}$$

where P is an $N \times K$ matrix associated with the users and Q is a $K \times M$ matrix associated with the items. Each row of the P matrix p_u could be seen as a K-dimensional vector with the feature values of user u that describe her, and each column of the Q matrix q_i is a K-dimensional vector with the values of the features that describe item i, with $K << N$ and $K << M$.

The most efficient method until now to factorize the rating matrix, although there are several methods, is the singular value decomposition (SVD) [30]. This method finds the two smaller matrices whose product minimizes the difference with the original ratings matrix (measured as a means squared error). In addition it uses gradient descent to learn a Matrix Factorization (by taking derivatives of the error function over the parameters of the model it is trying to infer). The predicted rating is then

$$r_{ui} = \sum_{k} p_{uk} q_{ik}. \tag{11.22}$$

SVD-MF algorithm is computationally very efficient and makes very good predictions. Also, it has the advantage that it results in intuitive meanings of the resultant matrices and that the resulting algorithm is scalable (see Fig. 11.4) so it can potentially handle very large datasets. The main problem with this approach is that features that describe the users and the items are the same, which imposes severe constraints on the expressiveness of the model (for instance, two users close in feature space must like the same type of items and there is little flexibility to account for the fact that some users might like some items but have different opinions about other items).

Moreover, as the prediction is (with some corrections) the scalar product of the users' and items' feature vectors, this is equivalent to assuming linearity between ratings instead of assuming that ratings are independent categories as was assumed for the bipartite SBM and MMSBM.

As an extension to the "classical" MF, we also consider a mixed-membership implementation of MF, the Mixed-Membership Matrix Factorization (MMMF) [12]. The MMMF model combines Matrix Factorization with a mixed membership context bias. In MMMF, users and items are endowed with both latent factor vectors (p_u and q_i) and discrete topic distribution parameters ($\theta^U_{uk} \in K^U$ and $\theta^M_{ij} \in K^M$). Together with the user and item topics, MMMF models also introduce the affinity of user u to item topic k as c^k_u and the affinity of item i to user topic j as d^j_i. The topic distribution parameters and the affinity of users and items to the topics jointly specify a context bias β^{jk}_{ui}. Therefore, a user generates a rating for an item by adding the contextual bias to the MF inner product with some Gaussian noise,

$$r_{ui} \sim \mathcal{N}(p_u \cdot q_i + \beta^{jk}_{ui}, \sigma^2). \tag{11.23}$$

In [12] authors consider two different MMMF models that differ in how the contextual bias is built. The Topic-Indexed Bias Model (TIB) assumes that the contextual bias decomposes into a latent user bias and a latent item bias so that $\beta^{jk}_{ui} = \sum_{k=1}^{K^M} c^{t(t)}_u \theta^{M(y)}_{ik} + \sum_{j=1}^{K^U} d^{j(t)}_i \theta^{U(t)}_{uj}$. The Topic-Indexed factor Model (TIF) assumes that the joint contextual bias is an inner product of topic-indexed factor vectors, so that $\beta^{jk}_{ui} = \sum_{k=1}^{K^M} \sum_{j=1}^{K^U} \theta^{M(y)}_{ik} \theta^{U(t)}_{uj} c^{k(t)}_u \cdot d^{j(t)}_i$. They use a Gibbs sampling MCMC procedure to draw samples of topic and parameter variables. Then, the posterior mean prediction for each user-item pair under these MMMF models is,

$$\frac{1}{T}\sum_{t=1}^{T}\left(p^{(t)}_u \cdot q^{(t)}_i + \beta^{jk}_{ui}\right). \tag{11.24}$$

The results shown in Sect. 11.5 are for the MMMF-TIF model since it outperforms the MMMF-TIB in all the datasets. Note that analogously to the MF, MMMF also assumes linearity between ratings values.

11.4.1 Advantages of Network-Based Models

There are a number of advantages to using the network-based models we have presented (the bipartite SBM and the MMSBM) compared to previous work on collaborative filtering.

First, unlike Matrix Factorization approaches such as [11] or their probabilistic counterparts [31–33], the ratings $r_{ui} \in \{1, 2, 3, 4, 5\}$ are not treated as integers. As has been established in the literature, giving a movie a rating of 5 instead of 1 does not mean the user likes it five times more [26]. Indeed, the results in Sect. 11.5 suggest that it is better to think of different ratings simply as different labels on the links of the network.

Second, network-based methods yield a distribution over the possible ratings directly, rather than a distribution over integers or reals that must be somehow mapped to the space of possible ratings [31–33]. The network-based models we have presented considered the observed ratings as a bipartite network with metadata (or labels) on the links. An alternative approach would be to consider a multi-layer representation of the data as in [34].

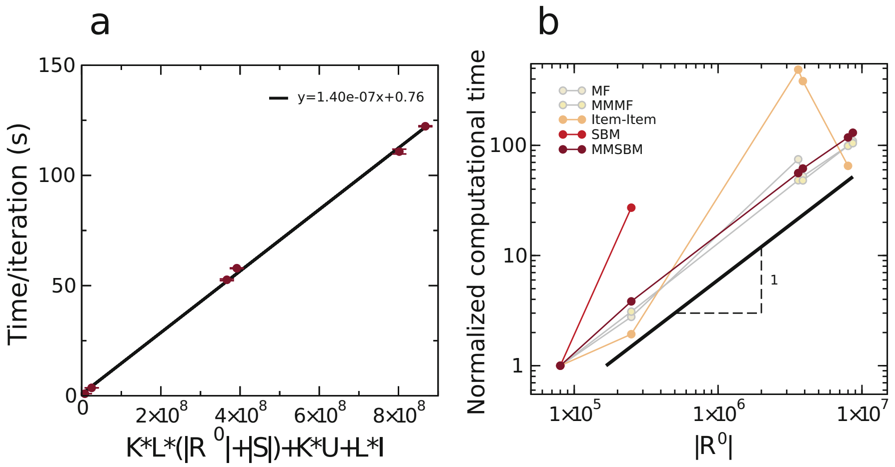


Fig. 11.4 Scalability: (**a**) Scalability of the MMSBM algorithm. Each point represents the average time per iteration in seconds for each of the datasets we use in the study (100K MovieLens, 10M MovieLens, Yahoo! Songs, W-M dating agency, M-W dating agency and Amazon books) versus the number of parameters computed at each iteration $K*L*(|R^{\mathcal{O}}|+|S|)+K*N+L*M$ where N is the number of users, M is the number of items and $|R^{\mathcal{O}}|$ is the number of observed ratings, $|S|$ is the number of different ratings values for each recommender systems and K and L are the number of groups for users and items, respectively ($K = L = 10$ for all the datasets; see Table 11.1 for remaining parameters for each dataset). The continuous line is the linear fit of the real data, which shows that the computational times per iteration scales linearly with the size of the corpus for the whole range. (**b**) Scaling of the different benchmark algorithms we consider in our analysis with the total number of observed ratings. The vertical axis is normalized by the computational time of the smallest dataset—100K MovieLens. MF, MMMF and MMSBM algorithms scale linearly with the total number of observed edges, while the Item-Item algorithm does not. Note that for the bipartite SBM we could only get results for the two smallest datasets, so we cannot establish a linear relationship in this case

Third, the bipartite SBM and the MMSBM do not assume that the matrices **Q** have any particular structure. In particular, they do not assume either that groups of individuals correspond to groups of items or that individuals prefer items that belong to their own group (which mathematically would result in diagonal **Q** matrices). Thus, the SBMs and the resulting algorithms can learn arbitrary couplings between groups of individuals and groups of items, and do so independently for each possible rating, thus overcoming the limitation of expressivity of MF factorization approaches that consider a diagonal **Q** matrix. Importantly, the MMMF does not circumvent this issue despite considering arbitrary couplings between users/items and topics. In fact, MMMF rating predictions are the sum of a MF term and a correction that uses mixed group memberships that are unrelated to the feature vectors [12]. While this is an improvement over MF, it does not fundamentally remove the limiting assumption that each group of users has a corresponding group of items that they prefer. Indeed, our numerical results show that the performance of MMMF is fairly close to that of MF in the datasets we considered.

Finally, all the network-based models presented here do not assume that individuals only see movies (say) that they like, and they do not treat missing links as zeroes or low ratings as is typically done in MF algorithms that need a full matrix to decompose. There are other physics-inspired methods that exploit the structure of the bipartite user-item network and use classical physics processes to make recommendations such as random walks [35] or heat diffusion [36]. However, all these approaches are used for link prediction, that is, they only try to predict which item would be collected by a user [37].

11.5 Results

We show the performance of the network-based and the Item-Item and MF algorithms for six datasets: the MovieLens 100K and 10M datasets with 100,000 and 10,000,000 ratings, respectively (https://movielens.org), Yahoo! songs (R3—https://webscope.sandbox.yahoo.com/catalog.php?datatype=r), Amazon books [38, 39] (http://jmcauley.ucsd.edu/data/amazon/), and the dataset from LibimSeTi.cz dating agency [40] (http://www.occamslab.com/petricek/data/), which is split into two datasets, one consisting of males rating females and vice versa. These datasets are diverse in the types of items considered, the sizes $|S|$ of the sets of possible ratings and the density of observed ratings (see Table 11.1).

To check the predictive power of the different algorithms we show the results for a fivefold cross validation in each of the six datasets. That is, we divide each dataset into five equal parts and we make the five possible combinations of using one part as the test set and the other four as the training set. We measure the predictability in terms of accuracy, that is, the number of ratings predicted correctly, and the mean absolute error (MAE). Figure 11.5 shows the performance for the two network-based models, the simple bipartite SBM (using the approach in [14]) and the Mixed-Membership Stochastic Block Model (MMSBM) (using the approach

Table 11.1 Dataset characteristics

Dataset	Ratings scale S	#Users	#Items	#Ratings
MovieLens 100K	$\{1, 2, 3, 4, 5\}$	943	1682	100,000
MovieLens 10M	$\{0.5, 1, 1.5, \dots, 5\}$	71,567	65,133	10,000,000
Yahoo! Songs	$\{1, 2, 3, 4, 5\}$	15,400	1000	311,700
M-W dating agency	$\{1, 2, \dots, 10\}$	220,970	135,359	4,852,455
W-M dating agency	$\{1, 2, \dots, 10\}$	135,359	220,970	10,804,040
Amazon book	$\{1, 2, 3, 4, 5\}$	73,091	539,145	4,505,893

The total number of possible ratings is different for each dataset; ratings are in a scale from 1 to 5 in all datasets for the two dating agency datasets, which have a rating scale from 1 to 10. Ratings are integers except for the MovieLens 10M dataset which allows half-integer values. Note that, in the latter case we expect a smaller MAE than if only integer values were allowed. All datasets have millions of ratings except for MovieLens 100K and Yahoo! Songs

in [15]). Moreover, we show the comparison of the network-based approaches with three benchmark algorithms (see Sect. 11.4): the Item-Item algorithm [10], which predicts r_{ui} based on the observed ratings of user u for items that are the most similar to i, a "classical" Matrix Factorization (MF) [11] and Mixed-Membership Matrix Factorization (MMMF) [12]; as well as a baseline naive algorithm that assigns to each test rating r_{ui} the average of the observed ratings for item i, that is $r_{ui} = \frac{1}{d_i}\sum_{u'\in\partial_i} r_{u'i}$.

The results for the MMSBM are for $K = L = 10$, i.e., that is 10 groups of users and 10 groups of items (recall that there is any correspondence between these groups). The performance for larger choices of K and L does not improve significantly [15]. Since iterating the EM equation of Eqs. (11.15)–(11.18) can lead to different solutions depending on the initial conditions, the results correspond to an average of the predicted probability distribution of ratings over 500 independent runs. There is a freely available implementation of the MMSMB in gitHub by Bill Jeffries (https://github.com/billjeffries/mixMemRec). The code is written in Spark's recommender library and can process large datasets. Notice however that the current implementation gives the results for a single run, therefore one should expect accuracies to be lower if a single run is considered.

The bipartite SBM does not require a pre-specification of model parameters since they are sampled by the algorithm. You can find a freely available implementation of the code (http://seeslab.info/downloads/network-c-libraries-rgraph/inrgraph-2.2.1/recommender/). For the Item-Item algorithm implemented in Lenskit, we set $k = 50$; for the Matrix Factorization we also used the Lenskit implementation with $k = 50$ features, a learning rate of 0.002 and an initialization of 0.1 for every user-feature and item-feature value as suggested in [26]. Finally, for MMMF we use the Matlab implementation provided by the authors (https://code.google.com/archive/p/m3f/).

Another thing to take into account is that the network-based models, both the bipartite SBM and the MMSBM, are probabilistic models. That means that for each rating a user gives an item we have a probability distribution of ratings that results from the average of the probabilities for all the sampling set. Therefore, we can choose how to make predictions from the probability distribution of ratings: the mode (that is the rating with the highest probability), the mean or the median. As stated earlier, we measure the performance in terms of accuracy and the mean absolute error (MAE), which gives us an idea how far predictions are from the real values. For the network-based model, the best estimator for the accuracy is the most likely rating from the probability distribution of ratings, while for the MAE the best estimator is the median. In contrast, the predictions of the MF, MMMF and Item-Item models are a single real per rating.

In Fig. 11.5 we find that in most cases the network-based approaches, the bipartite SBM and MMSBM, outperform the Item-Item algorithm, MF and MMMF. Indeed, when considering the accuracy the MMSBM is significantly better than MF and MMMF for all the datasets we tested, and better than the item-item algorithm in five out of six datasets, the only exception being the Amazon Books dataset. In terms of the mean absolute error (MAE), the MMSBM is the most accurate in four out of the six datasets (item-item, MF and MMMF produce smaller MAE in the Amazon Books and MovieLens 10M datasets). Note that the Amazon dataset is different from the others in that users only rate items after buying them, and knowing a priori the average rating of the item given by previous buyers, which might bias their choices.

Interestingly, the MMSBM approach produces results that are almost identical to those of the bipartite SBM [14] for the two examples for which inference with the bipartite SBM is feasible. In particular, the MMSBM achieves the same accuracy with $K = L = 10$ in the mixed-membership model as the bipartite SBM with sampling models with 50 groups on average. This suggests that many of the groups observed in [14] are in fact mixtures of larger groups, and that the additional expressiveness of the MMSBM allows us to succeed with a lower-dimensional model.

Given that in general Matrix Factorization approaches outperform the Item-Item model, and that the MMSBM and the bipartite SMB give similar results, we quantify the improvement of the MMSBM over the classical MF (with very similar results to the MMMF). To do so, we compute the relative improvements in the accuracy (%) as

$$\frac{(\mathrm{acc}_{\mathrm{MMSBM}}) - (\mathrm{acc}_{\mathrm{MF}})}{(\mathrm{acc}_{\mathrm{MF}})} * 100, \tag{11.25}$$

with improvements of 5% in the MovieLens 100K dataset, 45% in the MovieLens 10M, 41% in the Yahoo songs, 42% in the M-F LibimSeti agency, 27% in the F-M LibimSeti agency and finally a 3% improvement in the Amazon dataset. For the Amazon dataset, the relative improvement of the Item-Item over the MMSBM is of 13%.

11.6 Discussion

We have shown that network-based approaches, based on inference using the block structure of social networks, give predictions of human preferences that are in most of the cases significantly and considerably more accurate than leading collaborative filtering recommendation algorithms.

In the case of the simple bipartite SBM, it is worth noting that the gain in accuracy comes at the expense of computational cost as a result of the Monte Carlo sampling of the user and item partition space. Although the algorithm is able to give predictions on datasets in the order of $\sim$1000 of users and items and $\sim$100,000 of ratings, handling even one order of magnitude is challenging. Instead, the MMSBM inference using expectation-maximization method results in a scalable algorithm able to handle datasets with millions of ratings.

In any case, network-based recommender systems not only provide better predictions, but also have some desirable features: they are analytically tractable allowing for a mathematically rigorous approach, they are based on plausible social models, and they provide interpretable results.

With respect to mathematical rigour, the Bayesian approach used by the bipartite SBM [14] is the complete and correct probabilistic treatment of the observations. However, the results of the MMSBM suggest that introducing the mixed-membership of users and items is already equivalent to sampling over different sets of simple bipartite SBMs [15].

Importantly in both cases, we obtain an estimate of the whole probability distribution for each rating. From this, we can choose how to make predictions using the most likely rating, the mean or the median among others. In contrast, recommender systems like those based on Matrix Factorization or Item-Item give predictions that are a single number, the most likely rating (or a real number that should be rounded to the closer value in the ratings set), that may even be outside the rating range (for example, $r_{ui} = 1.1$ when $S \in \{0, 1\}$). Additionally, these algorithms assume that ratings are linearly spaced in the mind of users (that is, that the difference between $r = 1$ and $r = 2$ is the same as between $r = 4$ and $r = 5$), which does not seem to be in accordance with people's perception [26].

Finally, network-based approaches are based on models that were originally defined and are widely used to explain how social agents establish relationships, and are therefore in a better position to illuminate which social and psychological factors determine human preferences. As an interesting by-product of this, we note that it is possible to use them to infer demographic properties from ratings alone, a subject that is of much current interest for commercial purposes such as Social Marketing [41, 42].

The future of network-based recommender systems is likely to involve the introduction of contextual information about users and/or items into the inference process. The network-based models we have discussed in this chapter have the advantage that metadata can mathematically be introduced in the form of *a priori*

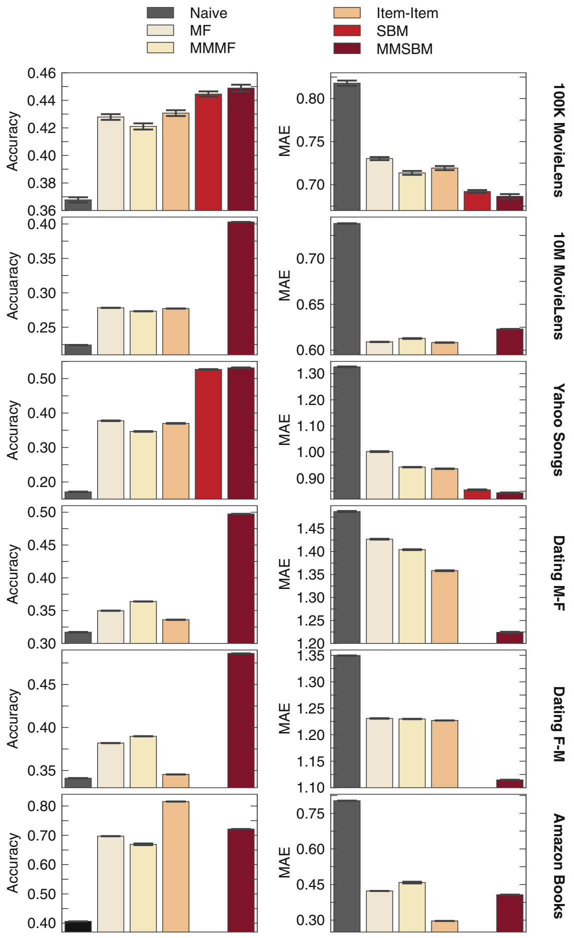


Fig. 11.5 Algorithm comparison: From top to bottom, the datasets are MovieLens 100K, MovieLens 10M, Yahoo Songs, men rating women (M-W) in the LibimSeTi dataset, women rating men (W-M) in the LibimSeTi dataset and Amazon books. The left column displays the accuracy of the algorithms in each dataset, i.e., the fraction of ratings that are exactly predicted by each algorithm. The right column displays the mean absolute error (MAE) in the predicted vs. actual rating, treated as an integer or half-integer. In all cases, the bars are the average of a fivefold cross-validation and the error bars correspond to the standard error of the mean. The bipartite SBM algorithm does not scale to the larger datasets, hence it was evaluated only on the MovieLens 100K and Yahoo Songs datasets. Importantly, bipartite SMB algorithm achieves similar accuracy to the MMSBM on the datasets it can handle. The MMSBM model and algorithm achieves the best (highest) accuracy in five out of six datasets, and the best (lowest) MAE in four out of six datasets

probabilities for model parameters and specifically for group membership vectors. The intuition behind this idea is simple: we expect users (items) with similar associated metadata to have similar membership vectors. With this type of approach, network-based models will be better suited to industrially relevant problems such as the problem that arises when a new product is introduced into the market. The use of relevant metadata can be informative about the most plausible group membership vectors for each item and therefore help in identifying the range of users who could potentially be interested in that product.

References

1. Castells M (2001) *The Internet Galaxy: Reflections on the Internet, Business, and Society*. (Oxford University Press, Inc., New York, NY, USA).
2. Kalakota R, Robinson M (1999) *e-Business*. (Addison Wesley Roadmap for Success).
3. Brynjolfsson E, Hu Y, Smith MD (2003) Consumer surplus in the digital economy: Estimating the value of increased product variety at online booksellers. *Management Science* 49(11):1580–1596.
4. Netflix Prize Rankings. Hacking NetFlix. http://www.hackingnetflix.com/2006/10/. Accessed 09-March-2017
5. The BellKor 2008 Solution to the Netflix Prize. http://www.research.att.com/people/Volinsky_Christopher_T/custom_index.html. Accessed 09-March-2017
6. Netflix Prize Leaderboard 2009. http://www.netflixprize.com/leaderboard.html. Accessed 09-March-2017
7. Lohr S (2009) A $1 million research bargain for Netflix, and maybe a model for others. *The New York Times* 22.
8. Bobadilla J, and Ortega F, and Hernando A and Gutiérrez A (2013) A. Recommender systems survey *Knowledge-Based Systems* 46:109–132.
9. Deshpande M, Karypis G (2004) Item-based top-n recommendation algorithms. *Acm Transactions on Information Systems* 22:143–177.
10. Sarwar B, Karypis G, Konstan J, Riedl J (2001) *Item-based collaborative filtering recommendation algorithms*, WWW '01. (ACM, New York, NY, USA), pp. 285–295.
11. Koren Y, Bell R, Volinsky C (2009) Matrix factorization techniques for recommender systems. *Computer* 42(8):30–37.
12. Mackey L, Weiss D, Jordan M I (2010) Mixed membership matrix factorization. *Proceedings of the 27th International Conference on Machine Learning*, pp 711–718.
13. Guimerà R, Sales-Pardo M (2009) Missing and spurious interactions and the reconstruction of complex networks. *Proc. Natl. Acad. Sci. U. S. A.* 106(52):22073–22078.
14. Guimerà R, Llorente A, Moro E, Sales-Pardo M (2012) Predicting human preferences using the block structure of complex social networks. *PLoS One* 7(9):e44620.
15. Godoy-Lorite A, Guimerà R, Moore C, Sales-Pardo M (2016) Accurate and scalable social recommendation using mixed-membership stochastic block models. *Proc. Natl. Acad. Sci. U. S. A* 113(50):14207–14212.
16. Girvan M, Newman ME (2002) Community structure in social and biological networks. *Proc. Natl. Acad. Sci. U.S.A.* 99(12):7821–7826.
17. Guimera R, Sales-Pardo M, Amaral LAN (2004) Modularity from fluctuations in random graphs and complex networks. *Physical Review E* 70(2):025101.
18. Guimera R, Sales-Pardo M, Amaral LA (2007) Classes of complex networks defined by role-to-role connectivity profiles. *Nature Physics* 3(1):63–69.
19. Fortunato S (2010) Community detection in graphs. *Physics Reports* 486(3):75–174.
20. Rombach MP, Porter MA, Fowler JH, Mucha PJ (2010) Core-Periphery Structure in Networks. *Physics Reports* 74(1):167–190.
21. White HC, Boorman SA, Breiger RL (1976) Social structure from multiple networks. I. Blockmodels of roles and positions. *American Journal of Sociology* pp. 730–780.

22. Rovira-Asenjo N, Gumí T, Sales-Pardo M, Guimerà R (2013) Predicting future conflict between team-members with parameter-free models of social networks. *Scientific Reports* 3:1999.
23. Aicher C, Jacobs AZ, Clauset A (2014) Learning latent block structure in weighted networks. *Journal of Complex Networks* 3(2):221–248.
24. Peixoto TP (2017) Nonparametric weighted stochastic block models. *arXiv preprint arXiv:1708.01432*.
25. Nowicki K, Snijders TAB (2001) Estimation and prediction for stochastic blockstructures. *Journal of the American Statistical Association* 96(455):1077–1087.
26. Ekstrand MD, Ludwig M, Konstan JA, Riedl JT (2011) Rethinking the recommender research ecosystem: reproducibility, openness, and LensKit. *Proceedings of the Fifth ACM Conference on Recommender Systems* pp. 133–140.
27. Jaynes ET (2003) *Probability Theory: The Logic of Science*. (Cambridge University Press).
28. Airoldi EM, Blei DM, Fienberg SE, Xing EP (2008) Mixed membership stochastic blockmodels. *J. Mach. Learn. Res.* 9(2008):1981–2014.
29. Ball B, Karrer B, Newman ME (2011) Efficient and principled method for detecting communities in networks. *Physical Review E* 84(3):036103.
30. Paterek A (2007) *Improving regularized singular value decomposition for collaborative filtering*. pp. 39–42.
31. Meeds E, Ghahramani Z, Neal RM, Roweis ST (2006) Modeling dyadic data with binary latent factors in *Advances in Neural Information Processing Systems 19*, eds. Schölkopf B, Platt J, Hoffman T. (MIT Press, Cambridge, MA), pp. 977–984.
32. Salakhutdinov R, Mnih A (2008) Probabilistic matrix factorization. *Advances in Neural Information Processing Systems (NIPS '08)* pp. 1257–1264.
33. Shan H, Banerjee A (2010) *Generalized Probabilistic Matrix Community for Collaborative Filtering*, Proceedings of the 2010 IEEE International Conference on Data Mining. (IEEE Computer Society, Washington, DC, USA), pp. 1025–1030.
34. Peixoto TP (2015) Model selection and hypothesis testing for large-scale network models with overlapping groups. *Phys. Rev. X* 5:011033.
35. Zhou T, Ren J, Medo M, Zhang Y-C (2007) Bipartite network projection and personal recommendation. *Physical Review E* 76(2007):046115
36. Zhou T, Kuscsik Z, Liu J-G, Medo M, Wakeling JR, Zhang Y-C (2010) Solving the apparent diversity-accuracy dilemma of recommender systems. *Proc. Natl. Acad. Sci. U. S. A* 107(2010):4511
37. Yu F, Zeng A, Gillard S, Medo M (2016) Network-based recommendation algorithms: A review. *Physica A: Statistical Mechanics and its Applications* 452(2016):192–208.
38. McAuley J, Targett C, Shi Q, van den Hengel A (2015) *Image-Based Recommendations on Styles and Substitutes*, SIGIR '15. (ACM, New York, NY, USA), pp. 43–52.
39. McAuley J, Pandey R, Leskovec J (2015) *Inferring Networks of Substitutable and Complementary Products*, KDD '15. (ACM, New York, NY, USA), pp. 785–794.
40. Brozovsky L, Petricek V (2007) *Recommender System for Online Dating Service*. (VSB, Ostrava).
41. Leo Y, Karsai M, Sarraute C, Fleury E (2016) Correlations of consumption patterns in social-economic networks. *arXiv preprint arXiv:1609.03756*.
42. Andreasen AR (2002) Marketing social marketing in the social change marketplace. *Journal of Public Policy & Marketing* 21(1):3–13.

Chapter 12
Using Network Alignment to Identify Conserved Consumer Behaviour Modelling Constructs

Luke Mathieson, Natalie Jane de Vries, and Pablo Moscato

Abstract Extracting topological information from networks is a central problem in many fields including business analytics. With the increase in large-scale datasets, effectively comparing similarities and differences between networks is impossible without automation. In some cases, computational search of simple subgraphs is used to understand the structure of a network. These approaches, however, miss the "global picture" of network similarity. Here we examine the NETWORK ALIGNMENT problem, in which we look for a mapping between vertex sets of two networks preserving topological information. Elsewhere, we showed that data analytics problems are often of varied computational complexity. We prove that this problem is W[1]-complete for several parameterizations. Since we expect large instances in the data analytics field, our result indicates that this problem is a prime candidate for metaheuristic approaches as it will be hard in practice to solve exact methods. We develop a memetic algorithm and demonstrate the effectiveness of the NETWORK ALIGNMENT problem as a tool for discovering structural information through an application in the area of consumer behaviour modelling. We believe this to be the first demonstration of such an approach in the social sciences and in particular a consumer analytics application.

Keywords Network alignment · Memetic algorithms · Data mining · Parameterized complexity · Consumer behaviour modelling · Betweenness Centrality · Customer Loyalty · Maximum Common subgraph isomorphism problem

L. Mathieson (✉)
School of Electrical Engineering and Computing, The University of Newcastle, Callaghan, NSW, Australia

School of Software, University of Technology Sydney, Ultimo, NSW, Australia
e-mail: luke.mathieson@uts.edu.au

N. J. de Vries · P. Moscato
School of Electrical Engineering and Computing, The University of Newcastle, Callaghan, NSW, Australia
e-mail: natalie.devries@newcastle.edu.au; Pablo.Moscato@newcastle.edu.au

12.1 Introduction

Networks and graphs are powerful tools for modelling knowledge and information originating from large databases. They facilitate the communication of ideas and enable information sharing. When they are derived from data obtained in the real world, these graphs may contain subgraphs of particular interest. This class of problems contains one which has defeated attempts to characterize its computational complexity, the *graph isomorphism problem*, and tight bounds on its computational complexity remain open questions at the time of writing this manuscript.

There is a lot of activity in data-driven science to model real-world problems with networks or graphs. However, when two different network models are presented, a natural question arises: How similar are they? In fact, the number of nodes could be dissimilar, and the underlying edge topology could be very different. For instance, if we are talking about comparing the supply chains graphs of two companies, we would like to have some sort of match between the nodes of the two networks such that a measure of interest is maximized. One such a measure is the number of edges that, after the matching of nodes, are “aligned”. This will indicate a “core component” of the two graphs that is maintained in both supply chain structures. Consequently, those edges which are not part of this “conserved core component” could be assumed to have an extra level of detail. If one of the supply chains is considered to be much more efficient than the other, perhaps those parts in that network which are *not* part of the core conserved component may be the most valuable for the success of this company in comparison with its competitors. Alternatively, the other company may like to revisit some part of their logistics structure since they may not be providing either a core benefit or could be improved by a new redesign. Other possible examples are the comparison of managerial structures in different companies (in that case we may have a directed acyclic graph), any other “business process” structure [11, 12], the structure of two communication, transportation or power networks. In biology, the NETWORK ALIGNMENT problem has application in comparison with metabolism of different species.

Very recently, prompted by the increased need for establishing semantic interoperability as well as needing to jointly analysing the results of biological datasets originating from different technologies, and interest in solving large instances of problems has been sparked in this domain. The computational complexity has led researchers to use metaheuristic methods, such as memetic algorithms, for the problems of *ontology alignment* [2, 32, 33]

The rest of this chapter is organized as follows: Firstly Sect. 12.1.1 provides a brief introduction to consumer behaviour modelling, model constructs and the application domain in general. In Sect. 12.2 we present background information regarding the dataset used (Sect. 12.2.1) and the technical details of the memetic algorithm developed (Sect. 12.2.2) along with the computational experiments performed (Sect. 12.2.3). Section 12.3 gives the results of the experiments conducted and discusses the effectiveness of the approach. The mathematically formal definition of the NETWORK ALIGNMENT problem and complexity theoretic results identifying the parameterized intractability of the problem are shown at the end of the Results section in Sect. 12.3.4. Finally, Sect. 12.4 summaries the work and discusses the future directions.

12.1.1 A Brief Background on Modelling Consumer Behaviour Constructs

The areas of online consumer behaviour and customer brand engagement have seen increasing interest in recent years [9, 24, 31]. Customer engagement in particular has been defined as customer behaviours towards a brand that go beyond purchase [31]. With the mainstream use of social media and online communications, such behaviours are of growing interest and importance to brand managers and are providing increasingly detailed and large data instances. Research into these models has largely been led by *a priori* hypothesizing followed by controlled experiments. See, for instance, [8, 9, 24] where hypotheses are tested using various modelling methodologies driven by theory-perspective.

Traditionally in this domain, data is obtained from consumers by designing either online or offline questionnaires to collect data in order to test hypotheses. Information about each of the specific traits that researchers aim to investigate is extracted using several questions on a common theme. These are called *constructs* which are theoretically defined and guided by *a priori* knowledge from the literature. Each construct is usually made up of multiple questions (for example, five questions all relating to customer loyalty) that attempt to encompass the whole "construct" of the aspect of interest (e.g. "loyalty"). Subsequently, the relationships between these constructs are hypothesized, and methodologies such as Structural Equation Modelling (SEM) can be used to statistically validate a working hypothesis. Particularly with SEM studies, the relationships between different constructs are investigated as well as the "loadings" of each item part of that construct. However, the items that make up each construct are usually not investigated individually.

Here, and in the previous work, we highlight that information may be hidden in the individual items (questions) and the relationships between these items rather than the theoretical constructs at the construct level. We have previously introduced the concept of *functional constructs* which are derived from data-driven methodology to find underlying structure in the dataset without *a priori* assumptions [9].

The introduction of a data-driven methodology to "reverse engineer" [9] consumer behaviour models is deeply rooted in the key scientific process of *observation*. In a data-driven approach, it is from initial observations that hypotheses suitable for testing should be developed. We do expect, however, that in the area of social sciences there would not be an initial agreement on a common set of questions that should be used to collect information.

Our work aims to address a central question: how to combine information about inter-variable relationships, as sampled in questionnaires, and identify the common structures present across the underlying behaviour models coming from the datasets. The relationships of interest in this particular case are the influences that different emotional and cognitive factors have on each other leading to customer engagement and/or loyalty. In [9] these relationships are expressed as a *network* (i.e. a directed graph). By exploring and analysing the common structure of these networks using an alignment process, we can refine the empirical observations. Ultimately, structures that are consistently observed across different datasets can lead to the construction of testable, falsifiable theories. We contest that this is the same structure that is found (conserved) across multiple, related datasets, this structure provides a solid base for further investigation into the behavioural model relationships.

In hypothesized studies where data is collected for the specific purpose at hand, it is easy to control the size of the instance or the number of questions you will ask (i.e. the number of features you will have in your dataset). However, with a more data-driven approach, collecting real-time online data, datasets can be of unprecedented size and dimensions. This is why scalable methods and complexity of the problem becomes so important and this is one of the motivations of this study.

The contributions of our work are several. First, we study the parameterized complexity of the *network alignment problem* and prove it to be W[1]-complete under several natural parameterizations. Second, through a series of computational experiments we collect evidence that shows that the morphism between nodes, obtained after the two networks are aligned, properly identifies/ congruent questions across the two studies. Third, we show using a perturbation-based approach that networks obtained from different populations still maintain topological invariants in functional constructs that are uncovered by the alignment process.

Our results show that network alignment is a powerful tool to identify common relationships among items that sample theoretical constructs and could, via the alignment mapping, identify questions that have similar functional roles in different studies.

12.1.2 Introducing the NETWORK ALIGNMENT *Problem*

The *subgraph isomorphism problem* is NP-complete [6]. In this problem, we are given two graphs G and H as input and we are required to determine whether G contains a graph that is isomorphic (equal to) to H. Another problem with applications in chemoinformatics and pharmacophore mapping is the *maximum common subgraph isomorphism problem* in which we are given two graphs G and G' and an integer $k > 0$ and we need to decide if G contains a subgraph of at least k vertices which is isomorphic to a subgraph of G'. This problem is also NP-Complete [19].

In some applications, the number of vertices of the "query" graph H is very small in comparison with the number of vertices in G and the isomorphism requirement is strict. From a modelling perspective, there are situations where we would be interested in relaxing this constraint. This leads to the *maximum common edge* *subgraph problem* in which, given two graphs G and G' as input, we are required to find a graph H with as many edges as possible which is isomorphic to both a subgraph in G and a subgraph in G'. This problem is also NP-complete [27].

In this paper, we address a problem that can be considered a generalization of some of the problems above in terms of its modelling capabilities: the *network alignment* problem. In this problem we are given two graphs G and G' and we are asked to identify a mapping between the vertices of the graphs that "aligns" as many edges as possible. The edges that align then help to identify the topological characteristics which are common to both graphs. The problem has important applications in many areas such as the inference of biological function by comparing an unknown network with a known, previously annotated genetic network. It has also found use in drug design [3], identification of gene modules from expression data [14] and analysis of PPI networks [25].

Following the work of [2, 25, 32, 33] we develop an approach based on memetic algorithms which allows us to address large instances of the network alignment problem. We present what we believe it is the first application of network alignment as a model tool in the area of computational social sciences. This work is motivated by our quest to model and understand consumer behaviour and is a notable area of interest both academically and to marketers.

12.2 Materials and Methods

The original raw data obtained through a paper-and-pen survey method at The University based in New South Wales, Australia in August 2013. Respondents were existing Facebook users only who had previously used a Facebook brand page. Answers were given on a seven-point *Likert* scale anchored with (1) (corresponding to the answer *"strongly disagree"*) to (7) (for the answer being *"strongly agree"*) "Theoretical constructs" were motivating sets of 3–6 questions to "measure" the construct as is consistent with other social science and marketing studies. More details can be found in [9] and [10] as well as in Chap. 26, and Chap. 5. These other papers and chapters can give an overall idea of what the raw data contains and how it was analysed. Here we use some networks derived from them and we explain how they were generated in the following subsections.

12.2.1 Data Collection and Preparation

The networks used for this work are drawn from data collected through consumer behaviour questionnaires administered as part of a prior study [9].

In brief, the individual questions form groups of variables called *constructs*, which address aspects of customer engagement and online consumer behaviours. Symbolic regression was performed for each variable to predict a model for the

Table 12.1 Theoretical construct abbreviations and references

Construct	Source	Code	Number of items
Usage intensity	[24]	UI	3
Brand involvement	[5]	INV	6
Self-brand-congruency	[21]	SBC	5
Functional value	[24]	FUV	4
Hedonic value	[24]	HED	4
Social interaction value	[24]	SOC	4
Co-creation value	[30]	CCV	6
Customer engagement	[24]	ENG	5
Customer loyalty	[24]	LO	6
SNS-specific loyalty behaviours	[28]	ON	3
Brand interaction value	[24]	BR	4
Flow	[29]	FLOW	6
Informational value	[4]	INF	3
Relationship-building value	[30]	RBV	5
Subjective knowledge	[4]	SK	5

variable as a function of all other variables. The network is constructed from these models where each variable is a vertex and an arc between two vertices exists if the source vertex is part of a model for the target vertex. Full details of the elucidation of the networks from the raw data can be found in [9]. From the data three networks were produced, which we label **Whole**, **A** and **B**, in keeping with the original study [9]. Each network has 69 vertices, corresponding to the original 69 questions and 250, 264 and 239 arcs, respectively (Table 12.1).

12.2.2 A Memetic Algorithm for Network Alignment

The memetic algorithm prototype was built in the Python Programming Language, version 2.7.6. The evolutionary algorithms package DEAP [17] was used as a basis for the underlying genetic algorithm, which we extend with custom mutation and crossover functions and a local search function. The SCOOP package [22] was used to provide parallel and distributed processing functionality. As the two input graphs are not modified, we represent them using a lightweight graph data structure wrapping a Python list which stores the vertex labels and a Python dictionary which constitutes an adjacency list representation of the edges.

The overall algorithm follows the typical memetic algorithm structure. Populations of individual solutions are generated then repeatedly mutated and crossbred with the “fittest” solutions being selected for mutation and crossover. Periodically the evolutionary cycle is interrupted by an iterative improvement phase where every individual is subjected to deterministic optimization (in our case via a local search procedure). The details of the number of generations, the frequency of the deterministic improvement and the size and number of the populations are all parameters which can be tuned to improve performance, either from an optimality standpoint or from a resource-use perspective.

12.2.2.1 Preprocessing

Although the NETWORK ALIGNMENT alignment problem is formulated such that the two graphs G_1 and G_2 can have different numbers of vertices, it is more convenient algorithmically to assume that the two graphs have the same number of vertices. As noted earlier (but in the inverse context), we lose no generality by "padding" the smaller instance with degree zero vertices. Except in the case where a rather unusual fitness function is chosen, the vertices simply act as placeholders and do not affect the solution.

To enhance the speed of determining adjacency in the graphs when computing the fitness function, the algorithm memorizes the results of adjacency testing for the graphs for all pairs of vertices in each graph. If not done with care this process can introduce a large, if not crippling, space overhead. For very large graphs, a naïve approach would require several gigabytes of memory—feasible, but not desirable. Increasing the graph size by an order of magnitude would increase the memory usage by two orders of magnitude which quickly becomes infeasible. To strike a balance, we store only the instances where two vertices are adjacent, exploiting exception handling mechanisms to deal with the non-adjacent cases. Of course this approach will also encounter the same problems for highly connected graphs. In this case we must resort to either computing the adjacencies as needed or using external memory techniques [1].

12.2.2.2 Initial Alignment

In the algorithm we include the optional ability to specify an initial (partial) alignment. If no initial alignment is specified, the initial alignment is randomly chosen. If a partial alignment is given, the elements which are not included in the initial alignment are randomly allocated.

For the experiments in this paper we routinely compute an initial alignment based on the *betweenness centrality* [18] of the vertices in each graph. The betweenness centrality for each vertex in each graph is calculated using the GPU-FAN software package.[1] The vertices are then ordered by the betweenness centrality and paired between the two graphs, starting with the vertices of highest centrality.

[1] http://bioinfo.vanderbilt.edu/gpu-fan/.

12.2.2.3 Individual Representation, Mutation, Crossover and Selection

The representation of the individual solutions, the mutation operator and the selection algorithm are drawn from the DEAP [17] libraries. Implementation details and source code can be found in the DEAP websites.[2,3] The crossover operator is a custom operator developed from the partially matched crossover operator of Goldberg and Lingle Jr. [20]. Here we give an overview of the relevant details.

Individual Representation As we ensure that the two graphs have the same number of vertices, the alignment is a bijective function from the vertices of one graph to the vertices of the other. This can easily represented by a function from $\{0, \ldots, n-1\}$ to $\{0, \ldots, n-1\}$—i.e. a permutation of the indices of the vertices. Thus an individual solution is represented in the algorithm as a permutation of the numbers $\{0, \ldots, n-1\}$, stored as a list with the indices of the list representing the domain of the alignment function.

Mutation Operator To mutate the individuals we use an index shuffling mutation,[4] which considers each element of the individual and, with a given probability, swaps it with a randomly chosen element of the individual. The probability of swapping an element is user defined, for this work, we used 0.05.

Crossover Operator To crossbreed two solutions we employ a modification of the partially matched crossover operation,[5] itself an implementation of a crossover operator developed for the TRAVELLING SALESMAN problem [20]. A vertex in the first graph is randomly selected, along with its 2-neighbourhood. The mappings of these vertices in the two individuals are then swapped.

Selection The individuals to be bred and mutated are selected by a tournament selection process[6] (whereby the same individual can be selected multiple times). For a given k and t, k tournaments are conducted, each between t randomly selected individuals. The fittest of the t is selected as the "winner" of that tournament. Within this paper, we choose k to be the number of individuals and t to be 3.

Restarting To assist with avoiding local optima, the algorithm employs a restart mechanism whereby if the best solution has not improved after a chosen number of iterations of mutation and crossover, the best solution so far is recorded and the population randomly reinitialized.

[2]DEAP Source code: https://github.com/DEAP/deap/.

[3]DEAP Documentation: http://deap.readthedocs.org/en/master/.

[4]deap.tools.mutShuffleIndexes.

[5]deap.tools.cxPartialyMatched.

[6]deap.tools.selTournament.

12.2.2.4 Local Search Optimization

To perform the deterministic optimization of each individual, we employ a simple swap-based local search procedure, similar to the 2-opt local search procedure [7], whereby we randomly select an element in the individual, then find the optimal swap with any other element in the individual. If such a swap exists (i.e. the optimal is not simply the identity mapping), we add the neighbours of the vertex represented by the preimage of the element to be swapped to a list of elements for which we search for an optimal swap. If no swap is found with the initially selected random element, we repeat the process with a new randomly selected implement until either a swap is found or all elements have been tested.

12.2.2.5 Post-Alignment Processing

Once we have an alignment of two networks we have a number of avenues for analysing the structure of the networks. At the most basic level, the alignment itself suggests relationships between the aligned vertices—vertices occupying topologically similar *locorum* are likely to be aligned, which suggests similarities in the rôles the vertices represent.

This, however, is only a first-order analysis of the information encoded in the alignment. The central notion of the NETWORK ALIGNMENT problem is that edge relationships are preserved—that is *topological* relationships are preserved. To uncover these we must look at those network features which are conserved and those which are not.

A number of interesting results can be derived from the network alignment. In this case, we are most interested in the *union* network where we combine the two aligned networks with each vertex in the union corresponding to the pair of mapped vertices from the two networks and the edges from each original network preserved.

12.2.3 Experimental Procedures

Three experiments were conducted to elucidate the effectiveness of the network alignment methodology in uncovering meaningful structure within the data.

1. The algorithm was applied to align the networks in each of the 3 possible combinations of the 3 datasets.
2. The algorithm was applied to one of the datasets and a randomly generated replacement for the second dataset. In doing so we provide evidence to support the claim that the results of aligning the original data represent meaningful structures.
3. The datasets were perturbed prior to alignment to explore the algorithm's resilience to noisy data.

12.2.3.1 Experiment 1: Normal Alignment

For each of the 3 datasets, **A**, **B** and **Whole**, an initial alignment was computed according to the betweenness centrality of the vertices as described in Sect. 12.2.2.2.

For each pair of the 3 datasets we compute 100 alignments and choose the *majority* alignment as the final alignment. That is, for each vertex of the first input graph, we collate the 100 images of that vertex under the alignments and select the image vertex that appears most often (in statistical terms, the *mode*).

With the majority alignment we compute the union networks.

12.2.3.2 Experiment 2: Aligning to Randomized Input

To establish the likelihood that the outcomes of Sect. 12.2.3.1 are the result of chance, we repeat the experiment but replace one graph with a graph of the same size and order but randomly placed edges.

12.2.3.3 Experiment 3: Perturbed Alignment

One of the difficulties in making such observations however is the density and complexity of the basic data. While an observable phenomena such as customer engagement can relatively easily be measured (or at least approximated), the system giving rise to this high-level phenomena is complex, noisy and difficult to control in an experimental sense. This study presents a methodology that helps to overcome this complexity by extracting from complex data the backbone of relationships between components of the data. This backbone can then be used to develop *data-lead* hypotheses and models.

To assess the effect of perturbation on the algorithm, we generate perturbed datasets, repeat the alignment procedure then compare the results. The data is perturbed by randomly decreasing the response value by 1 with a probability of $\frac{1}{3}$, increasing the response value by one with a probability of $\frac{1}{3}$ and otherwise leaving the value untouched, for every response value in the dataset excluding responses with value 1 and 7 (the extrema).

This perturbation was performed on the **B** dataset to produce the $\mathbf{B}'$ dataset. The network was inferred from this data as with the survey derived data. We then align **A** to $\mathbf{B}'$and **B** to $\mathbf{B}'$.

The robustness of the alignment technique against perturbation was tested using modularity based community detection followed by computing Cramér's V statistic [26] and the Adjusted Rand Index (ARI) [23] to determine the agreement between the modularity clusterings (and hence the preservation of the underlying topological information). As each vertex in the alignment graph represents two vertices from the original graphs, the contingency table was built by counting overlaps of the "left" (domain) component and "right" (codomain) component separately.

12.3 Results

We present the results from the Network Alignment algorithm, compare the different outcomes from the three datasets, study the effects of perturbation to simulate a "noisy" dataset and provide a brief analysis. Finally, the parameterized complexity of the network alignment problem is outlined and proved.

12.3.1 Majority Alignments and Union Graphs

We limit the presentation at this point to observing that in all three pairings, the majority alignment mapped each vertex representing a particular construct element to its counterpart in the second graph representing the same element. The only exceptions (those vertices misaligned) are explained in Sect. 12.3.2. Using the majority alignments, we compute the union networks. The networks are given in Figs. 12.1, 12.2 and 12.3.

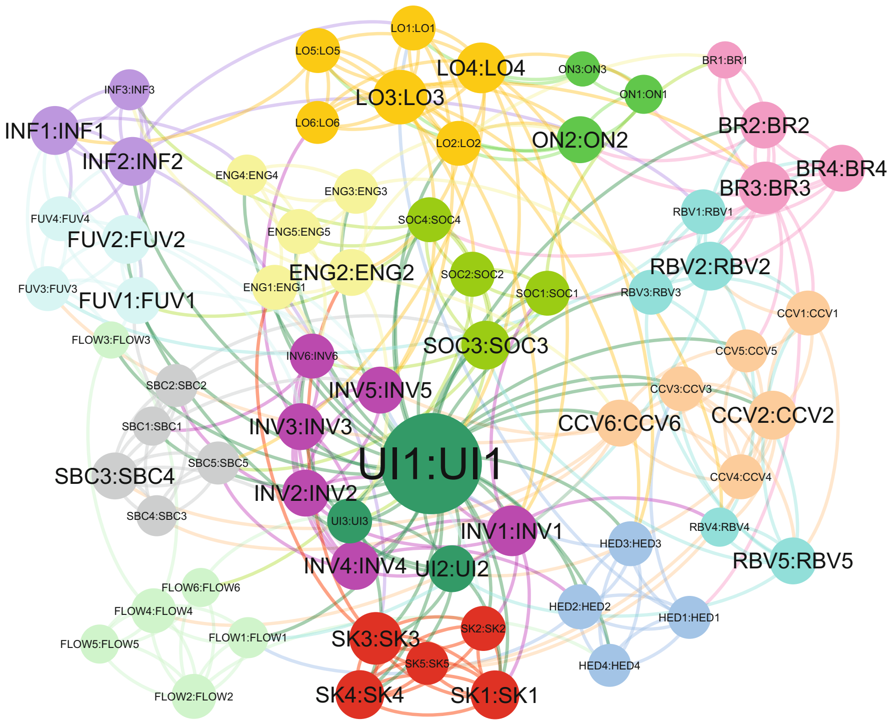


Fig. 12.1 Union network derived from the majority alignment of the **Whole** and **A** networks. The vertices are coloured by theoretical construct. The node size is proportional to the total number of in-degree and out-degree of a node. The arc's colour corresponds to the colour of the source node

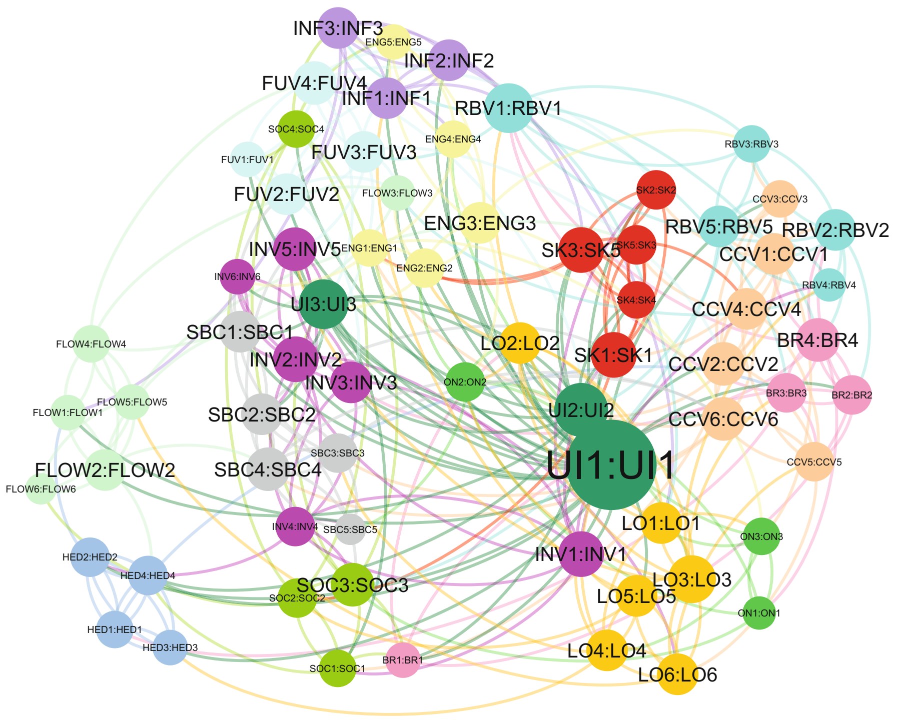


Fig. 12.2 Union network derived from the majority alignments of the **Whole** and **B** networks. The vertices are coloured by theoretical construct. The node size is proportional to the total number of in-degree and out-degree of a node. The arc's colour corresponds to the colour of the source node

12.3.2 Comparison Alignment with Random Graphs

Figures 12.4, 12.5 and 12.6 present the frequency count histograms of the alignment of each vertex over 100 runs of the memetic algorithm for each of the three pairings of networks. Figure 12.7 gives the frequency count histogram for the vertex alignments generated by 100 runs of the memetic algorithm using the **Whole** network and a randomly generated network which has the same number of edges as network **A** (as described in Sect. 12.2.3.2).

The algorithm produces an almost overwhelmingly perfect alignment, with exceptions of two items in the **A** with **B**′ alignment and one item in the **Whole** with **A** alignment. The only "misalignments" happen in the result of dataset **A** with **B**′ where CCV2 is aligned with ENG3 and INV2 with UI1. We note that in the "functional constructs" found through computing modularity, these items are part of the same construct (shown later in Fig. 12.12). The other misalignment is in the result of **Whole** and **A** networks, where SBC3 and SBC4 are interchanged in the mapping, which are both part of the same theoretical construct as well as functional construct in [9]. Also, the raw count data underlying the majority

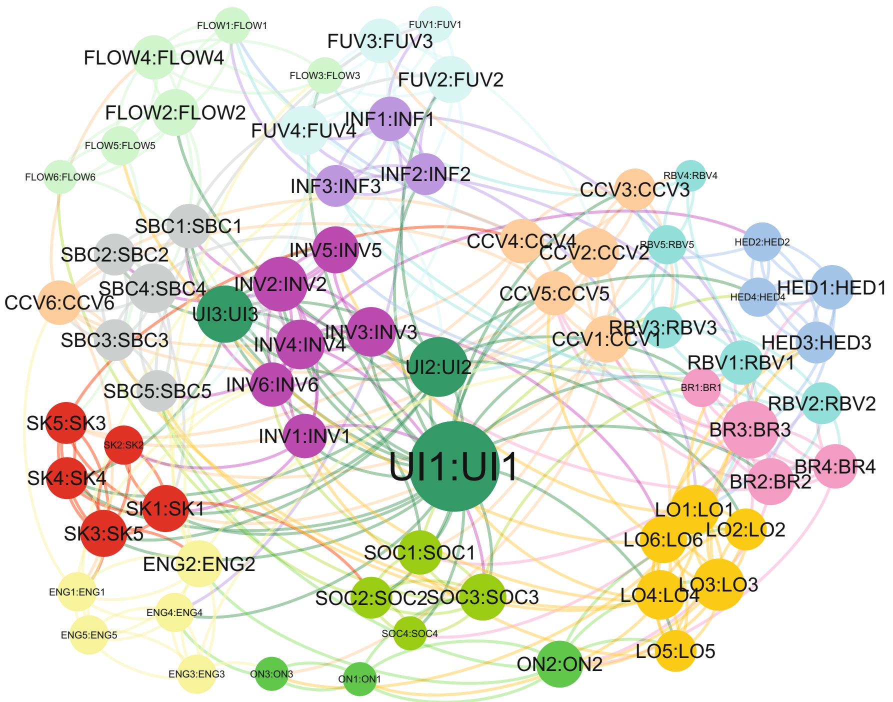


Fig. 12.3 Union network derived from the majority alignment of the **A** and **B** networks. The vertices are coloured by theoretical construct. The node size is proportional to the total number of in-degree and out-degree of a node. The arc's colour corresponds to the colour of the source node

alignment indicates however that SBC3 and SBC4 are essentially interchangeable, with SBC3 mapping to itself in 20 out of 100 alignments and to SBC4 in 24 alignments, and SBC4 mapping to SBC3 in 23 alignments and to SBC4 in 20 alignments.

We note the stark difference between the alignments of the **Whole**, **A** and **B** networks, and the alignment of the **Whole** network and the random network. This supports the assertion that the alignments of the true networks point to genuine structure in the networks and are far from the results of chance alone.

12.3.3 Effects of Perturbation

Figures 12.8 and 12.9 give the union networks derived from the majority alignments of the **A** and **B** networks with the perturbed **B**′ network, respectively. It is clear that the perturbation introduces a notable increase of the number of edges. However when the actual majority alignments are examined, we see that although the

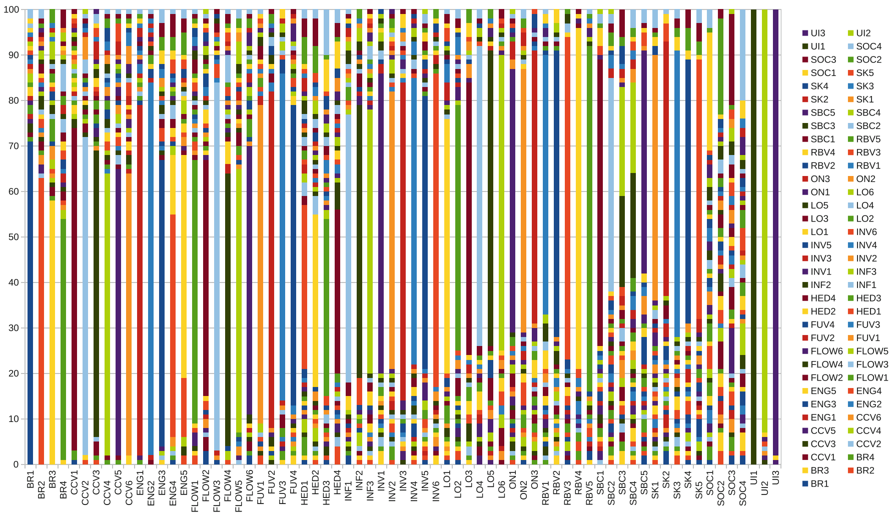


Fig. 12.4 Frequency counts of vertex alignments presented as a stacked histogram for the **Whole** and **A** networks. The data was collected from 100 runs of the memetic algorithm. The counts indicate that, in terms of the value of optimum value of the alignment, each vertex has a clear counterpart, excepting SBC3 and SBC4, which appear interchangeable. The contrast with the distribution of counts when aligning a random graph (q.v. Fig. 12.7) is stark

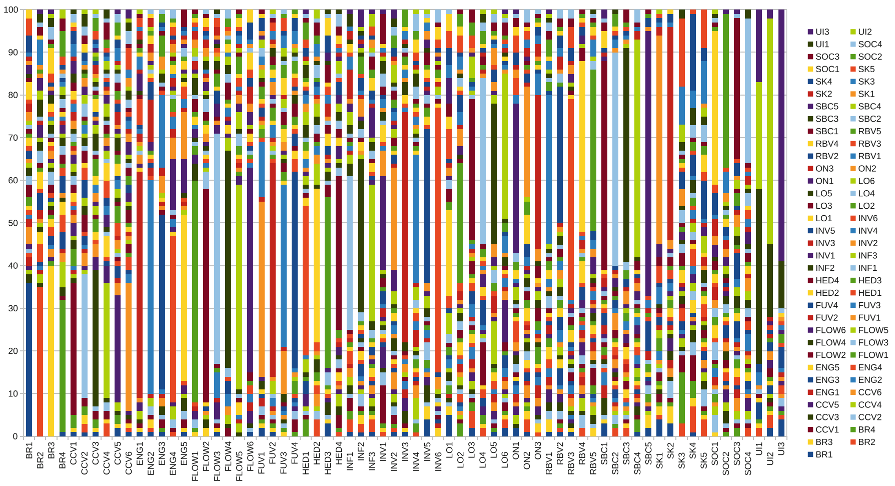


Fig. 12.5 Frequency counts of vertex alignments presented as a stacked histogram for the **Whole** and **B** networks. The data was collected from 100 runs of the memetic algorithm. As with Figs. 12.4 & 12.6, we see a marked difference with the random graph alignment in Fig. 12.7

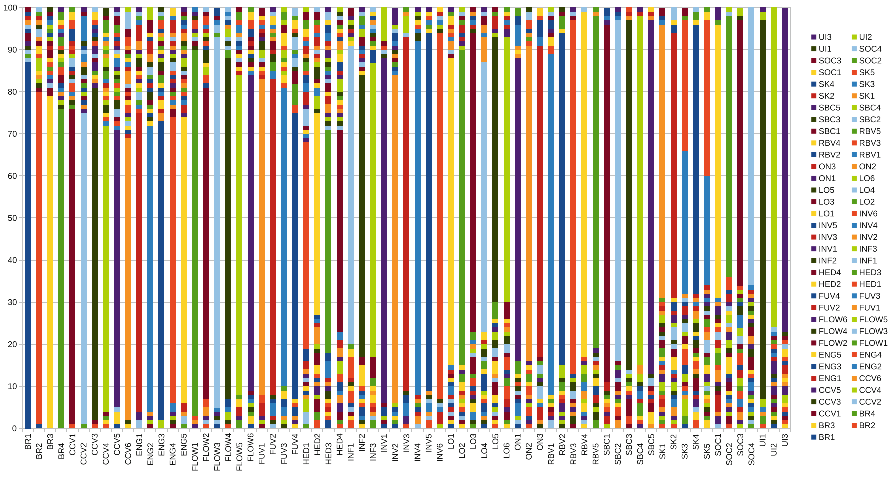


Fig. 12.6 Frequency counts of vertex alignments presented as a stacked histogram for the **A** and **B** networks. The data was collected from 100 runs of the memetic algorithm. We can see that the underlying topological structure of the two networks is being clearly uncovered by the alignment

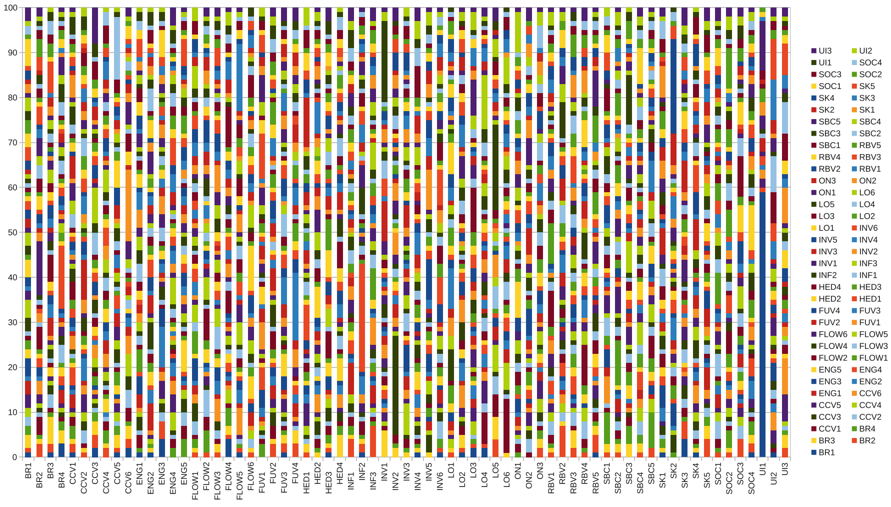


Fig. 12.7 Frequency counts of vertex alignments presented as a stacked histogram for the **Whole** network and a random network with the same number of edges as the **A** network. The alignment was performed 100 times to obtain the counts. The lack of similarity in the topological structures of the two networks is apparent in the essentially random distribution of counts

structure is diffused, the memetic algorithm is still able to detect it with high accuracy. In the alignment of **A** and **B**′, two pairs are switched (CCV2 with ENG2 and INV2 with UI1), but notably these occur as mutual swaps, rather than an extended chain of discrepancies. With the alignment of the **B** and **B**′ networks, we see a perfect mapping of items to themselves, despite the perturbation of approximately 60% of the values.

In Figs. 12.10 and 12.11 we see the effect of the perturbation on the overall alignment. The distinct diagonal band is still apparent but with notably reduced dominance. This gives an informal indication of the significance of the effect of noise in the data, and moreover the robustness of the algorithm in overcoming it.

The results of the clustering via modularity optimization are presented in Tables 12.5 and 12.6, and visually in Figs. 12.12 and 12.13. While the groupings are not perfectly aligned with the theoretical constructs, we note that most theoretical constructs are contained within a single group. Considering the two modularity-based clustering as statistical variables, we can apply some statistical tests to uncover the similarity of the two groupings, and hence indicate the likelihood that the topological information in the network is being recovered despite the perturbation. Cramér's V statistic gives a measure of the intercorrelation of two discrete variables as a value between 0 and 1. Here we take the allocation of the vertices of the union graphs as the variables. The contingency table for the overlap of the two groupings is given in Table 12.2. This gives a Cramér's V statistic value of 0.748, suggesting a reasonable agreement between the two groupings. The ARI measures the similarity of two data clusterings relative to the expected random change of finding any clustering, if the clustering is better than what we would expect from pure chance, the ARI value is positive, between 0 and 1, if the clustering is worse than what we expect from random chance, the ARI value is negative. In this case the ARI is 0.378, suggesting the clustering is notably better than a random assignment. It is also noted that this value is similar to those for the modularity groupings as computed from the raw data in [9], suggesting that the clusterings have as good an agreement as can be expected and moreover that the structure of the networks is still detectable, despite the perturbation.

If the alignment is robust against the presence of perturbations in the data, we would expect that topological information such as modularity based clusterings would be preserved. Although the groupings are not as visually apparent in Figs. 12.8 and 12.9, we can still obtain modularity groupings.

Tables 12.3 and 12.4 give the contingency tables for the overlap between the modularity classes of the perturbed union graphs and the modularity classes derived as functional constructs in [9]. Here we see stronger results for the ARI, suggesting that the functional constructs (i.e. those derived from topological information in the network) are still detectable, despite the perturbation. When comparing the functional constructs extracted from the **Whole** network with the modularity classes computed for the **A** union **B**′ graph (Table 12.5), we obtain a Cramér's V statistic of 0.762 and an ARI of 0.458. With the **Whole** network functional constructs and the **B** union **B**′ modularity classes (Table 12.6), we obtain a Cramér's V statistic of 0.700 and an ARI of 0.409.

Table 12.2 Contingency table for the interrater agreement of the modularity clusterings of the **A** and **B**′ union graph and the **B** and **B**′ union graph

		$\mathbf{B} \cup \mathbf{B}'$ Modularity groups						
		1	2	3	4	5	6	7
$\mathbf{A} \cup \mathbf{B}'$ modularity groups	1	14	4	0	12	0	0	0
	2	10	16	0	0	0	0	0
	3	6	10	8	2	0	0	0
	4	0	0	0	2	0	12	12
	5	0	0	0	0	16	0	0
	6	0	0	10	0	0	0	0

This gives a Cramér's V statistic value of 0.748 and an Adjusted Rand Index of 0.378. Together these suggest that the clusterings embody meaningful topological information

Table 12.3 Contingency table for the interrater agreement of the modularity clusterings of the **Whole** and **A** union **B**′ graphs

		$\mathbf{A} \cup \mathbf{B}'$ modularity groups					
		1	2	3	4	5	6
Whole modularity groups	1	10	20	0	0	0	0
	2	0	8	0	0	0	10
	3	18	0	10	0	0	0
	4	2	0	0	0	16	0
	5	0	0	0	16	0	0
	6	0	0	8	8	0	0
	7	0	0	10	2	0	0

This gives a Cramér's V statistic of 0.762 and an ARI of 0.458. Again, this suggests that the clusters are meaningful and moreover in this case that the NETWORK ALIGNMENT approach copes well with noisy data

12.3.4 Parameterized Complexity of the Network Alignment Problem

In this section we present the formalities of the NETWORK ALIGNMENT problem and we prove tight results for some parameterizations of this problem.

Definition 12.1 (Alignment) An *alignment* between two graphs G_1 and G_2 where $|V(G_1)| \leq |V(G_2)|$ is an injective function $f_{G_1,G_2} : V(G_1) \to V(G_2)$.

Table 12.4 Contingency table for the interrater agreement of the modularity clusterings of the **Whole** and **B** union **B**′ graphs

		B ∪ **B**′ modularity groups						
		1	2	3	4	5	6	7
Whole modularity groups	1	10	10	0	10	0	0	0
	2	0	8	10	0	0	0	0
	3	22	4	0	2	0	0	0
	4	0	0	0	2	16	0	0
	5	0	0	0	2	0	12	2
	6	0	0	8	0	0	0	8
	7	0	10	0	0	0	0	2

In this case we obtain a Cramér's V statistic of 0.700 and an ARI of 0.409, further supporting the NETWORK ALIGNMENT approach's robustness against noise

Table 12.5 **A** ∪ **B**′ modularity clustering groups. The modularity value for this grouping is 0.361424. The labels are of the form "Graph A Vertex:Graph B' Aligned Vertex". Although the groups do not align precisely with theoretical constructs, most constructs are contained, or mostly contained, within a single group

Group 1 (Size: 15)	CCV4:CCV4, CCV5:CCV5, CCV6:CCV6, CCV1:CCV1, CCV2:ENG3, CCV3:CCV3, ON3:ON3, RBV3:RBV3, RBV4:RBV4, RBV5:RBV5, ENG1:ENG1, ENG2:ENG2, ENG3:CCV2, ENG4:ENG4, ENG5:ENG5
Group 2 (Size 14)	INV3:INV3, INV6:INV6, SK2:SK2, INV4:INV4, SK4:SK4, INV5:INV5, INV2:UI1, UI1:INV2, UI3:UI3, INV1:INV1, SK1:SK1, SK3:SK3, SK5:SK5, UI2:UI2
Group 3 (Size 14)	HED4:HED4, BR4:BR4, BR3:BR3, BR2:BR2, BR1:BR1, HED2:HED2, SOC4:SOC4, SOC3:SOC3, SOC2:SOC2, SOC1:SOC1, HED1:HED1, HED3:HED3, RBV2:RBV2, RBV1:RBV1
Group 4 (Size 13)	INF3:INF3, INF2:INF2, INF1:INF1, FLOW5:FLOW5, FLOW4:FLOW4, FLOW6:FLOW6, FLOW1:FLOW1, FLOW3:FLOW3, FLOW2:FLOW2, FUV4:FUV4, FUV1:FUV1, FUV3:FUV3, FUV2:FUV2
Group 5 (Size 8)	ON2:ON2, ON1:ON1, LO2:LO2, LO3:LO3, LO1:LO1, LO6:LO6, LO4:LO4, LO5:LO5
Group 6 (Size 5)	SBC2:SBC2, SBC1:SBC1, SBC3:SBC3, SBC4:SBC4, SBC5:SBC5

Note that we do not actually lose any generality by requiring that $|V(G_1)| \leq |V(G_2)|$, as we can simply add isolated, dummy vertices to $V(G_2)$. Where context allows, we will omit the subscripts.

Definition 12.2 (Value of an Alignment) Given two graphs G_1 and G_2, the value of an alignment f, denoted val f, is defined as

$$\operatorname{val} f = \sum_{u,v \in V(G_1)} \tau_f(u, v)$$

Table 12.6 $\mathbf{B} \cup \mathbf{B}'$ modularity clustering groups. The modularity value for this grouping is 0.338813. The labels are of the form "Graph B Vertex:Graph B' Aligned Vertex". Again we see that most theoretical constructs are contained within a single group

Group 1 (Size: 16)	BR4:BR4, BR3:BR3, BR2:BR2, CCV4:CCV4, CCV5:CCV5, CCV6:CCV6, CCV1:CCV1, CCV3:CCV3, SK2:SK2, SK4:SK4, RBV3:RBV3, SK1:SK1, SK3:SK3, RBV2:RBV2, SK5:SK5, RBV5:RBV5
Group 2 (Size 16)	BR1:BR1, SOC4:SOC4, SOC3:SOC3, SOC2:SOC2, SOC1:SOC1, INV3:INV3, CCV2:CCV2, INV6:INV6, INV4:INV4, INV5:INV5, INV2:INV2, UI1:UI1, UI3:UI3, INV1:INV1, RBV4:RBV4, UI2:UI2
Group 3 (Size 9)	HED4:HED4, HED2:HED2, SBC2:SBC2, HED1:HED1, SBC1:SBC1, HED3:HED3, SBC3:SBC3, SBC4:SBC4, SBC5:SBC5
Group 4 (Size 8)	INF3:INF3, ON3:ON3, RBV1:RBV1, ENG1:ENG1, ENG2:ENG2, ENG3:ENG3, ENG4:ENG4, ENG5:ENG5
Group 5 (Size 8)	ON2:ON2, ON1:ON1, LO2:LO2, LO3:LO3, LO1:LO1, LO6:LO6, LO4:LO4, LO5:LO5
Group 6 (Size 6)	INF2:INF2, INF1:INF1, FUV4:FUV4, FUV1:FUV1, FUV3:FUV3, FUV2:FUV2
Group 7 (Size 6)	FLOW5:FLOW5, FLOW4:FLOW4, FLOW6:FLOW6, FLOW1:FLOW1, FLOW3:FLOW3, FLOW2:FLOW2

where $\tau_f : V(G_1) \times V(G_1) \to \mathbb{N}$ is given by

$$\tau_f(u, v) = \begin{cases} 1 \text{ if } uv \in E(G_1) \text{ and } f(u)f(v) \in E(G_2) \\ 0 \text{ otherwise} \end{cases}$$

We note that this is possibly the simplest (sensible) valuation of an alignment and much more complex ones are possible. For example, we can introduce a term that favours the alignment of particular pairs of vertices, or gives different weights for each aligned edge or vertex and of course non-integral and non-increasing (even non-monotone) valuations are possible. We believe however this simple version captures the basic practical usages but allows the avoidance of notational clutter in the complexity analysis. Within reason, the complexity ramifications of more complex valuations can be easily extrapolated from the basic form.

We observe informally that it is a non-trivial valuation of the edges that seems to give rise to the complexity (or more precisely, valuations that are dependent on more than one vertex and its image under the valuation). For example, given a labelled graph, the valuation that simply adds 1 for each label which matches the label of its image makes the NETWORK ALIGNMENT problem almost trivially polynomial.

We can now state the decision version of the NETWORK ALIGNMENT problem[7]:
NETWORK ALIGNMENT

[7]For those unfamiliar and interested in the technicalities of complexity theory, see [13, 16, 19].

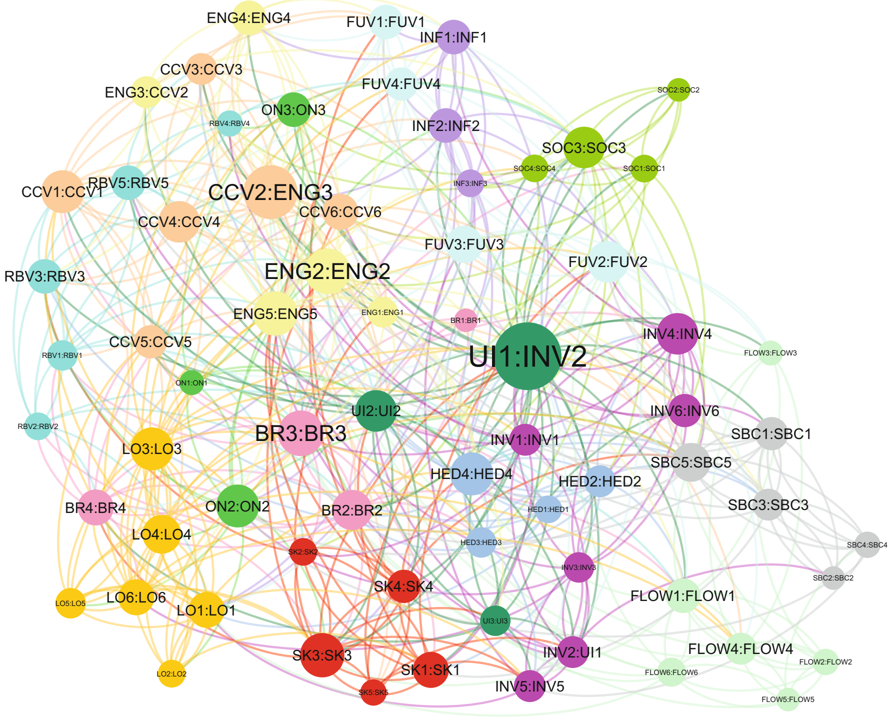


Fig. 12.8 Union network derived from the majority alignment of the **A** and perturbed **B**′ networks. The colours correspond to the theoretical constructs of the domain vertex in the mapped pair. The node size is proportional to the total number of in-degree and out-degree of a node. The arc's colour corresponds to the colour of the source node

Instance Two graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ with $|V_1| \leq |V_2|$, positive integer b.

Question Is there an alignment f with val $f \geq b$?

We note that the definition of an alignment, the value of an alignment and the NETWORK ALIGNMENT problem require no special treatment if we move to directed graphs.

The NETWORK ALIGNMENT problem is NP-complete [25]. We show that NETWORK ALIGNMENT is also intractable even in the stronger Parameterized Complexity sense. This complexity suggests that exact methods are unsuitable for solving the NETWORK ALIGNMENT problem and that a heuristic approach is sensible.

Theorem 12.1 NETWORK ALIGNMENT *is* W[1]*-complete when parameterized by*

– $|V(G_1)|$,

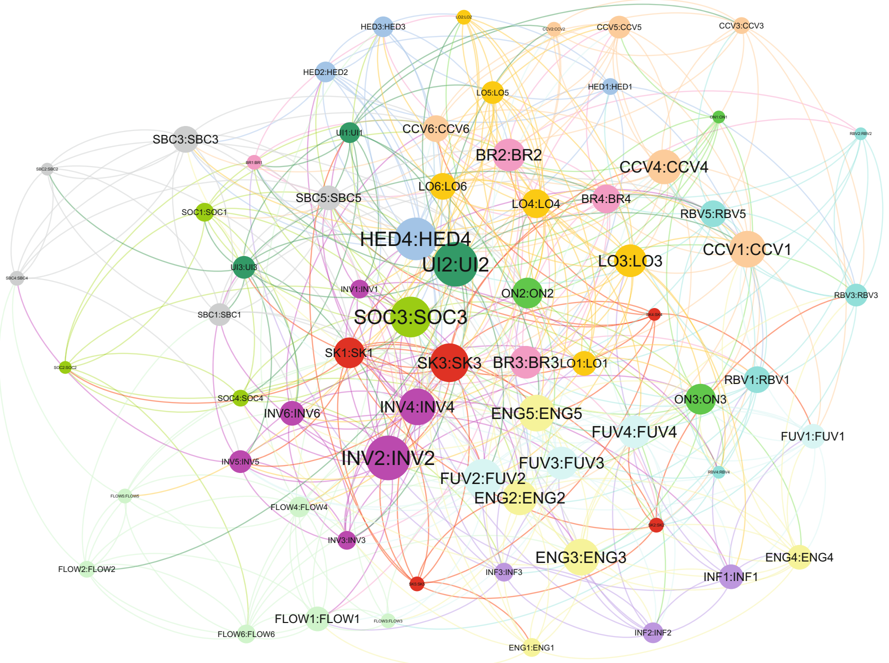


Fig. 12.9 Union network derived from the majority alignment of the **B** and perturbed **B**′ networks. The vertices are coloured according to the theoretical construct of the domain vertex in the mapped pair. The node size is proportional to the total number of in-degree and out-degree of a node. The arc's colour corresponds to the colour of the source node

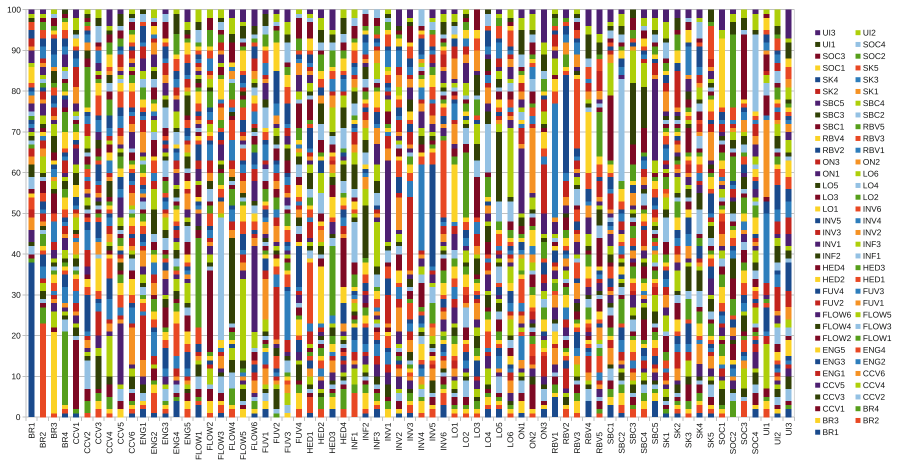


Fig. 12.10 Frequency counts of vertex alignments presented as a stacked histogram for the **A** and perturbed **B**′ networks. The data was collected from 100 runs of the memetic algorithm

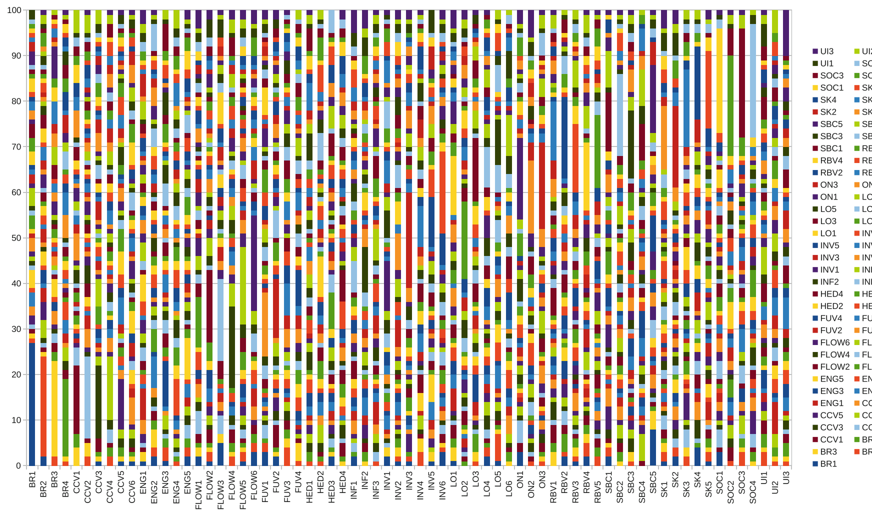


Fig. 12.11 Frequency counts of vertex alignments presented as a stack histogram for the **B** and perturbed **B**′ networks. The data was collected from 100 runs of the memetic algorithms

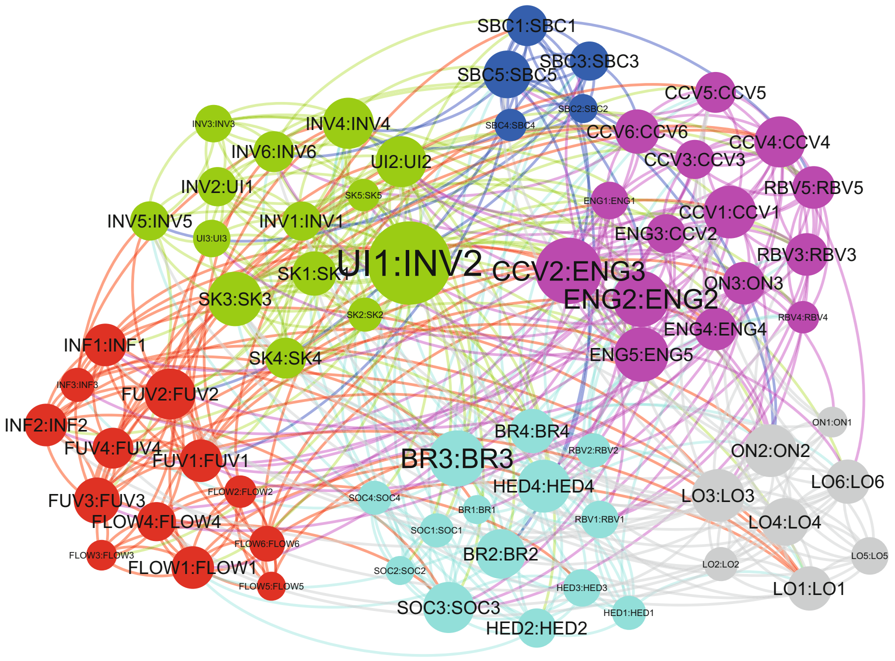


Fig. 12.12 Union network derived from the majority alignment of the **A** and perturbed **B**′ networks coloured by modularity-based community detection. The node size is proportional to the total number of in-degree and out-degree of a node. The arc's colour corresponds to the colour of the source node

– $|E(G_1)|$,
– b,

or any combination of these parameters.

We prove the inclusion and hardness separately.

Lemma 12.1 Network Alignment *is* W[1]*-hard when parameterized by*

– $|V(G_1)|$,
– $|E(G_1)|$,
– b,

or any combination of these parameters.

Proof The reduction is from the well known W[1]-complete problem k-Clique [13].

Given an instance (G, k) of k-Clique we construct an instance (G_1, G_2, b) of Network Alignment as follows:

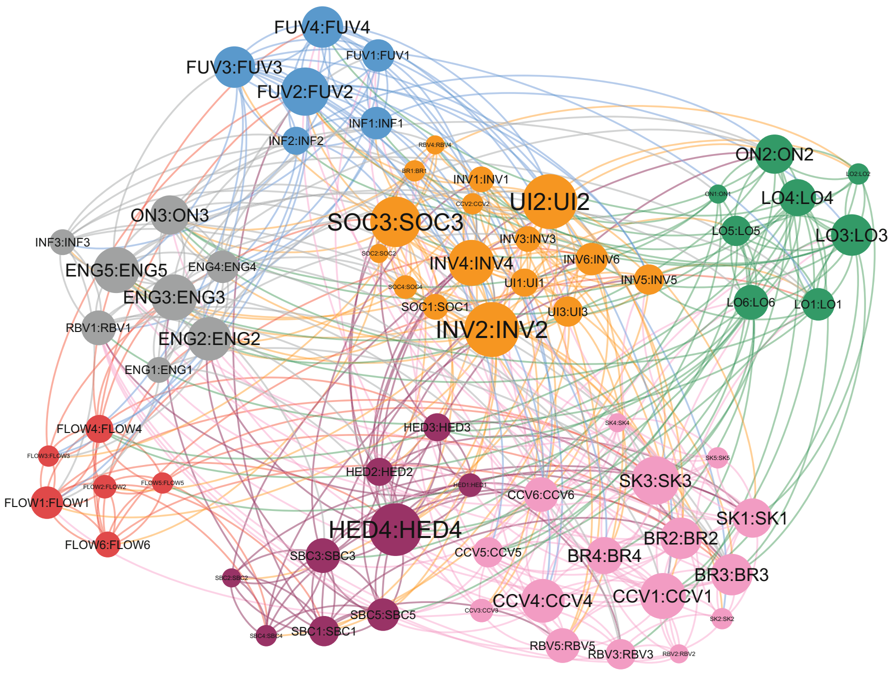


Fig. 12.13 Union network derived from the majority alignment of the **B** and perturbed **B**′ networks coloured by modularity-based community detection. The node size is proportional to the total number of in-degree and out-degree of a node. The arc's colour corresponds to the colour of the source node

– Set $G_1 = K_k$,
– Set $G_2 = G$, and
– Set $b = \binom{k}{2}$.

Clearly this reduction can be performed in polynomial time and also preserves the parameter.

Claim If (G, k) is a YES instance of k-CLIQUE, then (G_1, G_2, b) is a YES instance of NETWORK ALIGNMENT.

Let $U = \{u_1, \ldots, u_k\} \subseteq V(G)$ be the set of vertices forming the witness for the fact that (G, k) is a YES instance of k-CLIQUE. By construction these vertices also appear in G_2. Let $V(G_1) = \{v_1, \ldots, v_k\}$ be the vertices of the k-clique G_1. We can construct an alignment f between G_1 and G_2 as by simply setting $f(v_i) = u_i$ for each i (note that the choice of ordering on the vertices is unimportant).

As U induces a clique in G_2, we have $u_i u_j \in E(G_2)$ for each pair $i, j \in [1, k]$. Thus for every $v_i, v_j \in V(G_1)$ we have $\tau_f(v_i, v_j) = 1$ and hence as $|E(G_1)| = \binom{k}{2}$ we immediately obtain $\text{val}\, f = \binom{k}{2} = b$. Hence (G_1, G_2, b) is a YES instance of NETWORK ALIGNMENT.

Claim If (G_1, G_2, b) is a YES instance of NETWORK ALIGNMENT, then (G, k) is a YES instance of k-CLIQUE.

Let f be an alignment of G_1 and G_2 with $\text{val}\, f \geq b = \binom{k}{2}$ and let $U = \{u_1, \ldots, u_k\} \subseteq V(G_2)$ be the image of f. Without loss of generality, we may assume that $V(G_1) = \{v_1, \ldots, v_k\}$ and that $f(v_i) = u_i$ for each i. As $|V(G_1)| = k$, it is clear that $\text{val}\, h \leq \binom{k}{2}$ for any alignment h of G_1, regardless of G_2, thus we have that $\text{val}\, f = \binom{k}{2} = b$. Given that G_1 has k vertices, achieving this value requires that $\tau_f(v_i, v_j) = 1$ for each pair $i, j \in [1, k]$. Therefore $f(v_i)f(v_j) = u_i u_j \in E(G_2)$ for all $i, j \in [1, k]$. Hence U induces a k-clique in G_2. As $G_2 = G$, we have that (G, k) is a YES instance of k-CLIQUE.

Combining the two previous claims we have that NETWORK ALIGNMENT is W[1]-hard when parameterized by $|V(G_1)|$. To complete the proof of Lemma 12.1, we observe that $|V(G_1)|$, $|E(G_1)|$ and b are all functions of the parameter k of k-CLIQUE, hence any combination thereof is also a function of k alone. □

Lemma 12.2 NETWORK ALIGNMENT *is in* W[1] *when parameterized by*

– $|V(G_1)|$,
– $|E(G_1)|$,
– *b*,

or any combination of these parameters.

Proof To prove containment, we reduce NETWORK ALIGNMENT to MC(Σ_1) which is canonically W[1]-complete [15].

The parameterized MODEL CHECKING (MC) problem is the problem of, given a logical structure and a formula in some specified logic, deciding whether the structure satisfies the formula. Σ_t is a fragment of first order logic with only existentially quantified variables (for a detailed treatment see [16]). In this context, a graph is a suitable logical structure and thus to complete the reduction, we need only construct an existentially quantified first order formula that expresses the problem.

Let (G_1, G_2, b) be an instance of NETWORK ALIGNMENT, we construct an instance $(\mathcal{A}, \varphi)$ of MC(Σ_1). The relational structure $\mathcal{A}$ is constructed similarly to the standard relational structure for graphs, except we duplicate the vocabulary to separately identify the substructures corresponding to the two graphs. Thus the universe A of the structure is $V(G_1) \cup V(G_2)$, and the vocabulary $\tau = \{V_1, V_2, E_1, E_2\}$ where $V_i v$ is true if $v \in V(G_i)$ and $E_i uv$ is true if $uv \in E(G_i)$ for $i \in \{1, 2\}$. NETWORK ALIGNMENT can then be expressed by the following first-order formula φ:

$$\begin{aligned} F = \{F_x \in F_c : (&|S| > |C|) \\ &\cap (\text{minPixels} < |S| < \text{maxPixels}) \\ &\cap (|S_{\text{conected}}| > |S| - \epsilon)\} \end{aligned} \tag{12.1}$$

$$\begin{aligned} \varphi = \exists_{i\in[1,b],j\in[1,2]} u_{i,j}, v_{i,j} \Bigg(\Bigg(& \bigwedge_{i\in[1,b],j\in[1,2]} \big(V_1 u_{i.j} \wedge V_2 v_{i,j}\big) \Bigg) \wedge \\ & \Bigg(\bigwedge_{1\in[1,b]} \big(u_{i,1} \neq u_{i,2} \wedge v_{i,1} \neq v_{i,2}\big) \Bigg) \wedge \\ & \Bigg(\bigwedge_{i<j\in[1,b]} \big(u_{i,1} \neq u_{j,1} \vee u_{i,2} \neq u_{j,2}\big) \wedge \big(u_{i,2} \neq u_{j,1} \vee u_{i,1} \neq u_{j,2}\big) \Bigg) \wedge \\ & \Bigg(\bigwedge_{i<j\in[1,b],k\in[1,2]} \big(u_{i,k} = u_{j,k} \leftrightarrow v_{i,k} = v_{j,k}\big) \Bigg) \wedge \\ & \Bigg(\bigwedge_{i\in[1,b]} \big(E_1 u_{i,1} u_{i,2} \wedge E_2 v_{i,1} v_{i,2}\big) \Bigg) \Bigg) \end{aligned}$$

Informally, φ states that there are b pairs of vertices in each graph, where no two pairs are identical, that if two vertices from G_1 are the same, then so are the corresponding vertices from G_2 (and vice versa) and that there is an edge between each pair in G_1 and the corresponding pair in G_2. Clearly $\varphi \in \Sigma_1$ and $|\varphi| \leq g(b)$ for some function g.

The reduction thus consists only of translating G_1 and G_2 into $\mathcal{A}$ and generating φ for the given b, which can be achieved in polynomial time. As noted $|\varphi|$ is a function of b, so the parameter is preserved.

Claim If (G_1, G_2, b) is a YES instance of NETWORK ALIGNMENT, then $(\mathcal{A}, \varphi)$ is a YES instance of MC(Σ_1).

If (G_1, G_2, b) is a YES instance of NETWORK ALIGNMENT, there exists some alignment f with val $f \geq b$. As val $f \geq b$, there exists at least b distinct edges in G_1 that map to edges in G_2. More particularly there are $2b$ not necessarily distinct vertices in G_1 that induce those edges with corresponding vertices in G_2. Let $E_u = \{u_{1,1}u_{1,2}, \ldots, u_{b,1}u_{b,2}\} \subseteq E(G_1)$ be the set of b edges in some arbitrary order, and via slight abuse of notation, let $f(E_u)$ be the corresponding set of edges from G_2. Clearly φ is satisfied by assigning the endpoints of the edges to the variables of the formula according to the chosen ordering.

Claim If $(\mathcal{A}, \varphi)$ is a YES instance of MC(Σ_1), then (G_1, G_2, b) is a YES instance of NETWORK ALIGNMENT.

If $\varphi(\mathcal{A})$ is satisfied (i.e. $\mathcal{A} \models \varphi$), then there are b pairs of vertices from each graph that are distinct (as pairs) and each induces an edge. We can then construct an alignment f by mapping $f(u_{i,j}) = v_{i,j}$. This immediately gives val $f \geq b$.

The above claims together prove the soundness of the reduction.

Finally we observe that $b \leq |E(G_1)| \leq |V(G_1)|^2$, therefore NETWORK ALIGNMENT remains in W[1] under the alternate parameterizations. □

Combining Lemmas 12.1 and 12.2 we obtain the proof of Theorem 12.1.

12.4 Conclusions

In this chapter we had several aims. From a business and data analytics perspective, we showed that the problem of integrating information from different surveys (or other consumer behaviour datasets) can be seen as a problem in network analytics. Using real data and simulated perturbations on the responses of consumers, we have shown that underlying behavioural structures that can be inferred from individual surveys are "conserved" across different populations. In addition, we have shown that this also happens under reasonable variations on the responses measured on a Likert scale. This models the natural expected range on variations on individual responses when a scale that goes between 0 and 7 is chosen to quantify the answers.

From a computational perspective, we have shown that the use of network alignment and community detection together they help to identify the variables that have the same role in each of the behavioural models. The community detection method also helps to identify clusters of these variables which represent what we have in the past denominated as "functional constructs". In this study, the functional constructs generally combine multiple theoretical constructs. For instance, combining measured variables of engagement, co-creation value and relationship-building value (shown in purple in Fig. 12.12).

This chapter also brings a new result for the field of computational complexity. We have presented the first characterization in terms of parameterized complexity of the NETWORK ALIGNMENT problem. We have shown that the problem is W[1]-complete, meaning that it is currently believed that it is unlikely that a fixed-parameter algorithm exists for it unless FPT=W[1].

To the best of our knowledge, this is the first application of NETWORK ALIGNMENT in the area of computational social science and we have given an example of its application in the analysis of networks derived from the study of reverse engineering consumer behaviour. We have shown that NETWORK ALIGNMENT is a valuable tool for identifying questions that have the same role in surveys conducted with two independent groups of respondents. In addition, we looked at the robustness of this method. Using a perturbation-based approach we have generated a further "synthetic" test-set scenario which showed a remarkable consistency in the identification of questions using the networks derived via the reverse-engineering of behaviour approach proposed in [9]. We have also proposed a method based on memetic algorithms which can handle large instances of these problems.

Taken together, these constitute promising results and open new lines of investigation. By applying the approach proposed in [9] to reverse engineer models that link the responses of questions to two different surveys, NETWORK ALIGNMENT can be subsequently used to "integrate" the responses and to identify isomorphisms between the questions in both surveys. As a consequence NETWORK ALIGNMENT can be used as a tool for identifying questions that may have "similar roles" within the surveys and help to amalgamate the answers. This, in turn, would help behaviour theoreticians to identify new "functional constructs" [9] which could emerge from the understanding of groups of variables that have been isomorphically mapped by the alignment and that consistently appear together as a result of a modularity-based or other community detection algorithm.

Further research needs to be conducted to enhance the scalability of the approach to face the challenges of larger networks, since new automation technologies could easily scale the number of variables to several thousands. An advantage of the method proposed to deliver an approximate solution to the optimization problem, i.e. the memetic algorithm, is that it has an intrinsic parallelism which would fit very well with implementations that exploit heterogeneous distributed computing systems. Another area of future research is the development of *complete anytime* memetic algorithms, which are methods that generate *provably optimal* solutions via at least one exact solver being incorporated in the population. From the theoretical computer science perspective, it seems logical that further research should be conducted to analyse the parameterized complexity of the NETWORK ALIGNMENT problem under some structural graph parametrizations. This would motivate a more constrained approach for the creation of the networks which in turn will boost the scalability of the techniques when applied in tandem. Finally, while the results have been presented in the area of computational social science, this is a method of general applicability and we expect its adoption in other data-driven studies involving network analytics.

Acknowledgements Pablo Moscato acknowledges previous support from the Australian Research Council Future Fellowship FT120100060 and Australian Research Council Discovery Projects DP120102576 and DP140104183.

References

1. J. Abello, A. L. Buchsbaum, and J. R. Westbrook. A functional approach to external graph algorithms. *Algorithmica*, 32(3):437–458, 2002.
2. Giovanni Acampora, Pasquale Avella, Vincenzo Loia, Saverio Salerno, and Autilia Vitiello. Improving ontology alignment through memetic algorithms. In *Proceedings of the IEEE International Conference on Fuzzy Systems*, pages 1783–1790. IEEE, 2011.
3. M. Barbany, H. Gutiérrez de Terán, F. Sanz, and J. Villà-Freixa. Towards a mip-based alignment and docking in computer-aided drug design. *Proteins*, 56(3):585–594, 2004.
4. J. Carlson and A. O'Cass. Creating commercially compelling website-service encounters: an examination of the effect of website-service interface performance components on flow experiences. *Electronic Markets*, 21:237–253, 2011.
5. J. Carlson and A. O'Cass. Optimizing the online channel in professional sport to create trusting and loyal consumers: The role of the professional sports team brand and service quality. *Journal of Sport Management*, 26:463, 2012.
6. Stephen A. Cook. The complexity of theorem-proving procedures. In *Proceedings of the Third Annual ACM Symposium on Theory of Computing*, pages 151–158. ACM, 1971.
7. G. A. Croes. A method for solving traveling salesman problems. *Operations Research*, 6:791–812, 1958.
8. I. P. Cvijikj and F. Michahelles. Online engagement factors on Facebook brand pages. 3(4):843–861, 2013.
9. Natalie J de Vries, Jamie Carlson, and Pablo Moscato. A data-driven approach to reverse engineering customer engagement models: Towards functional constructs. *PLoS ONE*, 9(7), 2014.
10. Natalie J de Vries, Ahmed S Arefin, and Pablo Moscato. Gauging heterogeneity in online consumer behaviour data: A proximity graph approach. In *2014 IEEE Fourth International Conference on Big Data and Cloud Computing (BdCloud)*, pages 485–492. IEEE, 2014.
11. Remco Dijkman, Marlon Dumas, and Luciano García-Bañuelos. *Business Process Management: 7th International Conference, BPM 2009, Ulm, Germany, September 8–10, 2009. Proceedings*, chapter Graph Matching Algorithms for Business Process Model Similarity Search, pages 48–63. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.
12. Remco Dijkman, Marlon Dumas, and Luciano García-Bañuelos. *Graph Data Management: Techniques and Applications*, chapter Business Process Graphs: Similarity Search and Matching, pages 421–437. ICI Global, Hershey, 2012.
13. Rodney G. Downey and Michael R. Fellows. *Fundamentals of Parameterized Complexity*. Texts in Computer Science. Springer, 2013.
14. S. P. Ficklin and F. A. Feltus. Gene coexpression network alignment and conservation of gene modules between two grass species: maize and rice. *Plant Physiology*, 156(3):1244–56, 2011.
15. Jörg Flum and Martin Grohe. Fixed-parameter tractability, definability, and model checking. *SIAM Journal on Computing*, 31(1):113–145, 2001.
16. Jörg Flum and Martin Grohe. *Parameterized Complexity Theory*. Texts in Theoretical Computer Science. Springer, 2006.
17. Félix-Antoine Fortin, François-Michel De Rainville, Marc-André Gardner, Marc Parizeau, and Christian Gagné. DEAP: Evolutionary algorithms made easy. *Journal of Machine Learning Research*, 13:2171–2175, Jul 2012.
18. Linton Freeman. A set of measures of centrality based on betweenness. *Sociometry*, 40:35–41, 1977.

19. Michael R. Garey and David S. Johnson. *Computers and intractability: A guide to the theory of NP-completeness*. W. H. Freeman and Co., San Francisco, Calif., 1979.
20. David E. Goldberg and Robert Lingle Jr. Alleles, loci, and the traveling salesman problem. In John J. Grefenstette, editor, *Proceedings of the First International Conference on Genetic Algorithms and Their Applications*. Lawrence Erlbaum Associates, Publishers, 1985.
21. N. Hohenstein, M. J. Sirgy, A. Herrmann, and M. Heitmann. Self-congruity: Antecedents and consequences. In *34th La Londe International Research Conference in Marketing Communications and Consumer Behaviour*, pages 118–130, Aix en Provance, France, 2007.
22. Yannick Hold-Geoffroy, Olivier Gagnon, and Marc Parizeau. Once you SCOOP, no need to fork. In *Proceedings of the 2014 Annual Conference on Extreme Science and Engineering Discovery Environment*, page 60. ACM, 2014.
23. L. Hubert and P. Arabie. Comparing partitions. *Journal of Classifications*, 2:193–218, 1985.
24. B. Jahn and W. Kunz. How to transform consumers into fans of your brand. *Journal of Service Management*, 23(3):344–361, 2012.
25. Gunnar W. Klau. A new graph-based method for pairwise global network alignment. *BMC Bioinformatics*, 10(Suppl 1):S59, 2009.
26. A. M. Liebetrau. *Measures of Association*. SAGE Publications, Inc., Thousand Oaks, CA, 1983.
27. J. Marenco. Un algoritmo branch-and-cut para el problema de mapping. Master's thesis, Departamento de Computación, Universidade de Buenos Aires, 1999.
28. A. O'Cass and J. Carlson. Examining the effects of website-induced flow in professional sporting team websites. *Internet Research*, 20:115–134, 2010.
29. A. O'Cass and J. Carlson. An empirical assessment of consumers' evaluations of web site service quality: Conceptualizing and testing a formative model. *Journal of Services Marketing*, 26:419–434, 2012.
30. A. O'Cass and L. Ngo. Examining the firm's value creation process: A managerial perspective of the firm's value offering strategy and performance. *British Journal of Management*, 22:646–671, 2011.
31. J. van Doorn, K. N. Lemon, V. Mittal, S. Nass, D. Pick, P. Pirnir, and P. C. Verhoef. Customer engagement behavior: Theoretical foundations and research directions. *Journal of Service Research*, 13(3):253–266, 2010.
32. Autilia Vitiello. *Memetic algorithms for ontology alignment*. PhD thesis, Universita degli studi di Salerno, 2012.
33. Xingsi Xue and Yuping Wang. Optimizing ontology alignments through a memetic algorithm using both matchfmeasure and unanimous improvement ratio. *Artificial Intelligence*, 223(0):65–81, 2015.

Part IV
Memetic Algorithms

Chapter 13
Memetic Algorithms for Business Analytics and Data Science: A Brief Survey

Pablo Moscato and Luke Mathieson

Abstract This chapter reviews applications of Memetic Algorithms in the areas of business analytics and data science. This approach originates from the need to address optimization problems that involve combinatorial search processes. Some of these problems were from the area of operations research, management science, artificial intelligence and machine learning. The methodology has developed considerably since its beginnings and now is being applied to a large number of problem domains. This work gives a historical timeline of events to explain the current developments and, as a survey, gives emphasis to the large number of applications in business and consumer analytics that were published between January 2014 and May 2018.

Keywords Artificial intelligence · Data analytics · Data clustering · Machine learning · Memetic Algorithm

13.1 Introduction

Memetic Algorithms (MAs) is a field in constant expansion and evolution. This strategy for problem solving has a rich history that covers nearly three decades of investigations; see [300] for a personal account of the many research paths that have opened since the coalescence of the field. Several introductory tutorials, surveys and

P. Moscato (✉)
School of Electrical Engineering and Computing, The University of Newcastle, Callaghan, NSW, Australia
e-mail: Pablo.Moscato@newcastle.edu.au

L. Mathieson
School of Electrical Engineering and Computing, The University of Newcastle, Callaghan, NSW, Australia

School of Software, University of Technology Sydney, Ultimo, NSW, Australia
e-mail: luke.mathieson@uts.edu.au

even books have been published on this subject over many years, e.g. [18, 84, 101, 173, 218, 298, 301, 324, 326, 341–343] are reviews that provide references to a large number of previous manuscripts in the area.

The subject has also been addressed with different perspectives in several Encyclopaedias such as those of Operations Research and Management Science [313], Electrical and Electronics Engineering [70], and the Handbooks of Metaheuristics [303], and Approximation Algorithms and Metaheuristics [302] and other important collections in Engineering and Operations Research [308–310].

The MAs field continues to grow considerably and its wide impact has escalated in the past two decades. Currently, around 2000 academic papers have been published. We estimate that approximately 300 new articles appear each year, indicating the vitality of the technique. Restricting this review to applications on Business and Data Analytics has been timely, as the number of applications to this area is also skyrocketing. While this survey concentrates on the last few years, it also contains important links to previous literature in the field that some researchers may not be fully aware of. This said, we already offer an apology to the many other manuscripts that we have not been able to cover in a single book chapter. We are confident though that the references here would allow readers to guide themselves to other important reviews and works which are already available and can be indirectly identified, thanks to our references.

Like "Evolutionary Computation", the denomination that was proposed to name a field dedicated to the study of algorithms that are evolutionary in nature, and with an analogous intent, researchers proposed *"Memetic Computing"* as the field of study of Memetic Algorithms (MAs) and their many instantiations of the paradigm. Springer, the same company that is publishing this book, was the first international group to recognize and capitalize from the growing interest in the technique and, consequently, it created a journal with the same name back in 2007; the first issue [236] was published in 2009, almost a decade before the publication of this book.

The field is nowadays more vibrant than ever and new papers are appearing in the literature on a daily basis.

Thus it is nigh impossible to give a comprehensive view of the existing literature on Memetic Algorithms in a single book and entirely impossible to do it in a single chapter. One of these authors' last attempts to give a comprehensive view was Springer's *"Handbook of Memetic Algorithms"* [326] but since then the rate of occurrence of new publications in the literature has dramatically accelerated.

In 2013, the field of Memetic Computing (together with the highly related approach of Differential Evolution) was recognized by Thomson Reuters as *"One of the top 10 ranked research fronts in the combined areas of Mathematics, Computer Science and Engineering"*. The selection was made among 8000 research fronts currently found in Thomson Reuters Essential Science Indicators (ESI) database (see [210]).

This is an important feat for a technique with humble beginnings but great expectations. Its roots can be traced to work in the late 1980s, when one of the authors, working together with and M.G. Norman, at the California Institute of Technology while the latter was visiting the same group, proposed a new metaheuristic approach for the Travelling Salesman Problem (TSP) [305, 336] which was specifically designed to most effectively exploit the supercomputing hardware available to them at the time. The TSP is a classical combinatorial optimization problem in which the task is to find, given a complete matrix of non-negative integer distances between a set of cities, the shortest route that starts from one city and visits all the cities in the set while returning to the origin city. The history of MAs, which starts with this motivating problem (from the area of Transportation and Logistics but that is also at the absolutely core body of knowledge of both Computer Science and Discrete Applied Mathematics), is well described in [300].

Over the last three decades Moscato and Norman's "competitive and cooperative approach" for memetic computing on the TSP has inspired many others to use the technique on the same problem [275, 279], and its variants and generalizations (e.g. the asymmetric TSP [61], the symmetric Multi-objective TSP [392], the generalized TSP [53, 166], the fuzzy road transport TSP [59, 135], the dynamic TSP [268], the selective TSP on a road network [269, 365], the minmax multiple TSP [473]). Other contributions in the area come from a variety of sources and mixed techniques (e.g. [9, 65, 103, 126, 179, 212, 213, 247, 345, 470, 473]).

13.2 Memetic Algorithms Basics

The key assumption of MAs, or one of its basic working hypotheses, is that the use of a population of computational agents can dramatically accelerate the search process and, at the same time, improve the quality of the feasible solutions found in challenging computational problems. This would be of particular interest for problems for which we do not expect that a polynomial-time (i.e. an efficient) algorithm exists (e.g. NP-hard problems). The key issue is then *how* to achieve this synergy, as it is expected that "the whole should be better than the sum of parts" [296]. It is the job of the algorithmic engineer that designs the MA to come up with a good way of obtaining such a synergy and to prove that it exists (generally via computational experiments).

The computational agents in MAs can be employing the same basic techniques or different ones. So from an algorithmic design perspective, it may be important to have some prior knowledge of good existing methods for the problem to be addressed. Experienced researchers know well that, in general, good MAs come as a result of incorporating a significant amount of "problem domain knowledge" embodied in good algorithms and heuristics [298], so that correlation of local minima and high-quality solutions can be exploited [277, 296, 297]. Reimar Hofmann, as a summer scholar at the Edinburgh Parallel Computing Centre in 1993, co-supervised by Norman and Moscato, developed the first software tool for visualizing multiple trajectories of an MA for the TSP. Using a method based on the technique called Projection Pursuit [142], the trajectories of agents during MA run were optimally displayed; local minima appeared as being closely fit by a paraboloid, indicating the presence of a high-degree of correlation of local optima [178]. More than two decades later, this subject still catches the attention of researchers (see, for instance, the visualization results of [338]).

In their first collaboration, back in 1988, Norman and Moscato carefully chose to define their first memetic approach for the TSP in terms of "agents" [336]. In computing, this word is reserved for software entities that act upon an environment and observe it via some type of sensors. In Artificial Intelligence, an *intelligent agent* directs its activities in the environment towards achieving its goals. Many people see a natural analogy with the denomination of *"rational"* agents in Economics. For MAs, such a restriction is not necessary, e.g. agents could be programmed to perform non-rational decisions, or to exhibit random "altruistic" behaviour, etc. Agents may engage in interactions by exhibiting different *behaviours* [45]. These behaviours could also be dynamically modified at run time informed by the performance of the search process, thus allowing a distributed control mechanism that helps to give extra synergies to the group [296, 297].

One of the most common misconceptions is that MAs are just another form of heuristics which do not guarantee that the final solution obtained by the population has any guarantee of optimality. That is not true. A good example is work conducted in the area of hierarchical clustering for phylogenetics in speed-ups of several orders of magnitude were observed [98]. In fact when, in an MA, one (or more) agents are performing some type of exact search procedure, if one of these methods completes its run, should it deliver a feasible solution of the problem, then it should be the optimum one (or one of the potentially many optima that have the same value of the objective function). There is nothing that restricts MAs to be applied as "just" a heuristic without proof of optimality.

Today, some of the fastest methods for the TSP are based on these kind of MAs. The agents produce partial and complete (feasible) solutions for the problem being considered that bound the value of the optimal solution from above and below, eventually converging to the optimal result and returning a feasible solution (assuming at least one exists).

The early approaches are now considered to be part of the "first generation MAs". For instance, Moscato and Norman's first memetic approach for the TSP was defined as a "cooperative and competitive search". The agents were using the same technique (a metaheuristic known as Simulated Annealing highly popular and well-praised at the time). Periods of competition between the agents were interspersed with periods of cooperation by which the agents would exchange information. Without going too deep into the specific details which have been explained elsewhere, they used a procedure by which agents would "recombine" their solutions with one given by another agent. Their approach was deliberately minimalistic, they wanted to use the TSP as a benchmark for testing their core idea: *would the set of agents* (using specific cooperation and cooperation rules that somewhat restricted the team dynamics) *be able to obtain some sort of "synergistic" effect that would dramatically speed-up the search?*

These first MAs then needed to prove themselves in a highly competitive world of algorithmic and heuristic design. As a consequence, publication in peer-review journals required careful analysis and statistical validation of the claims. In general, referees were convinced that "the whole was better than the sum of parts" when the MAs would achieve significant better solutions (in terms of quality) and at highly reduced computing times. For instance, a "population" Pop of interacting agents (with cooperation, competition and individual search process) running together and "consuming" 1 day of CPU time in a computer should be significantly better than a 1-day run using a single agent and using the single individual search process. This said, these first MAs, for each new problem, had to justify their "existence" in the literature by providing evidence that each component (i.e. cooperation and competition mechanisms, constructive algorithms, iterative and improvement heuristics, etc.) was bringing something to the mix.

At the same time that research on MAs was progressing on some problems, a body of knowledge started to develop that would explain why they were working so well [297] and some key results followed in problems in the business area. They involved scheduling [97], and timetabling events [63, 348]. These early successes galvanized the attention and a number of other papers followed this trend in the new millennium [5, 12, 34, 96, 136–138, 150, 199, 226, 270, 272, 346, 403, 409, 425, 434, 446, 451]. Clearly, the success of MAs in Business applications (like the National Hockey League Timetabling/Scheduling [97]) and the results in "hard timetabling problems in practice" that academics "suffer" and know very well [63, 348] were catalytic for the interest in the community and motivated practitioners all around the world.

13.2.1 Structure of a Basic Memetic Algorithm

A number of other reasons can explain the popularity of MAs. One of them is linked to both code and algorithm reuse. The main structural framework of an MA will constitute code that allows, in some cases, distributed implementation and problem independence. This means that researchers/practitioners can develop a high-quality MA for a problem and then quickly adapt it for another problem. This does not mean that the MA for the new problem is of great quality, and may not be competitive against the state-of-the-art approach, but it could constitute a very good prototype in which to include more problem specific domain knowledge and new types of search operators tailored for this new "environment" for the search agents.

It is then proper to describe a prototypical template for a very basic MA approach (please refer to the template of a Memetic Algorithm, in Algorithm 4). As said before, the first implementations of MAs had to justify that periods of individual optimization, followed by one step in which the individuals exchange information about the results of the searches, followed by individual optimization again culminating with a necessary selection step (to keep the population limited in size), were delivering the required synergies between each algorithmic component.

Researchers quickly adapted methods to control the process, and clever techniques for creating the individual population, reducing the search to a configuration

Algorithm 4: A very basic Memetic Algorithm

```
1  for $j := 1$ to $\mu$ do
2      $pop_j :=$ new SearchAgent;               // initialize the population
3  end for
4  repeat
5      for $j := 1$ to $\mu$ do
6          $pop_j$.individualImprovement();      // optimization of solutions
7      end for
8      $Pop$.cooperate();                          // cooperation phase
9      for $j := 1$ to $\mu$ do
10         $pop_j$.individualImprovement();      // optimization of solutions
11     end for
12     $Pop$.compete();                            // competition phase
13 until Termination condition is true.
```

space that contained the optimal and high-quality solutions, soon developed in the literature. Since the core concept of MAs is in the use of problem domain knowledge (which includes the use of known heuristics and exact algorithms) researchers quickly introduced fast constructive techniques for creating the initial population. They also used them to "repair" solutions, e.g. to restore feasibility in process involving recombination of solutions like those presented in the first chapter of the book when the Minimum Vertex Cover problem was discussed. For instance, the "cooperative" step for the MA for the National Hockey League timetabling and scheduling of [97] required careful attention so that the population of solutions can "evolve" towards better solutions in configuration space.

Another important force that explains the growing interest in MAs is their intrinsic parallelism. This is not surprising; Moscato and Norman's initial approach was designed to maximally exploit MIMD parallel computing architectures. As a consequence, their contribution in the Parallel Computing and Transputer Application conference of Barcelona in September of 1992 is nowadays the most remembered and cited manuscript from that event [296]. Some researchers in the 1990s were surprised on the speed-ups that they can obtain from their MAs when they migrated their code from a sequential implementation to a parallel one. This is easy to understand if we look at the history [300]. MAs were conceptually designed to be an inherently parallel approach that could deliver the best results that can be obtained from heterogeneous distributed computing systems. This said, MAs were "aiming for the clouds" long before cloud computing had been invented. Academic researchers nevertheless exploited existing parallel computing infrastructures of the 1990s, then with the introduction of software tools for programming on them such as PVM (Parallel Virtual Machine) and the introduction of MPI (Message Passing Interface), they proposed MAs running on federated computing systems across different institutions and even countries [16, 487–490]. Since the original proposal of Moscato and Norman included "point-to-point" communication for the cooperation and competition phase on a fixed topology (and most of the time agents are engaged in individual optimization), parallel MAs like those had almost linear speed-ups in comparison with their sequential versions [150]. These features attracted top researchers and practitioners to the field who benefited from these new hardware and software technologies.

13.3 From Basics Towards "More Evolved" Memetic Algorithms

Although Algorithm 4 lays out a basic framework of an MA, and it is indeed a quite generic template that can accommodate many instantiations of the basic idea, researchers have proposed many variations "evolving" from its central theme of a population search based on cooperation and competition between agents.

The first one to mention is that, once it became more established that cooperation and competition steps interspersed with periods of individual optimization can produce interesting synergies, researchers felt less obliged to maintain the same methods for all agents. Researchers and practitioners started to employ *multiple* methods (either by dynamically randomizing agent's choice from an available "algorithmic portfolio" or by having it fixed, but different than that of other agents in the population). MAs then reaped the benefits of having problem domain knowledge from multiple heuristics, multiple representations of solutions for a given problem, the existence of different constructive heuristics and even different mathematical programming models of a given problem. Agents will then use them to improve individual solutions, in some cases in different sub-populations, e.g. in island-based models of MAs in which subpopulations and migration mechanisms for solutions are established. Different methods have been applied in sequence to solutions, or in a concurrent way and sometimes even the methods itself enter in competition.

A second direction for extension was the introduction of techniques for "diversity preservation", or, perhaps, a better term would have been "diversity management". Indeed, it was quickly recognized back in 1988–1989 that a population of agents, particularly using the same or very similar optimization techniques, may quickly lose the diversity of the population of solutions [296]. In [307] the authors proposed the use of a *tree-structured population*. This restricts the interactions of individual agents. We can think of the whole population as divided into subpopulations of size 4, composed in a ternary tree structure, so some agents would belong to two populations. This new approach also introduced some sort of memory of good solutions found in the past. Some key aspects of this approach are the following:

1. A subpopulation is composed of a *leader* node (at a higher level) and three *supporters* (in the level immediately below).
2. For a 13 node ternary tree, the three intermediate nodes hold agents that belong to two populations, i.e. it is the leader of the three supporters lower in the tree, and a support of its leader, higher in the tree.
3. The number of subpopulations can be increased by adding levels to the ternary tree (e.g. [61]).

4. Agents have more than one solution (for initially testing these new ideas [307] limited it to two, each with different processing applied to them, larger numbers are now accepted).
5. There are processes that control the flow of better solutions towards the top of the tree or how to engage in recombination (i.e. [45]).
6. Diversity loss control mechanisms are necessary and need to be defined (a large number of strategies have been proposed over the years starting from the initial MAs of the 1988–1989).

In [307], agents have different iterative improvement heuristics for the TSP. Solutions can then be optimized using a variety of individual optimization heuristics based on local search, e.g. approximate 2-Opt improvement, One-City Insertion and Two-City Insertion (subsets of the 3-Opt neighbourhood for local search). The cooperative phase involves the use of a polynomial-time algorithm that recombines two solutions and generates a new one (SEX—the Strategic Edge Crossover of [305]). While this brings the necessary focus to search for solutions that may differ a bit from the existing "parent" solutions, it may also contribute to reduce the diversity of them [297]. This delicate balance [194] between "exploration" and "exploitation" is always present in randomized search techniques, so understandably many alternatives were suggested by different MA designers [193, 195, 215, 323, 443]. The restart strategies proposed in [307] were followed by other techniques that employed more complex controlling mechanisms to manage diversity and to adapt according to the needs [58, 140, 152, 177, 405–407, 438, 505]. Still, the work [307] inspired a number of variations on MA basic template with the incorporation of multiple optimization search strategies, multipopulations and later even multiple recombination procedures.

The tree-based approach has proven to be a very robust and versatile strategy and it has been used in many MAs with very good results for a variety of problems, see, for instance, work on total tardiness single machine scheduling [137], the asymmetric TSP [61] and ordering data [312] (see [400] and [159] for the related problem of reconstructing shredded documents).

A third research direction for MAs was that of the improvement of the individual search strategies based on generic guiding principles. For instance, when local search based heuristics had been used the two most obvious extensions to consider were the use of metaheuristics like *Simulated Annealing* and its deterministic update variant now known as *Threshold Accepting* [296, 304, 336], *Tabu Search* [45, 297, 312], *Guided Local Search* [179] or GRASP [146]. The first one was used in the original proposal of MAs for the TSP back in 1988, and Tabu Search was adopted later due to its capacity to introduce diversity to the population while maintaining a good set of quality solutions at all times. In [297], a learning problem is addressed with an MA that employs Tabu Search. A relatively small population of 16 solutions is maintained by 16 agents (each solution being a binary array). A Tabu Search procedure for individual improvement of solutions is followed. When diversity falls below a given threshold, instead of following a simple Tabu Search approach, each agent i makes move i from its list of best moves available. The incorporation of this strategy allows that agents spreading further across the configuration space in different directions, thus contributing to increase the diversity. Once it is restored, agents return to the normal Tabu Search strategy. Another toy-problem (which is an excellent choice in terms of delivering a pedagogical introduction to metaheuristics), but which is also a problem domain of recognizable difficult for Simulated Annealing, has been addressed with MAs in which the agents employ Tabu Search in [45].

Applications of MAs with Simulated Annealing as a key component have been proposed to a number of problem domains including: sensor network deployment [294], GPS Surveying [133], discrete tomography [292], community detection [315], micro-simulation models of vehicular traffic flow [251] and data approximation with local-support curves [250]. Analogously, Tabu Search in MAs has been used in a number of applications including: ordering large datasets [271, 311, 312], for Max-2SAT-problems [55], multiobjective optimization [256], Multi-compartment Vehicle Routing Problems, [121], enrolment-based course timetabling problems [5, 6, 451], the travelling salesman problem [345], generation of low-autocorrelation binary sequences [56], the maximum diversity problem [471], feature point selection for augmented reality [267], Cell Assignment of Hybrid Nanoscale CMOL Circuits [389] and max-mean dispersion [222].

The so-called *adaptive* Memetic Algorithms [30, 48, 66, 104, 105, 110, 111, 172, 208, 219, 293, 330, 340, 347, 373, 410, 432, 443, 460] continue these lines, and more sophisticated methods now exist that allow the dynamic application of different heuristics or tuning of parameters that control the search process. A large number of applications have been exploring the use of adaptation methods with applications in some classical problems like Knapsack [163], credit assignment [431], discovering Resources in P2P networks [325], inverse problems in chemical kinetics [215], Online and Offline Control Design of PMSM Drives [69], DNA fragment assembly [115], personalized e-learning [7, 8], DNA sequence compression [530], multi-robot path planning [371, 372], identification of first-order saddle points [123], vehicle routing problem with time windows [318], quadratic assignment problem [168, 190, 191], remote sensing imagery clustering [257], data clustering [421, 422], maximum diversity [140], large-scale Latin hypercube designs [31], neural network design [413], optimal controller design [22], Bézier Curve Parameterization [186], electricity price forecast [491], data geometry analysis to select training data for support vector machines [320], software architecture optimization [385], software next release [65] and VLSI floorplanning [81].

While some fixed topologies have been used for defining interactions between agents in MAs, researchers are now also recognizing the need of dynamically altering them according to different needs [332, 333]. These include, for instance, the so-called *self-healing* [335], and *self-balancing* [334] in order to deal with the robustness of the MA computation in systems that can have interruptions or loose connection between agents. With an increasing number of applications requiring scale and larger run times this research brings these methodologies closer and closer to be highly useful in large-scale cloud environments.

13.4 A Survey of Applications in Business and Data Analytics from 2014 to May 2018

We have decided to review results on MAs in the past four and a half years as we expect that the previous surveys cited in the introduction have already covered some of the activity in the field before 2013. We then recommend the readers to check the works cited in Sect. 13.1 for further references and to pay attention to some of the references in the publications listed in this book that are older than 2013. Again, although we searched several databases, we give our apologies to those we have inadvertently left out during the survey process.

13.4.1 Problems Involving Location and Routing

While the early work in MAs for the TSP may have generated interest in the research community to extend the domain of application to problems involving routing, we can conjecture that the work related to *assignment* problems contributed to the current interest in applying MAs to location problems. Indeed, the previously cited contributions to timetabling, as well as those on graph colouring and, in particular, the *quadratic assignment problem* on instances that are on physical 2D grids [192] may have contributed to the interest. However, in the last four and a half years the number of applications has skyrocketed and we were gladly surprised with the variety of techniques and different problems that our survey has uncovered. Dynamic problems are also addressed and they have mixed characteristics (e.g. location together with routing requirements), network mapping, pick-up-and-delivery, etc. Table 13.1 shows that the activity seems to be an area strongly dominated by the requirements of optimization from businesses which are increasingly becoming customer-centric, including time-windows, changing demand, etc., but still from a Business-to-Consumer base, to facilitate the company operations in these new markets.

13.4.2 Scheduling, Timetabling and Other Problems in Logistics

As we explained in Sect. 13.2, results in scheduling and timetabling during the 1990s firmly established MAs in this domain, which continues to bring interesting applications and new types of implementations are being proposed in other areas of Logistics. Table 13.2 contains a list of problems that have been addressed with MAs from the areas of transportation, job-shop and flow-shop scheduling, flights planning, network design and identification of critical edges, and the study of business process choreographies are among the most interesting problems in this area studied with MAs since 2014.

Table 13.1 Memetic Algorithms in location and routing

Application area and paper	Method	Problem domain
Automated Guided Vehicle (AGV) Scheduling [78]	Particle Swarm Optimization integrated with Memetic Algorithm	Multiple constraint scheduling of AGV to minimize both the travel and waiting time
Dynamic Routing [500]	Memetic Algorithm	Four-Dimensional Fight Trajectory Planning
Dynamic Routing [362]	Multi-objective MA	Pickup and delivery problem with Time Windows and Demands
Dynamic Routing [244]	Memetic Algorithm	Dynamic Capacitated Arc Routing Problems
Dynamic Routing [492]	Multi-objective MA	Pickup and delivery problem with dynamic customer requests and traffic information
Dynamic Routing [386]	Multi-memory multi-population MA	Dynamic Shortest Path Routing in Mobile Ad hoc Networks
Dynamic Routing [531]	Multi-objective MA	Locality-sensitive hashing for one-to-many-to-one dynamic pickup-and-delivery
Forces deployment optimization [360]	Memetic Algorithm	Air defence command and control system for hybrid deployment of different types of weapons in multiple lines and sections
Location Analysis [93]	Multi-objective Memetic Algorithm	The Multi-objective version of the Obnoxious p-Median problem
Location and Routing [25]	Memetic Algorithm	Multi-objective waste location-routing
Location and Routing [203]	Memetic Algorithm	Capacitated Location-routing Problem with Mixed Backhauls
Location-Routing Problem (LRP) [25]	Memetic Algorithm	Hazardous waste location-routing problem
Location [32]	Two-objective MA	Node localization problem in wireless sensor networks
Location [187]	Memetic Algorithm	Virtual network mapping problem
Location [264]	Memetic Algorithm	Multilevel Uncapacitated Facility Location Problem
Location [67]	Algorithm Portfolio	Dynamic maximal covering location problem
Location [79, 86]	Hybrid MA	Coverage Optimization in Wireless Sensor Networks
Location [68]	Memetic Algorithm	Multi-objective Facility Location Problem
Path Planning in Robotix [331]	Memetic Algorithm	The problem of robot's path planning in dynamic environments
Routing and Scheduling [107]	Memetic Algorithm	Multi-constraint routing and scheduling for home health care service
Routing and Scheduling [319]	Memetic Algorithm	Pickup and Delivery Problem with Time Windows

(continued)

Table 13.1 (continued)

Application area and paper	Method	Problem domain
Routing [428]	Memetic Algorithm	Prize-Collecting Travelling Car Renter Problem
Routing [262]	Memetic Algorithm	Vehicle Routing Problems with dynamic requests
Routing [531]	Memetic Algorithm	one-to-many-to-one dynamic pickup-and-delivery
Routing [274]	Memetic Algorithm	Identification of the least-gasoline consumption path for vehicle navigation
Routing [383]	Island Memetic Algorithm	Vehicle Routing Problems
Routing [448]	Memetic Algorithm	Multi-vehicle Selective Pickup and Delivery Problem
Routing [402]	Memetic Algorithm	Dial-a-ride Problem with Transfers
Routing [29]	Memetic Algorithm	Multi-depot Periodic Vehicle Routing Problem
Routing [504]	Multi-objective Memetic Algorithm	Route planning in dynamic pickup-and-delivery problems
Routing [503]	Multi-objective Memetic Algorithm	Dynamic pickup-and-delivery route problems with time windows
Routing [416]	Memetic Algorithm	Capacitated arc routing problem to compute a path corresponding with the minimum cost
Routing [374]	Memetic Algorithm	Optimization of multi-constrained multicast routing in Mobile ad hoc network (MANET)
Routing [415]	Multi-objective Memetic Algorithm	Capacitated arc routing problem in Multi-objective settings
Routing [4]	Hybrid Grouping Evolution Strategy	Helicopter Routing Problem
Routing [41]	Variable Neighbourhood Descent Hybrid Heuristic	Two-echelon Vehicle Routing Problem with Simultaneous Pickup and Delivery
Routing [441]	Memetic Algorithm	Vehicle Routing with Route Balancing
Routing [506]	Memetic Algorithm	Capacitated Arc Routing Problem
Routing [129, 475]	Memetic Algorithm	Capacitated Arc Routing Problem
Routing [414, 520]	Memetic Algorithm	Periodic capacitated arc routing problem
Routing [473]	Memetic Algorithm	Minmax multiple travelling salesman problem
Routing [368]	Memetic Algorithm	Multi-objective Vehicle Routing Problem with time windows
Routing [27, 73]	Memetic Algorithm	Multi-Trip Vehicle Routing Problem
Routing [521]	Memetic Algorithm	Multi-period Vehicle Routing Problem with Profit

(continued)

Table 13.1 (continued)

Application area and paper	Method	Problem domain
Routing [65]	Memetic Algorithm	Travelling Salesman Problem
Routing [129, 131]	Memetic Algorithm	Capacitated Vehicle Routing Problem
Routing [318]	Memetic Algorithm	Minimizing distance in the vehicle routing problem with time windows
Routing [429]	Tabu Search Hybrid Metaheuristic	Multi-compartment Vehicle Routing Problem
Satellite Control System [354]	Hybrid Memetic Algorithm	Planar Version of the Space Vehicle's Attitude Control and slowing the Satellite Rotation
Scalar Optimization [404]	Memetic Algorithm	Usage within population-based search strategies in differential evolution
Transportation Network [117]	Memetic Algorithm	Explore the robustness of the Chinese air route network, and identify the vital edges
Vehicle Routing [160]	Memetic Algorithm with Simulated Annealing	Capacitated vehicle routing problem with time windows
Vehicle Routing Problem [85]	Ant Colony Optimization Based Memetic Algorithm	Multi-robot patrolling problems formulated as a multiple travelling salesman problem
Vehicle Routing Protocol [380]	Memetic Computing	Optimize the cost and delivery time trade-offs in distribution network
Vehicle Routing [122]	Memetic Algorithm	Scheduling and Routing Problem in a water and electricity distribution company
Vehicle Routing [28]	Memetic Algorithm	Multi-depot and periodic vehicle routing problem
Vehicle Routing [394]	Multi-objective Memetic Algorithm	Central depot vehicle routing problem to find the least cost paths for vehicles to provide services to all customers along with the specified constraints
Vehicle routing [411]	Memetic Algorithm	Multi-agent system to solve three-level freight distribution network
Vehicle Routing [161]	Parallel Memetic Algorithm	Capacitated Vehicle Routing Problem (VRP)
Vehicle Routing [231]	Memetic Algorithm	Multi-objective optimization, a variation of VRP with simultaneously delivery and pickup within a time window
Vehicle Routing [390]	Memetic Algorithm	Heterogeneous fleet school bus routing problem
Vehicle Routing [50]	Parallel Memetic Algorithm	Pickup and Delivery Problem with Time Windows to minimize the fleet size
Vehicle Routing [51]	Bacterial Memetic Algorithm (BMA)	Order Picking Routing Problem in warehousing operation
Vehicular Networks [242]	Memetic Algorithm	Centralized scheduling and controlling of vehicular communication network using software defined network architecture

Table 13.2 Memetic Algorithms in logistics

Application area and paper	Method	Problem domain
Assignment Problem [454]	Memetic Algorithm	Multidimensional assignment problem (MAP)
Assignment Problem [295]	Memetic Algorithm	Feasibly mapping catchment areas while minimizing travelling distance for parcel flow
Business Process Models [494]	Memetic Algorithm	Matching business process models to support the analysis, redesign and implementation in enterprises
Business Processes [128]	Memetic Algorithm	Mining Change Logs in Process Choreographies
Examination Timetabling [229]	Cellular Memetic Algorithm	Examination timetabling problems with a set of hard constraints and a set of soft constraints
Examination Timetabling [227]	Adaptive Coevolutionary Memetic Algorithm	Examination timetabling problems where the search conducted into two spaces: the heuristic space and the solution space
Examination Timetabling [228]	Multi-objective Memetic Algorithm	Uncapacitated Multi-objective examination timetabling problem
Flow-shop Scheduling [118]	Memetic Algorithm	Addressing the distributed blocking flow shop scheduling problem (DBFSP) with sequence dependent setup time (SDST)
Flow-Shop Scheduling [461]	NSGA-II based Memetic Algorithm	Parallel non-identical flowshop scheduling problem
Flow-shop Scheduling [523]	Multi-objective Memetic Algorithm	Permutation flowshop scheduling
Flow-shop scheduling [233]	Multi-objective MA	Multi-objective permutation flowshop problem
Flow-shop scheduling [417]	Memetic Algorithm	No-idle flow shop scheduling problem to minimize the makespan criterion
Flow-shop scheduling [493]	Memetic Algorithm	Makespan minimization in re-entrant permutation flowshop scheduling
Flow-shop scheduling [468]	Memetic Algorithm	MA with machine learning algorithms for the Multi-objective permutation flowshop problem
Flow-shop scheduling [241]	Memetic Algorithm	Permutation flowshop scheduling
Goods distribution [23]	Memetic Algorithm	Goods collection and delivery process in the multi-level distribution network of supply chain management
Integrated logistics network [120]	Memetic Algorithm	Flexible delivery path for Integrated forward/reverse supply chain model
Job-Shop Scheduling [52]	Memetic Algorithm	Flexible Job-Shop Scheduling Problem
Job-Shop Scheduling [459]	Memetic Algorithm	Multi-objective flexible Job Shop Scheduling Problem with complex combination of fuzzy processing time and fuzzy due date

(continued)

Table 13.2 (continued)

Application area and paper	Method	Problem domain
Job-Shop Scheduling [458]	Multi-objective Memetic Algorithm	Flexible Job Shop Scheduling Problem which simultaneously minimizes makespan, total workload and critical workload
Job-Shop Scheduling [363]	Memetic Algorithm	To solve the flexible Job Shop Scheduling Problems
Job-Shop Scheduling [223]	Memetic Algorithm	MA is used to minimize the makespan in Job Scheduling Problem
Job-Shop Scheduling [321]	Memetic Algorithm	Modelling a work-flow for Job Shop Scheduling Problem (JSSP)
Job-Shop Scheduling [220]	Memetic Algorithm	Multi-objective Job Shop Scheduling Problem
Job-Shop Scheduling [349]	Memetic Algorithm	Job Shop Scheduling problem with fuzzy sets modelling for uncertain duration and flexible due date
Job-Shop Scheduling [158]	Multi-objective Memetic Algorithm	Multi-objective flexible Job-Shop Scheduling Problem which also considered human factor and minimized the maximum completion time, maximum workload of machines and total workload of all machines
Job-Shop Scheduling [260]	Memetic Algorithm	Flexible Job Shop Scheduling
Job-Shop Scheduling [350]	Memetic Algorithm	Due-Date Satisfaction in Fuzzy Job Shop Scheduling
Job-Shop Scheduling [509]	Memetic Algorithm	Multiobjective Flexible Job Shop Scheduling
Logistics [39]	Memetic Algorithm	Supply Chains with Flexible Delivery Paths
Logistics [251]	Memetic Algorithm	Calibration of Micro-Simulation Models of Vehicular Traffic Flow
Logistics [419]	Memetic Algorithm	ISO 3832 Standard Luggage Packing Problem
Logistics [204]	Memetic Algorithm	Sustainability and Reliability of Transport in Container Terminals
Logistics [501]	Memetic Algorithm	Network-Wide Flights Planning Optimization
Logistics [399]	Memetic Algorithm	MultiLayer Hierarchical Ring Network Design
Logistics [116]	Memetic Algorithm	Identifying vital edges in Chinese air route networks
Logistics [19]	Memetic Algorithm	Permutation Flowshop Scheduling
Logistics [522]	Memetic Algorithm	Urban Transit Network
Logistics [38]	Memetic Algorithm	Supply Chain Model with Flexible Delivery
Logistics [141]	Hybrid MAs	Optimization of Two-level Reverse Distribution Networks
Logistics [182]	Fuzzy MA	2D Vector Packing with General Costs

(continued)

Table 13.2 (continued)

Application area and paper	Method	Problem domain
Logistics [515]	DE, Heuristics, Local Search and Parameter Adaptation	Marine Maintenance Scheduling
Logistics [246]	Memetic Algorithm	Combinational Disruption Management in Sequence-Dependent Permutation Flowshop
Logistics [480]	Fuzzy MA	Gas and Oil Project Time–Cost–Quality Trade-off
Redundancy Allocation [375]	Memetic Algorithm	Optimal reduction and redundancy methods for improving the reliability system
Scheduling [197]	Memetic Algorithm	Simultaneously optimizes the level of repair and spare parts decisions for fleet systems
Scheduling [369]	Memetic Algorithm	Efficiently schedule emergency supplies from logistics centres during large-scale natural disaster
Scheduling [420]	Memetic Algorithm	Dynamic software project scheduling
Scheduling [108]	Competitive Memetic Algorithm	Distributed permutation flow-shop scheduling (DPFSP) in Multi-objective scenario
Scheduling [119]	Memetic Algorithm	Planning of efficient berth scheduling for Marine container terminals (MCT)
Scheduling [164]	Harmony search-based Memetic Algorithms	Integrated production and transportation scheduling (IPTS) problem transformed into order assignment problem
Scheduling [424]	Memetic Algorithm	Optimize the schedule of the time-triggered network-on-chip used in industrial embedded systems
Scheduling [189]	Memetic Algorithm	Memetic Algorithm in non-linear personnel shift scheduling optimization problem to obtain a medium-term personnel shift roster with a maximized employee substitutability value
Scheduling [355]	Memetic Algorithm	Single batch processing machine scheduling problem with minimization of total earliness and tardiness
Scheduling [430]	Memetic Algorithm	Scheduling of sensor nodes in Underwater ad hoc network
Scheduling [486]	Memetic Differential Evolution	Energy-efficient scheduling in a bi-objective environment
Search-based Software Engineering [377]	Memetic Algorithm	Automatic discovery of software architectures
Supply Chain [40]	Memetic Algorithm	Distribution-allocation problem
Timetable Problem [339]	Memetic Algorithm	Optimize the academic timetable scheduling that satisfies medium and hard constraint
Timetabling [12, 226]	Memetic Algorithm	Examination Timetabling
Timetabling [54]	Variable Neighbourhood Descent	Course Timetabling
Timetabling [434]	Memetic Algorithm	Course Timetabling

13.4.3 Memetic Algorithms for Information Processing, Computing Technologies and Data Analytics

We have also identified that during the last years a number of researchers in MAs have concentrated their efforts in the application of MAs as a tool for optimization problems in the area of data analytics and data science. Instead of engaging in algorithmic/code competition with existing methods on "traditional" combinatorial optimization problems, the focus seems to be aiming at delivering practical solutions for the Information and Computing areas. People in this area seem to be pioneering new routes. The problems addressed are related to software engineering and architecture, and also very interesting new challenges in automation (like automatic test paper generation, reconstruction of shredded documents), as well as those covering issues in automatic document summarization, group anonymization, augmented reality, etc. (shown in Table 13.3).

A number of MAs involve optimization issues in feature selection, machine learning, data clustering, regression and classification, approximation of functions and ill-conditioned parametric inverse problems (shown in Table 13.4).

13.4.4 Services and Disruption Planning and Workforce Analytics

While research in this area is relatively new for MAs, we decided to have a subsection on workforce analytics and in service disruption planning as these are of great economic impact for governments. It is perhaps encouraging to note that there are quality papers in this group (in Table 13.5) that includes scheduling of personnel, projects and services. Together with the known overlap that exists between these applications and those of timetabling and rostering, we think that this surely is an area of future expansion of MA research activity.

13.4.5 Applications in Social Network Analysis and Graph Optimization

The first applications of Memetic Algorithms were in graph optimization problems which already had interesting solutions methods developed for them. This challenging scenarios made the MAs very robust and, understandably, several applications in other problems followed in other graph optimization problems like graph partitioning [276, 278], error correcting graph isomorphism [449], minimum-cost vertex-biconnectivity augmentation [248], inexact graph matching [36], balanced graph partitioning [42], the Arithmetic-Harmonic Cut in graphs [381], graph colouring [254], graph partitioning and k-partitioning [43, 145], and

Table 13.3 Memetic Algorithms in information and computing technologies

Application area and paper	Method	Problem domain
Algorithm Design [132]	User-centric Memetic Algorithm (UcMA)	Distributed computing and Gene ordering problem
Bioinformatics [162]	Memetic Algorithms	Cancer Classification Problem with K-Nearest Neighbour Algorithm
Bioinformatics [149]	Memetic Algorithm	Detecting motifs in biopolymers sequences
Bioinformatics [237]	Multimodal Memetic Algorithm	Tertiary protein structure prediction
Bioinformatics [467]	Memetic Algorithm	Combination chemotherapy problem with dose adjustment
Combinatorial Optimization [526]	Memetic Algorithm	Linear ordering problem with cumulative costs
Combinatorial optimization [379]	Memetic Algorithm	Multidimensional Knapsack Problem
Combinatorial optimization [329]	Memetic Algorithm	Hurdle problems with big valley structures
Crime prediction [382]	Memetic Algorithm	Forecast spatiotemporal patterns of criminal activity
Decision Optimization [370]	Memetic Algorithm	Teaching–Learning based optimization method to effectively handle the constraints and improve the search performance
Dynamic optimization [11]	Memetic Algorithm	In addition to local search it proposed the usage of direct and guided search process
Evolutionary Computing [255]	Multi-objective Memetic Algorithm	Improvement of Shuffled frog leaping algorithm
Game Theory [534]	Memetic Co-evolution	Tackle the Multi-objective Games (MOGs) with postponed preference articulation
Game Theory [484]	Memetic Algorithm	Reconstructing Evolutionary Game (EG) networks with $l_{1/2}$ regularization
Image Processing [94]	Memetic Algorithm	Phase retrieval problem in the field of Coherent Diffraction Imaging
Imaging Technique [95]	Memetic Algorithm	Phase retrieval applied to Coherent Diffractive Imaging (CDI)
Imaging Technique [185]	Memetic Algorithm	The surface reconstruction problem by using rational Bézier surfaces
Information Technology [395]	Multi-objective MA	Quadratic Assignment Problem
Information Technology [327]	Memetic Algorithm	Product Line Architecture Design
Information Technology [273]	Memetic Algorithm	Multi-document Summarization
Information Technology [91, 444]	Memetic Algorithm	Group Anonymization

(continued)

Table 13.3 (continued)

Application area and paper	Method	Problem domain
Information Technology [159]	Memetic Algorithm	Reconstruction of Shredded Documents
Information Technology [328]	Memetic Algorithm	Generation of multiple similarly optimal test papers automatically according to multiple user-specified assessment criteria
Information Technology [387]	Heterogeneous MAs	Big Data 2015 competition benchmark problems (data with and without noise)
Information Technology [512]	Memetic Multiagent System	Automation for Online Human-Like Social Behaviour Learning
Information Technology [87]	Memetic Algorithm	Continuous function optimization
Information Technology [385]	Adaptive MA	Architecture Optimization Problem
Information Technology [525]	Memetic Algorithm	Test Case Generation for Boundary value analysis
Information Technology [88]	Adaptive MA	Multiobjective Optimization for Software Next Release Problem
Information Technology [376]	Memetic Algorithm	Automatic Discovery of Software Architectures
Information Technology [267]	Memetic Algorithm	Automatic Feature Point Selection for Augmented Reality
Information Technology [495, 497]	Memetic Algorithm	Optimization of Ontology Alignments
Information Technology [205]	Memetic Algorithm	Priority-Based Task Scheduling in the Cloud Systems
Information Technology [286]	Memetic Algorithm	Software cost estimation
Information Technology [139]	Memetic Algorithm	Whole Test Suite Generation
Information Technology [335]	Memetic Algorithm	Fault-tolerance in unstable computing environments
Information Technology [65]	Memetic Algorithm	Software next release
Information Technology [469]	Memetic Algorithm	IP block mapping onto mesh-based networks-on-chip
Large scale global optimization [440]	Multi-objective Memetic Algorithm	Solving high-dimensional problems by applying cooperative co-evolution (CC)
Learning-based optimization [528]	Memetic Algorithm	Maximum Diversity Problem
Machine Learning [516]	Memetic Algorithm	Feature selection approach for hyperspectral images classification
Machine Learning [423]	Memetic Algorithm	Optimizing the Artificial Neural Network architecture
Machine Learning [153]	Memetic Algorithm	Wrapper-based Feature selection approach for handwritten words recognition

(continued)

Table 13.3 (continued)

Application area and paper	Method	Problem domain
Machine Learning [357]	Memetic Algorithm	Objective function approximation in radial basis function of Artificial Neural Networks
Machine Learning [147]	Memetic Algorithm	Learning optimization of artificial neural network and other network optimization problems
Machine Learning [290]	Memetic Algorithm	Deep learning approach applied on intrusion-detection problem
Machine Learning [533]	Memetic Algorithm	Mutual information (MI) based two-phase Memetic Algorithm (MA) for learning large-scale fuzzy cognitive maps (FCMs)
Multi-objective Optimization [358]	Memetic Algorithms	Approximation of n-dimensional Objective Functions using Artificial Neural Networks
Multi-objective Optimization [176]	Memetic Algorithm with NSGA-II	Reference point based Multi-objective optimization problem
Multi-objective Optimization [291]	Memetic Algorithm	To enhance the performance of NSGA-II algorithm
Multi-tasking Optimization [82]	Adaptive Memetic Algorithm	The adaptability problems for multitasking optimization
Multi-tasking optimization [83]	Cooperative Coevolutionary Memetic Algorithm	To address the slow convergence in local search and impotence of handling high-dimensional problem in multi-tasking optimization
Natural Language Processing [266]	Memetic Algorithm	Tagging part-of-speech
Numerical Optimization [113]	Memetic Differential Evolution	Analysing the Baldwin effect on a Memetic Algorithm to solve the constrained numerical optimization problems
Numerical Optimization [114]	Memetic Differential Evolution	Analysing the influence of the depth of direct local search methods in constrained numerical optimization problems in order to use as a local search operator (LSO) within a Memetic Algorithm
Numerical Optimization [477]	Memetic Differential Evolution	To achieve the Baldwin effect in existing memetic differential evolution algorithm
Optimization [439]	Surrogate-assisted Memetic Algorithm	Interval Multi-objective optimization problems
Optimization [196]	Memetic Algorithm	Optimization at two interconnected hierarchical levels
Optimization [165]	Memetic Particle Swarm Optimization	Numerical optimization of local search in PSO algorithm in the form of memes
Optimization [217]	Memetic Algorithm	To solve difficult discrete benchmark problems without any domain-specific knowledge

(continued)

Table 13.3 (continued)

Application area and paper	Method	Problem domain
Optimization [508]	Brain storm optimization (BSO) with chaotic local search (CLS)	Eliminating the problem of sticking into stagnation during exploitation phase of BSO
Puzzle [26]	Memetic Algorithm	Harmony Search-based solve Sudoku puzzle solution
Research Survey [102]	Memetic Algorithm	Review the paradigm, structure and applications of Memetic Algorithm
Software Testing [418]	Memetic Algorithm	Automated test data generation for effective software testing in reasonable times
Sorting Permutations [433]	Opposition-based Memetic Algorithm	Sorting the permutations based on biological data
Wireless Communication [517]	Memetic Algorithm	Sparse antenna array design by thinning method
Wireless Sensor Network [457]	Hybrid Memetic Algorithm	Target coverage problem with limited number of sensors

Table 13.4 Memetic Algorithms in data analytics

Application area and paper	Method	Problem domain
Data Analytics [174, 175]	Memetic Algorithm	Identification of overlapping topics based on link communities
Data Analytics [15]	Memetic Algorithm	Clustering and wrapper-based feature selection for electronic documents
Data Analytics [13]	Memetic Differential Evolution	Electronic documents clustering
Data Analytics [14]	Memetic Algorithm	Wrapper-based feature selection using K-means and Spherical K-means (SK-means) clustering for Electronic documents
Data Analytics [201]	Memetic Differential Evolution	Data clustering
Data Analytics [261]	Fireworks Algorithm	Analysis of Electroencephalogram (EEG) signals collected through Brain Computer Interfaces (BCI)
Data Analytics [464]	Memetic Algorithm	Cancer chemotherapy optimization combined with dose adjustment
Data Analytics [511]	Memetic Algorithm	Dynamic categorization of semantic category of fashion language
Data Analytics [37]	Memetic Algorithm	Selecting the genes in microarray data for classification
Data Analytics [202]	Memetic Algorithm	Feature Selection

(continued)

Table 13.4 (continued)

Application area and paper	Method	Problem domain
Data Analytics [249]	Memetic Algorithm	Ill-conditioned parametric inverse problems
Data Analytics [250]	Memetic Algorithm	Data Approximation with Local Support Curves
Data Analytics [518]	Memetic Algorithm	Extreme Learning Machine for Classification
Data Analytics [183]	Memetic Algorithm	Model Selection in Short-term Load Forecasting using Support Vector Regression
Data Analytics [398]	Memetic Algorithm	Feature Selection
Data Analytics [527]	Memetic Algorithm	Identification of critical nodes in sparse graphs
Data Analytics [151]	Memetic Algorithm	Simultaneous Instance and Feature Selection
Data Analytics [60]	Memetic Algorithm	Construction of Transductive Discrete Support Vector Machines
Data Analytics [320]	Adaptive MA	Selection of samples for Training Data for Support Vector Machines
Data Analytics [519]	Memetic Algorithm	MA-based Extreme Learning Machine
Data Analytics [422]	Memetic Algorithm	Data Clustering
Data Analytics [496]	Memetic Algorithm	Instance Coreference Resolution
Data Analytics [184]	Memetic Algorithm	Mid-term interval load forecasting using multi-output support vector regression
Data Analytics [359]	Memetic Algorithm	Fibre-reinforced polymer composites manufacturing optimization
Data Mining [447]	Memetic Algorithm	Fuzzy association rules mining
Text Mining [206]	Memetic Algorithm	Sentiment analysis from online product review
Text Mining [207]	Memetic Algorithm	Quantifying the sentiment words and phrases

as a base for problems with varied constraints such as sketch geolocation [145], the Steiner tree problem with budget, hop, and revenues [143], and the prize-collecting Steiner tree [211].

It is then perhaps very natural that we are observing an increased activity in the area of graph optimization problems and social network analytics. Due to the number of papers in these fields we decided to group them together. Graph colouring still is both a challenge and an opportunity for MAs; several manuscripts address this problem. Other "classical" studies relate to dominating sets, feedback vertex sets, identification of cliques and independent sets, intractable variants of spanning tree problems, variants of graph bi-partitioning, identification of critical nodes. The applications of some of these problems and techniques for "community detection" tend to be in areas where consumers are the central part of a business and need to be well-serviced or, alternatively, their behaviours need to be quantified such as viral

Table 13.5 Memetic Algorithms in service planning and workforce analytics

Application area and paper	Method	Problem domain
Geothermal maturity optimization [481]	Memetic Algorithm	Thermal maturity indexing and history modelling
Multiple Allocation [287]	Memetic Algorithm	Uncapacitated Multiple Allocation p-Hub Centre Problem
Oil Reservoir Steamflood Optimization [221]	Metamodel Assisted Memetic Algorithm	Optimize a field undergoing steamflood under two different oil-price scenarios
Structural Engineering [396]	Memetic Algorithm	Structural health monitoring (SHM) by vibration response measure of engineering structure
Student Affairs Management [474]	Memetic Differential Evolution Algorithm	Comprehensive quality evaluation of college students affairs
Traffic Control [388]	Memetic Algorithm	Optimizing the control of the movement of traffic on urban streets by determining the appropriate signal timing settings
Traffic Engineering [397]	Memetic Algorithm	Maintain the efficiency of road transport and urban road networks to meet the changing needs of commuters
Weapon Portfolio [502]	Self-adaptive Memetic Algorithm	Optimization in Weapon system portfolio problem
Workforce Analytics [188]	Memetic Algorithm	Maximization of Employee Substitutability in Personnel Shift Scheduling
Workforce Analytics [435]	Memetic Algorithm	Staff Scheduling Problems from Airport Security Services
Workforce Analytics [478]	Memetic Algorithm	Break Scheduling
Workforce Analytics [505]	Memetic Algorithm	Project Scheduling

marketing, epidemic threshold selection, web-service composition, flights planning. Another emerging area is the analytics for robust network design for malicious attacks. These approaches are listed in Table 13.6.

13.4.6 Personalization of Services: A Growing Area for MAs

While there are still MA applications in areas centred around the behaviour of individual companies (i.e. like bankruptcy prediction) there has been a clear trend since 2014 in applications that involve the personalization of services in Finance and Tourism. Equity option strategy discovery, portfolio optimization and others are present in our survey, matching the general trend of the machine learning community. In Tourism, the orienteering and the TSP with hotel selection also witness this trend in recommendation systems (Table 13.7). Other researchers are

Table 13.6 Memetic Algorithms in graph and social network analytics

Application area and paper	Method	Problem domain
Communication Network [24]	Memetic Algorithm	Sensor coverage optimization to maximize the coverage and lifetime of Wireless sensor network
Communication Network [235]	Memetic Algorithm	Set k-Cover problem for extending the lifetime of wireless sensor networks (WSNs)
Communication Network [75]	Memetic Algorithm	Transportation network reconstruction problem to minimize the maximum lateness for Post-disaster period
Communication Network [243]	Memetic Algorithm	Real-time data services via infrastructure-to-vehicle communication in the service range of a roadside unit using wireless communication
Communication Network [283]	Learning Automata-based Memetic Algorithm	Handoff and cabling costs management in cellular mobile networks design
Complex Network [148]	Memetic Algorithm	Community detection using modularity and general modularity density score
Graph Optimization [17]	Memetic Algorithm	Free clustered travelling salesman problem (FCTSP)
Graph-based Optimization [64]	Multi-objective Memetic Algorithm	Software next release and travelling salesman problems
Graph-based Optimization [452]	Discrete Bacterial Memetic Evolutionary Algorithm	Travelling Salesman Problem with Time Windows
Graph-based [499]	Memetic Algorithm	Web-service composition
Graph [20]	Memetic Algorithm	Multilevel hypergraph partitioning
Graph [44]	Hybrid Memetic Algorithm	Baldwin effect and Lamarckian evolution in a Memetic Algorithm for Euclidean Steiner tree problem
Graph [450]	Memetic Algorithm	Image processing, data clustering and Graph colouring
Graph [76]	Memetic Algorithm	Minimum conductance problem for network community detection
Graph [284]	Learning Automata-based Memetic Algorithm	To solve the graph isomorphism problem
Graph [253]	Memetic Algorithm	Travelling salesman problem for hotel selection
Graph [49]	Memetic Algorithm	Graph Clustering
Graph [289]	Memetic Algorithm	Combinatorial optimization of k-colouring problem of Graph
Graph [285]	Memetic Algorithm	Vertex colouring problem
Graph [77]	Hybrid Bridge-Based Memetic Algorithm	Minimum conductance graph partitioning for finding bottlenecks in complex networks

(continued)

Table 13.6 (continued)

Application area and paper	Method	Problem domain
Graph [171]	Two-objective Memetic Algorithm	Minimum sum colouring problem in graph
Graph [214]	Discrete Bacterial Memetic Evolutionary Algorithm	Travelling salesman problem
Graph [442]	Memetic Algorithm	Solving Asymmetric Travelling Salesman Problem (ATSP)
Graph [472]	Memetic Algorithm	Minimum independent dominating set problem is tackled with MA using two innovative ideas of forgetting-based vertex weighting strategy and the repairing-based crossover strategy
Graph [532]	Memetic Algorithm	Minimum graph colouring
Graph [288]	Memetic Algorithm	Minimum graph colouring
Graph [282]	Memetic Algorithm	Minimum graph colouring
Graphs [76]	Memetic Algorithm	"Min Conductance": an NP-hard graph partitioning problem
Graphs [110, 476]	(1+1) MA	Maximum Clique
Graphs [170]	Bi-objective MA	Minimum Sum Colouring
Graphs [507]	Multi-parent MA	Linear Ordering Problem
Graphs [74]	Memetic Algorithm	Three degree-dependent Spanning Tree
Graphs [198]	Memetic Algorithm	Minimization of cyclic cutwidth
Graphs [465]	Memetic Algorithm	Structural Balance in Signed Networks
Graphs [156]	Memetic Algorithm	Network Alignment
Graphs [239]	Hybrid MA	Minimum Weight Dominating Set
Graphs [71]	Memetic Algorithm	Weighted Feedback Vertex Set
Mobile ad hoc network (MANET) [408]	Self-generation and co-evolution based Memetic Optimization (SGC-MO)	Maintaining stable, energy efficient route and balancing load among mobile nodes
Network Analytics [245]	Memetic Algorithm	Community detection using community robustness index
Network Analytics [109]	Memetic Algorithm	Computing shortest path in multimodal transport network to satisfy multicriteria
Network Analytics [230]	Memetic Algorithm	Active module identification in biological network
Network Analytics [169]	Memetic Algorithm	Community detection using Connected Cohesion score to analyse complex networks
Network Analytics [467]	Memetic Algorithm	Community detection from bipartite networks using Baber modularity (Q_B) and modularity density (Q_D) measures
Network [510]	Memetic Algorithm	Community detection using single node entropy and partition entropy

(continued)

Table 13.6 (continued)

Application area and paper	Method	Problem domain
Network [259]	Multilevel Memetic Algorithm	Community detection in multiplex networks using extended multiplex modularity
Network [483]	Game-based Memetic Algorithm	Minimum vertex cover problem
Networks [485]	Multiobjective MA	Community Detection
Networks [501]	Memetic Algorithm	Network-wide Flights Planning
Networks [412]	Elitist Pareto MA	Virtual Network Embedding
Networks [524]	Memetic Algorithm	Enhancing the Robustness of Scale-free Networks/Graphs against Malicious Attacks
Operational Research [513]	Memetic Algorithm	Unequal circles packing problem
Social network [167]	Memetic Algorithm	Formalized the existence of critical state in networks, converting it to critical diameter, then minimized the average path length (APL) in social network
Social Network [356]	Memetic Algorithm	Mining Students' learning experience from their Twitter comments
Social Network [437]	Memetic Algorithm	Tourist experience formation model from Facebook comments
Social Networks [238]	Memetic Algorithm	Hierarchical and overlapping community structures detection in social networks
Social Networks [240]	Memetic Algorithm	Minimum positive influence dominating set problem as a constrained integer linear programming applied on Social Network
Social Networks [232]	Memetic Algorithm	Link clustering based Memetic Algorithm for detecting overlapping communities
Social Networks [317]	Memetic Algorithm	Community Detection
Social Networks [76]	Memetic Algorithm	"Min Conductance": an NP-hard graph partitioning problem
Social Networks [316]	Memetic Algorithm	Community Detection
Social Networks [314]	Memetic Algorithm	Community Detection
Social Networks [258]	Memetic Algorithm	Community Detection
Social Networks [315]	Memetic Algorithm	Community Detection
Social Networks [498]	Memetic Algorithm	Epidemic Threshold Selection
Social Networks [498]	Memetic Algorithm	Epidemic Threshold Selection
Social Networks [157]	Memetic Algorithm	Influence Maximization/Viral Marketing
Social Networks [527]	Memetic Algorithm	Identification of critical nodes
Social Networks [527]	Memetic Algorithm	Identification of critical nodes
Social Networks [281]	Memetic Algorithm	Community Detection
Social Networks [514]	Memetic Particle-Swarm Algorithm	Community Detection

(continued)

Table 13.6 (continued)

Application area and paper	Method	Problem domain
Social Networks [234]	Memetic Algorithm	Community Detection
Social Networks [239]	Memetic Algorithm	Minimum Weight Dominating Set Problem
Social Networks [239]	Memetic Algorithm	Minimum Weight Dominating Set Problem
Social Networks [239]	Memetic Algorithm	Minimum Weight Dominating Set Problem
Social Networks [154]	Memetic Algorithm	Attention Maximization
Weighted Graph-based [366]	Memetic Algorithm	Resource allocation
Weighted Graph-based [482]	Memetic Algorithm	Resource allocation

Table 13.7 Other applications of MAs for personalization of services in Finance, Games, Tourism, Music

Application area and paper	Method	Problem domain
Animation [436]	Memetic Algorithm	Real-time articulated kinematic motion
E-commerce [106]	Memetic Algorithm	Enhancement of an AmI-Based Framework for U-commerce in Plan Shopping
Electric cars [378]	Memetic Algorithm	Fleet size and mix vehicle routing problem with electric modular vehicles
Electric Energy Storage (EES) [391]	Memetic Computing	Minimizing the load cost by optimum scheduling in the Unit Commitment Problem (UCP)
Engineering Design [455]	Memetic Algorithm	Optimization in mechanism design
Finance [322]	Memetic Algorithm	Bankruptcy Prediction
Finance [453]	Memetic Algorithm	Equity Option Strategy Discovery and Optimization
Finance [384]	Memetic Algorithm	Cardinality-constrained Portfolio Optimization with Transaction Costs
Finance [33]	Memetic Algorithm	Financial time series prediction
Finance [344]	Memetic Algorithm	Consumption Load Planning
Games [407]	Memetic Algorithm	Solution of Sudoku puzzles
Games [180]	Memetic Multi-Agent System	Creating human-like non-player game characters
Hybrid Electric Vehicles (HEV) [90]	Memetic Algorithm	Enhancing the control strategy in parallel HEV to reduce fuel consumption and emissions without sacrificing the vehicle performance
Literature [317]	Memetic Algorithm	Computational Stylistics

(continued)

Table 13.7 (continued)

Application area and paper	Method	Problem domain
Manufacturing [181]	Memetic Algorithm	Real time alternative routings selection problem in a Flexible Manufacturing System (FMS)
Music [263]	Memetic Algorithm	Personalization in Music Composition
Music [35]	Memetic Algorithm	Music Composition
Music [200]	Memetic Algorithm	Music Composition
Orienteering Problem [252]	Memetic Algorithm	Orienteering Problem with Mandatory Visits and Exclusionary Constraints
Personal Transit [127]	Honeybee Mating Algorithm	Personal Rapid Transit Routing and Guidance
Process Optimization [463]	Memetic Algorithm	Multi-objective optimization of train operation
Product Design [209]	Hybrid Memetic Algorithm	Iterated Function System Fractals in Jewellery Design Applications
Product Design [144]	Multi-objective Memetic Algorithm	Multi-physics, complex inverse problems
Product Design [462]	Memetic Algorithm	Synthesis of heat exchanger networks (HENs) to minimize the total annual cost
Product Design [361]	Bat Algorithm Memetic Approach	Design of power system stabilizers
Product Design [89]	Memetic Algorithm	Control strategy optimization for energy generation, usage and saving in Hybrid electric vehicle (HEV)
Product Design [426]	Memetic Algorithm	Practical broadband optimization of layered thin-film materials of Optical Films
Product Design [427]	Memetic Algorithm	Effective thin-film optimization method for Thermal and Energy applications
Product Engineer [225]	Memetic Algorithm	Online approach to tackle system parameters that reduces the cost function value in electrical measurements
Product Manufacturing [367]	Memetic Algorithm	Flexible manufacturing system for Automated Guided Vehicle System (AGVS)
Project Management [401]	Memetic Algorithm	Stochastic time–cost–quality trade-off problem (STCQTP)
Recommendation Systems [466]	Memetic Algorithm	Topic aware recommendation systems
Tourism [72]	Memetic Algorithm	Travelling Salesperson with Hotel Selection
Tourism [265]	Memetic Algorithm	Orienteering Problem
Tourism [112]	Memetic Algorithm	The Orienteering Problem with Hotel Selection
Tourism [80]	Memetic Algorithm	Skyline Scenic Routes Planning from User-Generated Digital Footprints
Tourism [216]	Iterated Tabu Search	Personalized Multi-period Tour Recommendation

looking at problem domains that could finally deliver recommendations in music (including Music Composition), games and animation, e-commerce and jewellery, as well as computational stylistics (via analyses that use MAs for community detection in graphs).

13.5 About the Other Contributions in This Book

This book presents applications of MAs to a variety of different problems. Two, in fact, are in a different part as they relate to problems in networks.

In “Using Network Alignment to Identify Conserved Consumer Behaviour Modelling Constructs”, the authors present a Memetic Algorithm for the problem of *Network Alignment*. In this problem, which can be seen as a generalization of the well-known *graph isomorphism problem*, we are given two graphs $G_1(V_1, E_1)$ and $G_2(V_2, E_2)$ and, without losing generalization we will assume that $|V_1| \geq |V_2|$, then the task is to find an isomorphism that maps the vertices of V_2 into V_1 such that a certain objective function is maximized. The problem domain presented by Mathieson, de Vries and Moscato is the study of surveys that relate to engagement of consumers to brands in a social media setting but the approach is quite general and can be applied to other problem domains.

In “A Memetic Algorithm to Detect Overlapping Communities (MADOC)”, Gabardo, Berretta and Moscato present a method to uncover community structure in graphs. Most algorithms are based on finding a partition of the set of vertices of a graph. In this case, the authors propose an approach based on the concept of overlapping communities. They employ an approach based on the use of the *line graph*, which can be constructed in polynomial time, followed by a Memetic Algorithm that finds non-overlapping communities (i.e. a partition) of the nodes of the line graph. Back from the transformation process this reveals interesting overlapping communities of nodes in the original graph. The study is based on Amazon’s co-purchasing network and an illustration of the technique using the characters of the *“A Game of Thrones”* series of books (with the interactions obtained from the novel *“A Storm of Swords”*.

Following our survey, in “A Memetic Algorithm for the Team Orienteering Problem”, Trachanatzi et al. present an approach for a problem which has application to create tourist guides for cities. This problem belongs to the area of Vehicle Routing Problems with Profits. This situation happens when time is limited to visit all the attractions and the tourist is forced to make a selection according to some “value” that a priori gives to these points of interest. A feasible path then needs to be constructed, so the earliest version of this problem was related to the Multiple tour Maximum Collection Problem.

In the second chapter of this part of the collection, by Biesinger, Hu and Raidl, the authors present an interesting study of two hypermarket chains that want to open stores in Vienna. In their contribution “A Memetic Algorithm for Competitive Facility Location Problems”, two non-cooperative companies compete for market share. One of them, the leader, aims to choose the locations knowing that the follower company would lower its market share. This is a type of *bi-level* optimization problem and six customer behaviour scenarios and different demand models are considered. A Memetic Algorithm is employed by the leader and the solution evaluated by the follower is obtained via greedy algorithms and mixed integer linear programming techniques. The approach is quite general, allowing practitioners to experiment with different customer behaviours and demand models, thus constituting a new tool for evaluating alternative economic scenarios.

Finally, another application of Memetic Algorithms presented in this part relates to visualization of large datasets. In "Visualizing Products and Consumers: A Gestalt Theory inspired method", the authors present an approach which relies on our inherent perceptual characteristics, together with an optimization technique, to layout in two dimensions objects which are linked by a certain similarity metric. The approach is illustrated with an application to organize the "Marvel Universe" (according to the characters interactions), wines (according to their physicochemical properties) and customers of a telecommunications company that either leave its services (i.e. those that "churn") or that remain on it. The different datasets allow to understand the characteristics of the methods and its potential use for problems involving millions of customers and/or products.

13.6 Future Directions for Memetic Algorithms

There are a number of interesting future directions that can be traced back to previous works where the ideas have been proposed. However, there seems to be more freedom within some well-established journals, and more open editorial policies, so these ideas can be now fully developed by a new generation of researchers. The reasons are many and only in part they relate to improvements in both hardware and software. Most importantly, a cultural change has occurred that has made the field mature. We will try to condense some of them and point also to some of the opportunities ahead.

13.6.1 Evolving Individual Search Techniques

During the first decade of MAs, researchers have to somehow prove his colleagues that the individual search techniques of each agent can "boost" their performance when interspersed with periods of recombination of solutions or by the exchange of information. This said, through the anonymous peer-review of these publications, the community was somewhat "forcing" the authors of the submitted papers to maintain the individual search processes fixed. There were logical arguments put forward, for instance, that the same amount of CPU time spent in performing concurrent searches (by the group of agents but not interacting) was indeed not statistically worse than when they interact.

This restriction had a positive effect on the development of MAs. It allowed researchers to concentrate in developing clever recombination algorithms for creating new solutions from high-developed ones. The progress in the TSP is a clear example, ultimately leading to recombination operators like Merz' Distance Preserving Crossover [275, 279], and the Strategic Arc Crossover for the asymmetric TSP [61]. Both are examples of techniques that work well with the individual local search techniques used by the agents.

Today, with researchers interested in developing new methods that work well on practical instances of real world problems, we have more freedom to propose the "evolution" also of local search using other evolutionary methods (like genetic programming for the capacitated vehicle routing problem [131]). This is likely a very hot area in which we expect to expand in the very near future.

13.6.2 Learning While We Search

Another growing trend is the use of machine learning techniques that can help change the dynamics of the MA [21, 129, 258, 280, 282, 393, 468] or even allow to increase knowledge on solutions methods via the principle of "transfer learning" [130, 512].

An important direction in MA design is the one that investigates the use of "surrogate" and adaptive guiding functions that aim to accelerate the search processes, something which is nowadays considered absolutely necessary when applying MAs for some particular type of problems [155, 168, 364, 445, 529]. Obviously, the increased interest in Multiobjective MAs (quite self-evident at least from the references included in this brief survey) indicates that these two areas will also deliver many publications at the intersection [351–353]. Surrogate-based MAs will be of particular importance for the development of optimization techniques that involve simulation (as well as the introduction of new "meta-analytic" procedures as discussed elsewhere in this book).

13.6.3 Instances as Adversaries (But Also as Design Colleagues)

In both [296] and [307] the authors carefully create some TSP instances with the aim of "confusing" either the individual local search or the recombination of solutions. This said, since its origins, researchers in MAs were interested in making the technique more robust by expanding the knowledge of the weaknesses on specific instances. Later on, this translated in the development of grammar-based generators (of TSP instances) that were named "fractal" (see [306, 337] and the references therein) and these instances were subsequently used to test performance of MAs and other methods [61, 179, 456].

In [306] a suggestion is made that some of these "fractal" TSP instances can be made "harder in practice" (for MAs and other techniques) by displacing some of the cities to slightly different positions (while the same grammar is maintained). In [134] the authors show that removing some cities, while reducing the size, make them harder to be solved in practice by CONCORDE (a high quality exact solver based on some multi-parent recombination algorithm). However, when the number of cities removed is too high, reduced computational costs are also observed (as perhaps expected). This confirms that the structure of the TSP instance has a very important role in the practical observed results in optimally solving them.

In this sense, instances are our "colleagues" for the algorithm design of MAs. We support the view that carefully crafted class of instances, on which an algorithm does not perform well, can give more insights than copious computational experiments on "fashionable" sets of instances which are part of the "folklore" of the field. An interesting question arises: *"Can the generation of these instances be also automated?"*. This question was clearly answered in the affirmative in [10], in which the authors are able to obtain, via an evolutionary algorithm, a new generator of fractal instances that makes CONCORDE exhibit a behaviour that it is four-orders of magnitude slower than in one of the instances of [306] when used as a seed. This shows that highly structured instances can still be a source of "adversarial examples" to test the performance of algorithms and challenge us to provide more robust alternatives.

13.6.4 Development of New Theoretical Results

As we discussed above, finding difficult instances for particular heuristics and algorithms is important from a design perspective. Inspection on the characteristics of these instances can lead to new insights. In addition, they have an important use for theoretical progress. The reason is simple, a large number of theoretical analyses of algorithmic complexity in computer science are based on finding the worst-case instances. In [306], the authors propose a sort of "game", between instances and algorithms, where some constructive algorithms can *fail* in finding the optimal solution for some instances. Co-evolution of instances and algorithms could be key to make them better. This idea is taken forward in [92] in which the authors propose a technique inspired in *comparative analysis* to design recombination operators (and use graph colouring as an illustrative example).

Can we mathematically prove what in computer science is called *tighter results*? What if some advances in some area algorithmic design are not possible because the design problem itself is computationally intractable? We can read in [300] that this issue was raised as early as 1991 that perhaps the problem of designing an *optimal* recombination operator, under some circumstances, would be as difficult as solving an NP-complete problem (so that it is unlikely that an efficient algorithm may exist). Research efforts may then go in the direction of having heuristics, approximation algorithms or other mathematical programming methods which can provide fast but “provably good” recombination of solutions [46]. Moscato even proposed the creation of a computational complexity class to deal with these theoretical issues [299] called *Polynomial Merger Algorithms*.

Theoretical advances in this area are recognized as one of the challenges and duties of the Evolutionary Computation field [99], including of course MAs. Soon it was discovered that multi-parent recombination (under very general circumstances) leads to intractable problems [100]. A few years later Anton Eremeev proved that optimal recombination operators can be designed for problems such as the maximum weight set packing problem, the minimum weight set partition problem. Several NP-hard cases of optimal recombination were observed in the knapsack, the set covering, the p-median problems [124]. Some recombination tasks regarding Boolean linear programming problems become NP-hard when we have with three variables per inequality but efficient algorithms exist when we have *at most* two variables per inequality [124]. Theoretical research, connected with development of implementations that challenge the state-of-the-art of some of these problems [47], is certainly a good way to mature the field and we hope the subject will attract more attention in the future and it looks as a very good trend [125]. Very recently some theoretical results in the study of the complexity of some problems involving *patterns* bring new insights on the computational complexity of multi-parent recombination [224].

13.6.5 Applications, Applications, Applications

If the real estate mantra is *“Location, location, location!”*, perhaps the title of this subsection is the mantra for many MAs practitioners. They are right, undoubtedly you get better and better with practice in the field by working in a myriad of different application domains. In turn, the experience obtained in one domain is quickly translated into better methods for a different domain.

When you are given a problem in a different application area, during the process of understanding the problem (and some of its variations) you may learn about representations of feasible solutions, mechanisms for improvements, data structures that accelerate the search, etc. All this problem domain knowledge does not only help you to develop more sophisticated MAs, but some of the techniques may also be applied in a different domain. Our impression is that, over the years, the best strategy to develop experience in MAs is to adapt yourself for a life-long learning experience in which exposure to different alternative approaches is necessary. After all, this field has been pushing forward hybridization of techniques for three decades! For those that really challenge themselves, once a very successful MA has been developed, a stylistic challenge remains, that is, to make it as simple as possible so that we can truly understand how it is possible to "crack" seemingly intractable problems, thanks to the process of cooperation, competition (and the necessary communication) between a set of interacting agents.

13.7 Conclusion

Over the past three decades, MAs have been continuously delivering high quality solutions for challenging real-world problems in many domains. This survey, limited as it is to only a few years, connects the current activities to MAs roots and it shows that MA is one of the most successful and adaptable approaches available to benefit from problem knowledge. While the No-Free-Lunch theorem [479] guarantees that, ultimately, the performance of an algorithm could not consistently be good for the majority of problems, in some way MAs circumvent this problem. By their core philosophy, and its adaptability and utilitarianism (enforced to the algorithmic designer, but also in the fruit of her/his design), MAs allow a free exploitation of the best characteristics of multiple approaches for the problem at hand. It is the pioneer technique in proposing hybridization of techniques and it just keeps impacting business and data analytics around the world and it seems stronger than ever.

Acknowledgements Pablo Moscato acknowledges previous support from the Australian Research Council Future Fellowship FT120100060 and Australian Research Council Discovery Projects DP120102576 and DP140104183. The authors thank Mohammad Nazmul Haque for his help during the preparation of this manuscript.

References

1. (2013) Proceedings of the IEEE Congress on Evolutionary Computation, CEC 2013, Cancun, Mexico, June 20–23, 2013, IEEE, URL http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=6552460
2. (2016) IEEE Congress on Evolutionary Computation, CEC 2016, Vancouver, BC, Canada, July 24–29, 2016, IEEE, URL http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=7636124
3. (2017) 2017 IEEE Congress on Evolutionary Computation, CEC 2017, Donostia, San Sebastián, Spain, June 5–8, 2017, IEEE, URL http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=7959755
4. Abbasi-Pooya A, Kashan AH (2017) New mathematical models and a hybrid grouping evolution strategy algorithm for optimal helicopter routing and crew pickup and delivery. Computers & Industrial Engineering 112:35–56, URLs https://doi.org/10.1016/j.cie.2017.08.007, https://www.sciencedirect.com/science/article/pii/S0360835217303492
5. Abdullah S, Turabieh H (2012) On the use of multi neighbourhood structures within a tabu-based memetic approach to university timetabling problems. Inf Sci 191:146–168

6. Abdullah S, Turabieh H, McCollum B, McMullan P (2010) A tabu-based memetic approach for examination timetabling problems. In: RSKT, Springer, Lecture Notes in Computer Science, vol 6401, pp 574–581
7. Acampora G, Gaeta M, Ballester EM, Vitiello A (2011) An adaptive multi-agent memetic system for personalizing e-learning experiences. In: FUZZ-IEEE, IEEE, pp 123–130
8. Acampora G, Gaeta M, Loia V (2011) Combining multi-agent paradigm and memetic computing for personalized and adaptive learning experiences. Computational Intelligence 27(2):141–165
9. Agharghor A, Riffi ME, Chebihi F (2016) A memetic hunting search algorithm for the traveling salesman problem. In: Mohajir ME, Chahhou M, Achhab MA, Mohajir BEE (eds) 4th IEEE International Colloquium on Information Science and Technology, CIST 2016, Tangier, Morocco, October 24–26, 2016, IEEE, pp 206–209, URL https://doi.org/10.1109/CIST.2016.7805043
10. Ahammed F, Moscato P (2011) Evolving l-systems as an intelligent design approach to find classes of difficult-to-solve traveling salesman problem instances. In: EvoApplications (1), Springer, Lecture Notes in Computer Science, vol 6624, pp 1–11
11. Akandwanaho SM, Viriri S (2017) A spy search mechanism (SSM) for memetic algorithm (MA) in dynamic environments. In: Phon-Amnuaisuk S, Ang SP, Lee SY (eds) Multi-disciplinary Trends in Artificial Intelligence, Springer International Publishing, Cham, pp 450–461
12. Al-Betar MA, Khader AT, Doush IA (2014) Memetic techniques for examination timetabling. Annals OR 218(1):23–50, URL https://doi.org/10.1007/s10479-013-1500-7
13. Al-Jadir I, Wong KW, Fung CC, Xie H (2017) Differential evolution memetic document clustering using chaotic logistic local search. In: Liu D, Xie S, Li Y, Zhao D, El-Alfy ESM (eds) Neural Information Processing, Springer International Publishing, Cham, pp 213–221
14. Al-Jadir I, Wong KW, Fung CC, Xie H (2017) Text dimensionality reduction for document clustering using hybrid memetic feature selection. In: Phon-Amnuaisuk S, Ang SP, Lee SY (eds) Multi-disciplinary Trends in Artificial Intelligence, Springer International Publishing, Cham, pp 281–289
15. Al-Jadir I, Wong KW, Fung CC, Xie H (2017) Text document clustering using memetic feature selection. In: Proceedings of the 9th International Conference on Machine Learning and Computing, ACM, New York, NY, USA, ICMLC 2017, pp 415–420, URL http://doi.acm.org/10.1145/3055635.3056603
16. Alba E, Almeida F, Blesa MJ, Cabeza J, Cotta C, Díaz M, Dorta I, Gabarró J, León C, Luna J, Moreno LM, Pablos C, Petit J, Rojas A, Xhafa F (2002) MALLBA: A library of skeletons for combinatorial optimisation (research note). In: Monien B, Feldmann R (eds) Euro-Par 2002, Parallel Processing, 8th International Euro-Par Conference Paderborn, Germany, August 27–30, 2002, Proceedings, Springer, Lecture Notes in Computer Science, vol 2400, pp 927–932, URL https://doi.org/10.1007/3-540-45706-2_132
17. Alsheddy A (2017) Solving the free clustered TSP using a memetic algorithm. International Journal of Advanced Computer Science and Applications 8(8), URL http://dx.doi.org/10.14569/IJACSA.2017.080852
18. Amaya JE, Porras CC, Leiva AJF (2015) Memetic and hybrid evolutionary algorithms. In: Kacprzyk J, Pedrycz W (eds) Springer Handbook of Computational Intelligence, Springer, pp 1047–1060, URL https://doi.org/10.1007/978-3-662-43505-2_52
19. Amirghasemi M, Zamani R (2017) An effective evolutionary hybrid for solving the permutation flowshop scheduling problem. Evolutionary Computation 25(1):87–111, URL https://doi.org/10.1162/EVCO_a_00162
20. Andre R, Schlag S, Schulz C (2017) Memetic Multilevel Hypergraph Partitioning. ArXiv e-prints 1710.01968
21. António CC (2014) A memetic algorithm based on multiple learning procedures for global optimal design of composite structures. Memetic Computing 6(2):113–131
22. Arab A, Alfi A (2015) An adaptive gradient descent-based local search in memetic algorithm applied to optimal controller design. Inf Sci 299:117–142

23. Arango-Serna MD, Serna-Uran CA, Zapata-Cortes JA (2018) Multi-agent System Modeling for the Coordination of Processes of Distribution of Goods Using a Memetic Algorithm, Springer International Publishing, pp 71–89. URL https://doi.org/10.1007/978-3-319-56871-3_4
24. Arivudainambi D, Balaji S (2017) Improved memetic algorithm for energy efficient sensor scheduling with adjustable sensing range. Wireless Personal Communications 95(2):1737–1758, URL https://doi.org/10.1007/s11277-016-3883-7
25. Asgari N, Rajabi M, Jamshidi M, Khatami M, Farahani RZ (2017) A memetic algorithm for a multi-objective obnoxious waste location-routing problem: a case study. Annals of Operations Research 250(2):279–308, URL https://doi.org/10.1007/s10479-016-2248-7
26. Assad A, Deep K (2017) Harmony search based memetic algorithms for solving Sudoku. International Journal of System Assurance Engineering and Management URL https://doi.org/10.1007/s13198-017-0620-x
27. Ayadi R, Benadada Y (2013) Memetic algorithm for a multi-objective vehicle routing problem with multiple trips. IJCSA 10(2):72–91
28. Azad AS, Islam M, Chakraborty S (2017) A heuristic initialized stochastic memetic algorithm for MDPVRP with interdependent depot operations. IEEE Transactions on Cybernetics 47(12):4302–4315, https://doi.org/10.1109/TCYB.2016.2607220
29. Azad AS, Islam MM, Chakraborty S (2017) A Heuristic Initialized Stochastic Memetic Algorithm for MDPVRP With Interdependent Depot Operations. IEEE Transactions on Cybernetics 47(12):4302–4315, URL https://doi.org/10.1109/TCYB.2016.2607220
30. Azevedo CRB, Gordon VS (2009) Adaptive terrain-based memetic algorithms. In: GECCO, ACM, pp 747–754
31. Aziz M, Tayarani-N M (2014) An adaptive memetic particle swarm optimization algorithm for finding large-scale Latin hypercube designs. Eng Appl of AI 36:222–237
32. Aziz M, Tayarani-N M, Meybodi MR (2016) A two-objective memetic approach for the node localization problem in wireless sensor networks. Genetic Programming and Evolvable Machines 17(4):321–358
33. Baboli M, Abadeh MS (2015) Financial time series prediction by a hybrid memetic computation-based support vector regression (MA-SVR) method. International Journal of Operational Research 23(3):321–339, URLs https://doi.org/10.1504/IJOR.2015.069625, http://www.inderscienceonline.com/doi/abs/10.1504/IJOR.2015.069625, http://www.inderscienceonline.com/doi/pdf/10.1504/IJOR.2015.069625
34. Bader-El-Den MB, Poli R, Fatima S (2009) Evolving timetabling heuristics using a grammar-based genetic programming hyper-heuristic framework. Memetic Computing 1(3):205–219, URL https://doi.org/10.1007/s12293-009-0022-y
35. Ballester EM, Cadenas JM, Ong Y, Acampora G (2016) Memetic music composition. IEEE Trans Evolutionary Computation 20(1):1–15
36. Bärecke T, Detyniecki M (2007) Memetic algorithms for inexact graph matching. In: IEEE Congress on Evolutionary Computation, IEEE, pp 4238–4245
37. Begum S, Chakraborty S, Banerjee A, Das S, Sarkar R, Chakraborty D (2018) Gene selection for diagnosis of cancer in microarray data using memetic algorithm. In: Bhateja V, Coello Coello CA, Satapathy SC, Pattnaik PK (eds) Intelligent Engineering Informatics, Springer Singapore, pp 441–449
38. Behmanesh E, Pannek J (2016) A memetic algorithm with extended random path encoding for a closed-loop supply chain model with flexible delivery. Logistics Research 9(1):22:1–22:12
39. Behmanesh E, Pannek J (2016) Modeling and random path-based direct encoding for a closed loop supply chain model with flexible delivery paths. IFAC-PapersOnLine 49(2):78–83, URLs https://doi.org/10.1016/j.ifacol.2016.03.014, https://www.sciencedirect.com/science/article/pii/S2405896316300143, 7th {IFAC} Conference on Management and Control of Production and Logistics {MCPL} 2016Bremen, Germany, 22–24 February 2016
40. Behmanesh E, Pannek J (2018) Assessment of Memetic and Genetic Algorithm for a Flexible Integrated Logistics Network. International Journal of Industrial and Manufacturing Engineering 12(3):1403, URL http://www.waset.org/abstracts/81478

41. Belgin O, Karaoglan I, Altiparmak F (2018) Two-echelon vehicle routing problem with simultaneous pickup and delivery: Mathematical model and heuristic approach. Computers & Industrial Engineering 115:1–16, URLs https://doi.org/10.1016/j.cie.2017.10.032, https://www.sciencedirect.com/science/article/pii/S0360835217305193
42. Benlic U, Hao J (2010) An effective multilevel memetic algorithm for balanced graph partitioning. In: ICTAI (1), IEEE Computer Society, pp 121–128
43. Benlic U, Hao J (2011) A multilevel memetic approach for improving graph k-partitions. IEEE Trans Evolutionary Computation 15(5):624–642
44. Bereta M (2018) Baldwin effect and Lamarckian evolution in a memetic algorithm for Euclidean Steiner tree problem. Memetic Computing URL https://link.springer.com/content/pdf/10.1007/s12293-018-0256-7.pdf
45. Berretta R, Moscato P (1999) The number partitioning problem: An open challenge for evolutionary computation ? In: Corne D, Dorigo M, Glover F (eds) New Ideas in Optimization, McGraw-Hill, pp 261–278
46. Berretta R, Rodrigues LF (2004) A memetic algorithm for a multistage capacitated lot-sizing problem. International Journal of Production Economics 87(1):67–81, URLs http://dx.doi.org/10.1016/S0925-5273(03)00093-8, http://www.sciencedirect.com/science/article/pii/S0925527303000938
47. Berretta R, Cotta C, Moscato P (2004) Enhancing the Performance of Memetic Algorithms by Using a Matching-Based Recombination Algorithm, Springer US, Boston, MA, pp 65–90. URL https://doi.org/10.1007/978-1-4757-4137-7_4
48. Bhowmik P, Rakshit P, Konar A, Kim E, Nagar AK (2012) DE-TDQL: an adaptive memetic algorithm. In: IEEE Congress on Evolutionary Computation, IEEE, pp 1–8
49. Biedermann S, Henzinger M, Schulz C, Schuster B (2018) Memetic Graph Clustering. ArXiv e-prints 1802.07034
50. Blocho M, Nalepa J (2018) Complexity analysis of the parallel memetic algorithm for the pickup and delivery problem with time windows. In: Gruca A, Czachórski T, Harezlak K, Kozielski S, Piotrowska A (eds) Man-Machine Interactions 5, Springer International Publishing, Cham, pp 471–480
51. Bódis T, Botzheim J (2018) Bacterial Memetic Algorithms for Order Picking Routing Problem with Loading Constraints. Expert Systems with Applications 105:196–220, URL https://www.sciencedirect.com/science/article/pii/S0957417418301891
52. Böning C, Prinzhorn H, Hund EC, Stonis M (2017) A memetic algorithm for an energy-costs-aware flexible job-shop scheduling problem. Int J Soc Behav Educ Econ Bus Ind Eng 11(5):1223–1236
53. Bontoux B, Artigues C, Feillet D (2010) A memetic algorithm with a large neighborhood crossover operator for the generalized traveling salesman problem. Computers & OR 37(11):1844–1852, URL https://doi.org/10.1016/j.cor.2009.05.004
54. Borchani R, Elloumi A, Masmoudi M (2017) Variable neighborhood descent search based algorithms for course timetabling problem: Application to a Tunisian university. Electronic Notes in Discrete Mathematics 58:119–126, URLs https://doi.org/10.1016/j.endm.2017.03.016, https://www.sciencedirect.com/science/article/pii/S1571065317300525, 4th International Conference on Variable Neighborhood Search
55. Borschbach M, Exeler A (2008) A tabu history driven crossover operator design for memetic algorithm applied to max-2sat-problems. In: GECCO, ACM, pp 605–606
56. Boskovic B, Brglez F, Brest J (2014) Low-autocorrelation binary sequences: on the performance of memetic-tabu and self-avoiding walk solvers. CoRR abs/1406.5301
57. Bosman PAN (ed) (2017) Genetic and Evolutionary Computation Conference, Berlin, Germany, July 15–19, 2017, Companion Material Proceedings, ACM, URL http://doi.acm.org/10.1145/3067695
58. Botzheim J (2012) A novel diversity induction method for bacterial memetic algorithm by hibernation of individuals. In: 2012 Sixth International Conference on Genetic and Evolutionary Computing, ICGEC 2012, Kitakyushu, Japan, August 25–28, 2012, IEEE, pp 328–331, URL https://doi.org/10.1109/ICGEC.2012.25

59. Botzheim J, Földesi P, Kóczy LT (2009) Solution for fuzzy road transport traveling salesman problem using eugenic bacterial memetic algorithm. In: Carvalho JP, Dubois D, Kaymak U, da Costa Sousa JM (eds) Proceedings of the Joint 2009 International Fuzzy Systems Association World Congress and 2009 European Society of Fuzzy Logic and Technology Conference, Lisbon, Portugal, July 20–24, 2009, pp 1667–1672, URL http://www.eusflat.org/proceedings/IFSA-EUSFLAT_2009/pdf/tema_1667.pdf
60. Brandner H, Lessmann S, Voß S (2013) A memetic approach to construct transductive discrete support vector machines. European Journal of Operational Research 230(3):581–595
61. Buriol LS, França M, Moscato P (2004) A new memetic algorithm for the asymmetric traveling salesman problem. J Heuristics 10(5):483–506, URL https://doi.org/10.1023/B:HEUR.0000045321.59202.52
62. Burke EK, Ross P (eds) (1996) Practice and Theory of Automated Timetabling, First International Conference, Edinburgh, U.K., August 29 - September 1, 1995, Selected Papers, Lecture Notes in Computer Science, vol 1153, Springer, URL https://doi.org/10.1007/3-540-61794-9
63. Burke EK, Newall JP, Weare RF (1995) A memetic algorithm for university exam timetabling. In: [62], pp 241–250, URL https://doi.org/10.1007/3-540-61794-9_63
64. Cai X, Cheng X, Fan Z, Goodman E, Wang L (2017) An adaptive memetic framework for multi-objective combinatorial optimization problems: studies on software next release and travelling salesman problems. Soft Computing 21(9):2215–2236, URL https://doi.org/10.1007/s00500-015-1921-0
65. Cai X, Cheng X, Fan Z, Goodman ED, Wang L (2017) An adaptive memetic framework for multi-objective combinatorial optimization problems: studies on software next release and travelling salesman problems. Soft Comput 21(9):2215–2236
66. Calderín JF, Masegosa AD, Rosete-Suárez A, Pelta DA (2013) Adaptation schemes and dynamic optimization problems: A basic study on the adaptive hill climbing memetic algorithm. In: NICSO, Springer, Studies in Computational Intelligence, vol 512, pp 85–97
67. Calderín JF, Masegosa AD, Pelta DA (2017) An algorithm portfolio for the dynamic maximal covering location problem. Memetic Computing 9(2):141–151
68. Capitanescu F, Marvuglia A, Benetto E, Ahmadi A, Tiruta-Barna L (2017) Linear programming-based directed local search for expensive multi-objective optimization problems: Application to drinking water production plants. European Journal of Operational Research 262(1):322–334, URL https://doi.org/10.1016/j.ejor.2017.03.057
69. Caponio A, Cascella GL, Neri F, Salvatore N, Sumner M (2007) A fast adaptive memetic algorithm for online and offline control design of PMSM drives. IEEE Trans Systems, Man, and Cybernetics, Part B 37(1):28–41
70. Carlos C, E GJ, Luke M, Pablo M (2016) Memetic Algorithms: A Contemporary Introduction, American Cancer Society, pp 1–15. URLs https://doi.org/10.1002/047134608X.W8330, https://onlinelibrary.wiley.com/doi/abs/10.1002/047134608X.W8330, https://onlinelibrary.wiley.com/doi/pdf/10.1002/047134608X.W8330
71. Carrabs F, Cerrone C, Cerulli R (2014) A memetic algorithm for the weighted feedback vertex set problem. Networks 64(4):339–356, URL http://dx.doi.org/10.1002/net.21577
72. Castro M, Sörensen K, Vansteenwegen P, Goos P (2013) A memetic algorithm for the travelling salesperson problem with hotel selection. Computers & OR 40(7):1716–1728
73. Cattaruzza D, Absi N, Feillet D, Vidal T (2014) A memetic algorithm for the multi trip vehicle routing problem. European Journal of Operational Research 236(3):833–848
74. Cerrone C, Cerulli R, Raiconi A (2014) Relations, models and a memetic approach for three degree-dependent spanning tree problems. European Journal of Operational Research 232(3):442–453
75. Chagas JBC, Santos AG, Souza MJF (2018) A memetic algorithm for the network construction problem with due dates. In: Abraham A, Muhuri PK, Muda AK, Gandhi N (eds) Intelligent Systems Design and Applications, Springer International Publishing, Cham, pp 209–220

76. Chalupa D (2017) A Memetic Algorithm for the Minimum Conductance Graph Partitioning Problem. ArXiv e-prints 1704.02854
77. Chalupa D, Hawick KA, Walker JA (2018) Hybrid bridge-based memetic algorithms for finding bottlenecks in complex networks. Big Data Research URL https://www.sciencedirect.com/science/article/pii/S2214579617303738
78. Chawla V, Chanda AK, Angra S (2018) Scheduling of multi load AGVs in FMS by modified memetic particle swarm optimization algorithm. Journal of Project Management 3(1):39–54
79. Chen C, Mukhopadhyay SC, Chuang C, Lin T, Liao M, Wang Y, Jiang J (2015) A hybrid memetic framework for coverage optimization in wireless sensor networks. IEEE Trans Cybernetics 45(10):2309–2322
80. Chen C, Chen X, Wang L, Ma X, Wang Z, Liu K, Guo B, Zhou Z (2017) MA-SSR: A memetic algorithm for skyline scenic routes planning leveraging heterogeneous user-generated digital footprints. IEEE Trans Vehicular Technology 66(7):5723–5736, URL https://doi.org/10.1109/TVT.2016.2639550
81. Chen J, Liu Y, Zhu Z, Zhu W (2017) An adaptive hybrid memetic algorithm for thermal-aware non-slicing VLSI floorplanning. Integration 58:245–252
82. Chen Q, Ma X, Sun Y, Zhu Z (2017) Adaptive memetic algorithm based evolutionary multi-tasking single-objective optimization. In: Shi Y, Tan KC, Zhang M, Tang K, Li X, Zhang Q, Tan Y, Middendorf M, Jin Y (eds) Simulated Evolution and Learning, Springer International Publishing, Cham, pp 462–472
83. Chen Q, Ma X, Zhu Z, Sun Y (2017) Evolutionary multi-tasking single-objective optimization based on cooperative co-evolutionary memetic algorithm. In: 2017 13th International Conference on Computational Intelligence and Security (CIS), pp 197–201, https://doi.org/10.1109/CIS.2017.00050
84. Chen X, Ong Y, Lim M, Tan KC (2011) A multi-facet survey on memetic computation. IEEE Trans Evolutionary Computation 15(5):591–607, URL https://doi.org/10.1109/TEVC.2011.2132725
85. Chen X, Zhang P, Du G, Li F (2018) Ant colony optimization based memetic algorithm to solve bi-objective multiple traveling salesmen problem for multi-robot systems. IEEE Access URL https://ieeexplore.ieee.org/abstract/document/8341754/
86. Chen Z, Li S, Yue W (2014) Memetic algorithm-based multi-objective coverage optimization for wireless sensor networks. Sensors 14(11):20,500–20,518
87. Chen Z, Wang R, Sánchez RV, de Oliveira JV, Li C (2018) An adaptive genomic difference based genetic algorithm and its application to memetic continuous optimization. Intell Data Anal 22(2):363–382
88. Cheng X, Huang Y, Cai X, Wei O (2014) An adaptive memetic algorithm based on multiobjective optimization for software next release problem. In: GECCO (Companion), ACM, pp 185–186
89. Cheng YH, Lai CM (2017) Control strategy optimization for parallel hybrid electric vehicles using a memetic algorithm. Energies 10(3), URLs https://doi.org/10.3390/en10030305, http://www.mdpi.com/1996-1073/10/3/305
90. Cheng YH, Lai CM, Teh J (2017) Memetic algorithm for fuel economy and low emissions parallel hybrid electric vehicles. In: 2017 IEEE 8th International Conference on Awareness Science and Technology (iCAST), pp 219–222, https://doi.org/10.1109/ICAwST.2017.8256449
91. Chertov O, Tavrov D (2014) Two-phase memetic modifying transformation for solving the task of providing group anonymity. In: WCSC, Springer, Studies in Fuzziness and Soft Computing, vol 342, pp 239–253
92. Coll P, Durán G, Moscato P (1999) On worst-case and comparative analysis as design principles for efficient recombination operators: A graph coloring case study. In: Corne D, Dorigo M, Glover F (eds) New Ideas in Optimization, McGraw-Hill, pp 279–294
93. Colmenar J, Martí R, Duarte A (2018) Multi-objective memetic optimization for the bi-objective obnoxious p-median problem. Knowledge-Based Systems 144:88 – 101, URLs https://doi.org/10.1016/j.knosys.2017.12.028, http://www.sciencedirect.com/science/article/pii/S0950705117306068

94. Colombo A, Caro LD, Galli DE (2017) Memetic phase retrieval and HPC for the imaging of matter at atomic resolution. In: Parallel Computing is Everywhere, Proceedings of the International Conference on Parallel Computing, ParCo 2017, 12–15 September 2017, Bologna, Italy, pp 67–76, URL https://doi.org/10.3233/978-1-61499-843-3-67
95. Colombo A, Galli DE, De Caro L, Scattarella F, Carlino E (2017) Facing the phase problem in coherent diffractive imaging via memetic algorithms. Scientific Reports 7:42,236, URL https://www.nature.com/articles/srep42236
96. Conant-Pablos SE, Magaña-Lozano DJ, Terashima-Marín H (2009) Pipelining memetic algorithms, constraint satisfaction, and local search for course timetabling. In: Aguirre AH, Borja RM, García CAR (eds) MICAI 2009: Advances in Artificial Intelligence, 8th Mexican International Conference on Artificial Intelligence, Guanajuato, México, November 9–13, 2009. Proceedings, Springer, Lecture Notes in Computer Science, vol 5845, pp 408–419, URL https://doi.org/10.1007/978-3-642-05258-3_36
97. Costa D (1995) An evolutionary tabu search algorithm and the NHL scheduling problem. INFOR: Information Systems and Operational Research 33(3):161–178, URL http://dx.doi.org/10.1080/03155986.1995.11732279
98. Cotta C, Moscato P (2003) A memetic-aided approach to hierarchical clustering from distance matrices: application to gene expression clustering and phylogeny. Biosystems 72(1):75–97, URLs http://dx.doi.org/10.1016/S0303-2647(03)00136-9, http://www.sciencedirect.com/science/article/pii/S0303264703001369, computational Intelligence in Bioinformatics
99. Cotta C, Moscato P (2004) Evolutionary computation: Challenges and duties. In: Menon A (ed) Frontiers of Evolutionary Computation, Springer US, Boston, MA, pp 53–72, URL https://doi.org/10.1007/1-4020-7782-3_3
100. Cotta C, Moscato P (2005) The parameterized complexity of multiparent recombination. In: Proceedings of the Sixth Metaheuristics International Conference (MIC 2005), Vienna, Austria, August 22–26, 2005, pp 237–242
101. Cotta C, Moscato P (2007) Memetic algorithms. In: Gonzalez TF (ed) Handbook of Approximation Algorithms and Metaheuristics., Chapman and Hall/CRC, URL https://doi.org/10.1201/9781420010749.ch27
102. Cotta C, Mathieson L, Moscato P (2017) Memetic Algorithms, Springer International Publishing, Cham, pp 1–32. URL https://doi.org/10.1007/978-3-319-07153-4_29-1
103. Créput J, Koukam A (2009) A memetic neural network for the Euclidean traveling salesman problem. Neurocomputing 72(4–6):1250–1264, URL https://doi.org/10.1016/j.neucom.2008.01.023
104. Dang HV, Kinsner W (2016) Adaptive multiobjective memetic optimization. IJCINI 10(4):21–58
105. Dang HV, Kinsner W (2016) An information theoretic criterion for adaptive multiobjective memetic optimization. In: ICCI*CC, IEEE Computer Society, pp 15–28
106. D'Aniello G, Orciuoli F, Parente M, Vitiello A (2014) Enhancing an AmI-based framework for u-commerce by applying memetic algorithms to plan shopping. In: INCoS, IEEE, pp 169–175
107. Decerle J, Grunder O, El Hassani AH, Barakat O (2018) A memetic algorithm for a home health care routing and scheduling problem. Operations Research for Health Care 16:59–71, URLs https://doi.org/10.1016/j.orhc.2018.01.004, http://www.sciencedirect.com/science/article/pii/S2211692317300735
108. Deng J, Wang L (2017) A competitive memetic algorithm for multi-objective distributed permutation flow shop scheduling problem. Swarm and Evolutionary Computation 32:121–131, https://doi.org/10.1016/j.swevo.2016.06.002, URL http://www.sciencedirect.com/science/article/pii/S2210650216300281
109. Dib O, Caminada A, Manier MA, Moalic L (2017) A memetic algorithm for computing multicriteria shortest paths in stochastic multimodal networks. In: Proceedings of the Genetic and Evolutionary Computation Conference Companion, ACM, New York, NY, USA, GECCO '17, pp 103–104, URLs https://doi.org/10.1145/3067695.3076064, http://doi.acm.org/10.1145/3067695.3076064

110. Dinneen MJ, Wei K (2013) A (1+1) adaptive memetic algorithm for the maximum clique problem. In: IEEE Congress on Evolutionary Computation, IEEE, pp 1626–1634
111. Dinneen MJ, Wei K (2013) On the analysis of a (1+1) adaptive memetic algorithm. In: Memetic Computing, IEEE, pp 24–31
112. Divsalar A, Vansteenwegen P, Sörensen K, Cattrysse D (2014) A memetic algorithm for the orienteering problem with hotel selection. European Journal of Operational Research 237(1):29–49
113. Dominguez-Isidro S, Mezura-Montes E (2017) The Baldwin effect on a memetic differential evolution for constrained numerical optimization problems. In: Proceedings of the Genetic and Evolutionary Computation Conference Companion, ACM, New York, NY, USA, GECCO '17, pp 203–204, URL http://doi.acm.org/10.1145/3067695.3076096
114. Domínguez-Isidro S, Mezura-Montes E (2017) Study of direct local search operators influence in memetic differential evolution for constrained numerical optimization problems. In: 2017 International Conference on Electronics, Communications and Computers, CONIELECOMP 2017, Cholula, Mexico, February 22–24, 2017, pp 1–8, URL https://doi.org/10.1109/CONIELECOMP.2017.7891831
115. Dorronsoro B, Alba E, Luque G, Bouvry P (2008) A self-adaptive cellular memetic algorithm for the DNA fragment assembly problem. In: IEEE Congress on Evolutionary Computation, IEEE, pp 2651–2658
116. Du W, Liang B, Yan G, Lordan O, Cao X (2016) Identifying vital edges in Chinese air route network via memetic algorithm. CoRR abs/1608.00142
117. Du W, Liang B, Yan G, Lordan O, Cao X (2017) Identifying vital edges in Chinese air route network via memetic algorithm. Chinese Journal of Aeronautics 30(1):330–336, URLs https://doi.org/10.1016/j.cja.2016.12.001, http://www.sciencedirect.com/science/article/pii/S1000936116302163
118. Duan W, Li Z, Yang Y, Liu B, Wang K (2017) EDA based probabilistic Memetic Algorithm for distributed blocking permutation flowshop scheduling with sequence dependent setup time. In: 2017 IEEE Congress on Evolutionary Computation (CEC), pp 992–999, https://doi.org/10.1109/CEC.2017.7969416
119. Dulebenets MA (2017) A novel memetic algorithm with a deterministic parameter control for efficient berth scheduling at marine container terminals. Maritime Business Review 2(4):302–330, URL https://www.emeraldinsight.com/doi/abs/10.1108/MABR-04-2017-0012
120. E Behmanesh JP, Behmanesh E, Pannek J (2018) Ranking Parameters of a Memetic Algorithm for a Flexible Integrated Logistics Network. In: Freitag M, Kotzab H, Pannek J (eds) Dynamics in Logistics, Springer International Publishing, pp 76–85
121. El-Fallahi A, Prins C, Calvo RW (2008) A memetic algorithm and a tabu search for the multi-compartment vehicle routing problem. Computers & OR 35(5):1725–1741
122. El-Yaakoubi A, El-Fallahi A, Cherkaoui M, Reghioui M (2017) Tabu Search and Memetic Algorithms for a Real Scheduling and Routing Problem. Logistics Research 10(7), URL https://www.bvl.de/files/1951/1988/1852/2239/10.23773-2017_7.pdf
123. Ellabaan MMH, Chen X, Nguyen QH (2012) Multi-modal valley-adaptive memetic algorithm for efficient discovery of first-order saddle points. In: SEAL, Springer, Lecture Notes in Computer Science, vol 7673, pp 83–92
124. Eremeev AV (2008) On complexity of optimal recombination for binary representations of solutions. Evolutionary Computation 16(1):127–147, URL http://dx.doi.org/10.1162/evco.2008.16.1.127
125. Eremeev AV, Kovalenko JV (2014) Optimal recombination in genetic algorithms for combinatorial optimization problems: Part ii. Yugoslav Journal of Operations Research 24:165–186
126. Farkas M, Földesi P, Botzheim J, Kóczy LT (2009) Approximation of a modified traveling salesman problem using bacterial memetic algorithms. In: Rudas IJ, Fodor JC, Kacprzyk J (eds) Towards Intelligent Engineering and Information Technology, Studies in Computational Intelligence, vol 243, Springer, pp 607–625, URL https://doi.org/10.1007/978-3-642-03737-5_44

127. Fatnassi E, Chebbi O, Chaouachi J (2016) Discrete honeybee mating optimization algorithm for the routing of battery-operated automated guidance electric vehicles in personal rapid transit systems. Swarm and Evolutionary Computation 26:35–49, URLs https://doi.org/10.1016/j.swevo.2015.08.001, https://www.sciencedirect.com/science/article/pii/S2210650215000619
128. Fdhila W, Rinderle-Ma S, Indiono C (2014) Memetic algorithms for mining change logs in process choreographies. In: ICSOC, Springer, Lecture Notes in Computer Science, vol 8831, pp 47–62
129. Feng L, Ong Y, Lim M, Tsang IW (2015) Memetic search with interdomain learning: A realization between CVRP and CARP. IEEE Trans Evolutionary Computation 19(5):644–658
130. Feng L, Ong Y, Tan A, Tsang IW (2015) Memes as building blocks: a case study on evolutionary optimization + transfer learning for routing problems. Memetic Computing 7(3):159–180, URL https://doi.org/10.1007/s12293-015-0166-x
131. Feng L, Ong Y, Chen C, Chen X (2016) Conceptual modeling of evolvable local searches in memetic algorithms using linear genetic programming: a case study on capacitated vehicle routing problem. Soft Comput 20(9):3745–3769, URL https://doi.org/10.1007/s00500-015-1971-3
132. Fernández-Leiva AJ, Gutiérrez-Fuentes Á (2018) On distributed user-centric memetic algorithms. Soft Computing URL https://link.springer.com/article/10.1007/s00500-018-3049-5
133. Fidanova S, Alba E, Molina G (2008) Memetic simulated annealing for the GPS surveying problem. In: NAA, Springer, Lecture Notes in Computer Science, vol 5434, pp 281–288
134. Fischer T, Stützle T, Hoos H, Merz P (2005) An analysis of the hardness of TSP instances for two high performance algorithms. In: Proceedings of the Sixth Metaheuristics International Conference (MiC 2005), Vienna, Austria, August 22–26, 2005, pp 361–367
135. Földesi P, Botzheim J (2010) Modeling of loss aversion in solving fuzzy road transport traveling salesman problem using eugenic bacterial memetic algorithm. Memetic Computing 2(4):259–271, URL https://doi.org/10.1007/s12293-010-0037-4
136. Fonseca GHG, Santos HG (2013) Memetic algorithms for the high school timetabling problem. In: [1], pp 666–672, URL https://doi.org/10.1109/CEC.2013.6557632
137. França PM, Mendes A, Moscato P (2001) A memetic algorithm for the total tardiness single machine scheduling problem. European Journal of Operational Research 132(1):224–242, URL https://doi.org/10.1016/S0377-2217(00)00140-5
138. França PM, Gupta JN, Mendes AS, Moscato P, Veltink KJ (2005) Evolutionary algorithms for scheduling a flowshop manufacturing cell with sequence dependent family setups. Computers and Industrial Engineering 48(3):491–506, URLs http://dx.doi.org/10.1016/j.cie.2003.11.004, http://www.sciencedirect.com/science/article/pii/S0360835204001986, groupTechnology/Cellular Manufacturing
139. Fraser G, Arcuri A, McMinn P (2015) A memetic algorithm for whole test suite generation. Journal of Systems and Software 103:311–327
140. de Freitas ARR, Guimarães FG, Silva RCP, Souza MJF (2014) Memetic self-adaptive evolution strategies applied to the maximum diversity problem. Optimization Letters 8(2):705–714
141. Freitas ARR, Silva VMR, Campelo F, Guimarães FG (2014) Optimizing two-level reverse distribution networks with hybrid memetic algorithms. Optimization Letters 8(2):753–762
142. Friedman JH, Tukey JW (1974) A projection pursuit algorithm for exploratory data analysis. IEEE Trans Computers 23(9):881–890, URL https://doi.org/10.1109/T-C.1974.224051
143. Fu Z, Hao J (2015) Dynamic programming driven memetic search for the Steiner tree problem with revenues, budget, and hop constraints. INFORMS Journal on Computing 27(2):221–237
144. Gajda-Zagórska E, Schaefer R, Smolka M, Pardo D, Álvarez-Aramberri J (2017) A multi-objective memetic inverse solver reinforced by local optimization methods. Journal of Computational Science 18:85–94
145. Galinier P, Boujbel Z, Fernandes MC (2011) An efficient memetic algorithm for the graph partitioning problem. Annals OR 191(1):1–22
146. Gallardo JE, Cotta C (2015) A grasp-based memetic algorithm with path relinking for the far from most string problem. Eng Appl of AI 41:183–194, URL https://doi.org/10.1016/j.engappai.2015.01.020

147. Ganjefar S, Tofighi M (2018) Optimization of quantum-inspired neural network using memetic algorithm for function approximation and chaotic time series prediction. Neurocomputing 291:175–186, URL https://www.sciencedirect.com/science/article/pii/S0925231218302418
148. Gao D, Cai Z (2017) Community mining algorithm of complex network based on memetic algorithm. In: ISPACS, IEEE, pp 450–455
149. Garbelini JMC, Kashiwabara AY, Sanches DS (2018) Sequence motif finder using memetic algorithm. BMC bioinformatics 19(1)
150. Garcia V, França PM, Mendes A, Moscato P (2006) A parallel memetic algorithm applied to the total tardiness machine scheduling problem. In: 20th International Parallel and Distributed Processing Symposium (IPDPS 2006), Proceedings, 25–29 April 2006, Rhodes Island, Greece, IEEE, URL https://doi.org/10.1109/IPDPS.2006.1639514
151. García-Pedrajas N, de Haro-García A, Pérez-Rodríguez J (2014) A scalable memetic algorithm for simultaneous instance and feature selection. Evolutionary Computation 22(1):1–45
152. Garza-Fabre M, Kandathil SM, Handl J, Knowles JD, Lovell SC (2016) Generating, maintaining, and exploiting diversity in a memetic algorithm for protein structure prediction. Evolutionary Computation 24(4):577–607, URL https://doi.org/10.1162/EVCO_a_00176
153. Ghosh M, Malakar S, Bhowmik S, Sarkar R, Nasipuri M (2017) Memetic algorithm based feature selection for handwritten city name recognition. In: Mandal JK, Dutta P, Mukhopadhyay S (eds) Computational Intelligence, Communications, and Business Analytics, Springer Singapore, Singapore, pp 599–613
154. Godinho P, Moutinho L, Pagani M (2017) A memetic algorithm for maximizing earned attention in social media. Journal of Modelling in Management 12(3):364–385, URL https://doi.org/10.1108/JM2-10-2015-0078
155. Goh CK, Lim D, Ma L, Ong Y, Dutta PS (2011) A surrogate-assisted memetic co-evolutionary algorithm for expensive constrained optimization problems. In: IEEE Congress on Evolutionary Computation, IEEE, pp 744–749
156. Gong M, Peng Z, Ma L, Huang J (2016) Global biological network alignment by using efficient memetic algorithm. IEEE/ACM Trans Comput Biology Bioinform 13(6):1117–1129
157. Gong M, Song C, Duan C, Ma L, Shen B (2016) An efficient memetic algorithm for influence maximization in social networks. IEEE Comp Int Mag 11(3):22–33
158. Gong X, Deng Q, Gong G, Liu W, Ren Q (2017) A memetic algorithm for multi-objective flexible job-shop problem with worker flexibility. International Journal of Production Research URL http://www.tandfonline.com/doi/abs/10.1080/00207543.2017.1388933
159. Gong YJ, Ge YF, Li JJ, Zhang J, Ip W (2016) A splicing-driven memetic algorithm for reconstructing cross-cut shredded text documents. Applied Soft Computing 45:163–172, URLs https://doi.org/10.1016/j.asoc.2016.03.024, http://www.sciencedirect.com/science/article/pii/S1568494616301338
160. González OM, Segura C, Peña SIV, León C (2017) A memetic algorithm for the Capacitated Vehicle Routing Problem with Time Windows. In: 2017 IEEE Congress on Evolutionary Computation (CEC), IEEE, pp 2582–2589, https://doi.org/10.1109/CEC.2017.7969619
161. González OM, Segura C, Peña SIV (2018) A parallel memetic algorithm to solve the capacitated vehicle routing problem with time windows. International Journal of Combinatorial Optimization Problems and Informatics 9(1):35–45
162. Goweda A, Elmogy M, Barakat S (2017) Blending Memetic Search Strategy with K-Nearest Neighbor Algorithm for Cancer Classification Problem. Journal of Next Generation Information Technology 8(3)
163. Guo X, Wu Z, Yang G (2005) A hybrid adaptive multi-objective memetic algorithm for 0/1 knapsack problem. In: Australian Conference on Artificial Intelligence, Springer, Lecture Notes in Computer Science, vol 3809, pp 176–185
164. Guo Z, Shi L, Chen L, Liang Y (2017) A harmony search-based memetic optimization model for integrated production and transportation scheduling in MTO manufacturing. Omega 66:327–343, URLs https://doi.org/10.1016/j.omega.2015.10.012, http://www.sciencedirect.com/science/article/pii/S0305048315002169, new Research Frontiers in Sustainability

165. Gupta S, Sahni H (2018) Memes evolution technique in memetic particle swarm optimization. International Journal of Applied Engineering Research 13(8):6477–6486
166. Gutin G, Karapetyan D (2010) A memetic algorithm for the generalized traveling salesman problem. Natural Computing 9(1):47–60, URL https://doi.org/10.1007/s11047-009-9111-6
167. H Du J Wang XHWD (2017) A memetic algorithm to optimize critical diameter. Swarm and Evolutionary Computation URL https://www.sciencedirect.com/science/article/pii/S2210650217301414
168. Handoko SD, Nguyen DT, Yuan Z, Lau HC (2014) Reinforcement learning for adaptive operator selection in memetic search applied to quadratic assignment problem. In: GECCO (Companion), ACM, pp 193–194
169. Haque MN, Mathieson L, Moscato P (2017) A memetic algorithm for community detection by maximising the connected cohesion. In: 2017 IEEE Symposium Series on Computational Intelligence (SSCI), pp 1–8, https://doi.org/10.1109/SSCI.2017.8285404
170. Harrabi O, Fatnassi E, Bouziri H, Chaouachi J (2017) A bi-objective memetic algorithm proposal for solving the minimum sum coloring problem. In: GECCO (Companion), ACM, pp 27–28
171. Harrabi O, Fatnassi E, Bouziri H, Chaouachi J (2017) A bi-objective memetic algorithm proposal for solving the minimum sum coloring problem. In: Proceedings of the Genetic and Evolutionary Computation Conference Companion, ACM, New York, NY, USA, GECCO '17, pp 27–28 URL http://doi.acm.org/10.1145/3067695.3082035
172. Hart WE (2003) Locally-adaptive and memetic evolutionary pattern search algorithms. Evolutionary Computation 11(1):29–51
173. Hart WE, Ktaliorasnogor N, Smith JE (2004) Editorial introduction special issue on memetic algorithms. Evolutionary Computation 12(3):v–vi, URL https://doi.org/10.1162/1063656041775009
174. Havemann F, Gläser J, Heinz M (2015) A link-based memetic algorithm for reconstructing overlapping topics from networks of papers and their cited sources. In: ISSI, Bogaziçi University Printhouse
175. Havemann F, Gläser J, Heinz M (2017) Memetic search for overlapping topics based on a local evaluation of link communities. Scientometrics 111(2):1089–1118, URL https://doi.org/10.1007/s11192-017-2302-5
176. Hernandez Mejia JA, Schütze O, Cuate O, Lara A, Deb K (2017) RDS-NSGA-II: a memetic algorithm for reference point based multi-objective optimization. Engineering Optimization 49(5):828–845
177. Herrera-Poyatos A, Herrera F (2017) Genetic and memetic algorithm with diversity equilibrium based on greedy diversification. CoRR abs/1702.03594, URL http://arxiv.org/abs/1702.03594
178. Hofmann R (1992) Parallel evolutionary trajectories. Research Report SS92-11, Edinburgh Parallel Computing Centre
179. Holstein D, Moscato P (1999) Memetic algorithms using guided local search: A case study. In: Corne D, Dorigo M, Glover F (eds) New Ideas in Optimization, McGraw-Hill, pp 235–244
180. Hou Y, Feng L, Ong Y (2016) Creating human-like non-player game characters using a memetic multi-agent system. In: IJCNN, IEEE, pp 177–184
181. Houari H, Houbad Y, Souier M, Sari Z, Nassima K (2017) Adaptation of Memetic Algorithm with Population Management for the Improvement of the Performances of Flexible Manufacturing Systems. The Open Automation and Control Systems Journal 10, URL https://benthamopen.com/FULLTEXT/TOAUTOCJ-9-2
182. Hu Q, Wei L, Lim A (2017) The two-dimensional vector packing problem with general costs. Omega URLs http://dx.doi.org/10.1016/j.omega.2017.01.006, http://www.sciencedirect.com/science/article/pii/S030504831730052X
183. Hu Z, Bao Y, Xiong T (2014) Comprehensive learning particle swarm optimization based memetic algorithm for model selection in short-term load forecasting using support vector regression. Appl Soft Comput 25:15–25

184. Hu Z, Bao Y, Chiong R, Xiong T (2015) Mid-term interval load forecasting using multi-output support vector regression with a memetic algorithm for feature selection. Energy 84:419–431, URLs http://dx.doi.org/10.1016/j.energy.2015.03.054, http://www.sciencedirect.com/science/article/pii/S0360544215003485
185. Iglesias A, Gálvez A (2017) Memetic electromagnetism algorithm for surface reconstruction with rational bivariate Bernstein basis functions. Natural Computing 16(4):511–525, URL https://doi.org/10.1007/s11047-016-9562-5
186. Iglesias A, Gálvez A, Collantes M (2016) Four adaptive memetic bat algorithm schemes for bézier curve parameterization. Trans Computational Science 28:127–145
187. Inführ J, Raidl GR (2016) A memetic algorithm for the virtual network mapping problem. J Heuristics 22(4):475–505
188. Ingels J, Maenhout B (2017) A memetic algorithm to maximise the employee substitutability in personnel shift scheduling. In: EvoCOP, Lecture Notes in Computer Science, vol 10197, pp 44–59
189. Ingels J, Maenhout B (2017) A memetic algorithm to maximise the employee substitutability in personnel shift scheduling. In: Hu B, López-Ibáñez M (eds) Evolutionary Computation in Combinatorial Optimization, Springer International Publishing, Cham, pp 44–59
190. Inostroza-Ponta M, Berretta R, Mendes A, Moscato P (2006) An automatic graph layout procedure to visualize correlated data. In: Bramer M (ed) Artificial Intelligence in Theory and Practice: IFIP 19th World Computer Congress, TC 12: IFIP AI 2006 Stream, August 21–24, 2006, Santiago, Chile, Springer US, Boston, MA, pp 179–188
191. Inostroza-Ponta M, Mendes A, Berretta R, Moscato P (2007) An integrated QAP-based approach to visualize patterns of gene expression similarity. In: Proceedings of the 3rd Australian Conference on Progress in Artificial Life, ACAL '07, Springer-Verlag, Berlin, Heidelberg, Lecture Notes in Computer Science, vol. 4828, pp 156–167
192. Inostroza-Ponta M, Berretta R, Moscato P (2011) QAPgrid: A two level QAP-based approach for large-scale data analysis and visualization. PLOS ONE 6(1):1–18, URL https://doi.org/10.1371/journal.pone.0014468
193. Ishibuchi H, Shibata Y (2004) Mating scheme for controlling the diversity-convergence balance for multiobjective optimization. In: Deb K, Poli R, Banzhaf W, Beyer H, Burke EK, Darwen PJ, Dasgupta D, Floreano D, Foster JA, Harman M, Holland O, Lanzi PL, Spector L, Tettamanzi A, Thierens D, Tyrrell AM (eds) Genetic and Evolutionary Computation - GECCO 2004, Genetic and Evolutionary Computation Conference, Seattle, WA, USA, June 26–30, 2004, Proceedings, Part I, Springer, Lecture Notes in Computer Science, vol 3102, pp 1259–1271, URL https://doi.org/10.1007/978-3-540-24854-5_121
194. Ishibuchi H, Yoshida T, Murata T (2003) Balance between genetic search and local search in memetic algorithms for multiobjective permutation flowshop scheduling. IEEE Trans Evolutionary Computation 7(2):204–223, URL https://doi.org/10.1109/TEVC.2003.810752
195. Ishibuchi H, Tanigaki Y, Akedo N, Nojima Y (2013) How to strike a balance between local search and global search in multiobjective memetic algorithms for multiobjective 0/1 knapsack problems. In: [1], pp 1643–1650, URL https://doi.org/10.1109/CEC.2013.6557758
196. Islam MM, Singh HK, Ray T, Sinha A (2017) An Enhanced Memetic Algorithm for Single-Objective Bilevel Optimization Problems. Evolutionary Computation 25(4):607–642
197. Jain A, Rao GK, Rawat M, Lad BK (2017) Memetic algorithm to optimize level of repair and spare part decisions for fleet system. In: 2017 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM), pp 1935–1939, https://doi.org/10.1109/IEEM.2017.8290229
198. Jain P, Srivastava K, Saran G (2016) Minimizing cyclic cutwidth of graphs using a memetic algorithm. J Heuristics 22(6):815–848
199. Jat SN, Yang S (2008) A memetic algorithm for the university course timetabling problem. In: 20th IEEE International Conference on Tools with Artificial Intelligence (ICTAI 2008), November 3–5, 2008, Dayton, Ohio, USA, Volume 1, IEEE Computer Society, pp 427–433, URL https://doi.org/10.1109/ICTAI.2008.126

200. Jeong J, Kim Y, Ahn CW (2017) A multi-objective evolutionary approach to automatic melody generation. Expert Systems with Applications 90:50–61, URLs https://doi.org/10.1016/j.eswa.2017.08.014, https://www.sciencedirect.com/science/article/pii/S0957417417305511
201. Jiang L, Xie D (2018) An efficient differential memetic algorithm for clustering problem. IAENG International Journal of Computer Science 45(1), URL http://www.iaeng.org/IJCS/issues_v45/issue_1/IJCS_45_1_17.pdf
202. Jimenez F, Sanhueza C, Berretta R, Moscato P (2017) A multi-objective approach for the (α, β)-k-feature set problem using memetic algorithms. In: [57], pp 207–208, URL http://doi.acm.org/10.1145/3067695.3076106
203. Karaoglan I, Altiparmak F (2015) A memetic algorithm for the capacitated location-routing problem with mixed backhauls. Computers & OR 55:200–216
204. Kavakeb S, Nguyen TT, Benmerikhi M, Yang Z, Jenkinson I (2014) An improved memetic algorithm to enhance the sustainability and reliability of transport in container terminals. In: CISDA, IEEE, pp 1–8
205. Keshanchi B, Navimipour NJ (2016) Priority-based task scheduling in the cloud systems using a memetic algorithm. Journal of Circuits, Systems, and Computers 25(10):1–33
206. Keshavarz H, Abadeh MS (2017) MVP: Memetic Voter Patterns for aspect extraction in sentiment analysis. In: 2017 2nd Conference on Swarm Intelligence and Evolutionary Computation (CSIEC), pp 89–94, https://doi.org/10.1109/CSIEC.2017.7940154
207. Keshavarz H, Abadeh MS (2017) SOMA: Semantic Orientation inference using Memetic Algorithm. In: 2017 2nd Conference on Swarm Intelligence and Evolutionary Computation (CSIEC), pp 83–88, https://doi.org/10.1109/CSIEC.2017.7940153
208. Kheng CW, Chong SY, Lim M (2012) Centroid-based memetic algorithm - adaptive Lamarckian and Baldwinian learning. Int J Systems Science 43(7):1193–1216
209. Kielarova SW (2016) Development of hybrid memetic algorithm and general regression neural network for generating iterated function system fractals in jewelry design applications. In: ICSI (1), Springer, Lecture Notes in Computer Science, vol 9712, pp 280–289
210. King C, Pendlebury D (2013) Research fronts 2013: 100 top-ranked specialities in the sciences and social sciences. Thomson Reuters, New York p 32
211. Klau GW, Ljubic I, Moser A, Mutzel P, Neuner P, Pferschy U, Raidl GR, Weiskircher R (2004) Combining a memetic algorithm with integer programming to solve the prize-collecting Steiner tree problem. In: GECCO (1), Springer, Lecture Notes in Computer Science, vol 3102, pp 1304–1315
212. Kóczy LT, Földesi P, Tuu-Szabo B (2016) A discrete bacterial memetic evolutionary algorithm for the traveling salesman problem. In: [2], pp 3261–3267, URL https://doi.org/10.1109/CEC.2016.7744202
213. Kóczy LT, Földesi P, Tuu-Szabo B (2017) An effective discrete bacterial memetic evolutionary algorithm for the traveling salesman problem. Int J Intell Syst 32(8):862–876, URL https://doi.org/10.1002/int.21893
214. Kóczy LT, Földesi P, Tüű-Szabó B (2017) Enhanced discrete bacterial memetic evolutionary algorithm-An efficacious metaheuristic for the traveling salesman optimization. Information Sciences URL https://www.sciencedirect.com/science/article/pii/S0020025517309866
215. Kononova AV, Hughes KJ, Pourkashanian M, Ingham DB (2007) Fitness diversity based adaptive memetic algorithm for solving inverse problems of chemical kinetics. In: IEEE Congress on Evolutionary Computation, IEEE, pp 2366–2373
216. Kotiloglu S, Lappas T, Pelechrinis K, Repoussis P (2017) Personalized multi-period tour recommendations. Tourism Management 62:76–88, URLs https://doi.org/10.1016/j.tourman.2017.03.005, https://www.sciencedirect.com/science/article/pii/S0261517717300572
217. Kowol M, Pietak K, Kisiel-Dorohinicki M, Byrski A (2017) Agent-based evolutionary and memetic black-box discrete optimization. Procedia Computer Science 108:907–916, URLs https://doi.org/10.1016/j.procs.2017.05.173, http://www.sciencedirect.com/science/article/pii/S1877050917307573, international Conference on Computational Science, ICCS 2017, 12–14 June 2017, Zurich, Switzerland

218. Krasnogor N (2012) Memetic algorithms. In: Rozenberg G, Bäck T, Kok JN (eds) Handbook of Natural Computing, Springer, pp 905–935, URL https://doi.org/10.1007/978-3-540-92910-9_29
219. Krasnogor N, Smith J (2000) A memetic algorithm with self-adaptive local search: TSP as a case study. In: GECCO, Morgan Kaufmann, pp 987–994
220. Kurdi M (2017) An improved island model memetic algorithm with a new cooperation phase for multi-objective job shop scheduling problem. Computers & Industrial Engineering 111:183–201, URLs https://doi.org/10.1016/j.cie.2017.07.021, http://www.sciencedirect.com/science/article/pii/S0360835217303236
221. Kyriacou S, Sarma P, Hunt I, et al (2017) Constrained, multi-objective, steamflood injection redistribution optimization, using a cloud-distributed, metamodel-assisted, memetic optimization algorithm. In: SPE Reservoir Characterisation and Simulation Conference and Exhibition, Society of Petroleum Engineers, URL https://www.onepetro.org/conference-paper/SPE-186010-MS
222. Lai X, Hao J (2016) A tabu search based memetic algorithm for the max-mean dispersion problem. Computers & OR 72:118–127
223. Lamos-Díaz H, Aguilar-Imitola K, Pérez-Díaz YT, Galván-Núñez S (2017) A memetic algorithm for minimizing the makespan in the Job Shop Scheduling problem. Revista Facultad de Ingeniería 26(44):111
224. Lancia G, Mathieson L, Moscato P (2017) Separating sets of strings by finding matching patterns is almost always hard. Theor Comput Sci 665:73–86, URL https://doi.org/10.1016/j.tcs.2016.12.018
225. Lee DC (2017) Filtering, smoothing, memetic algorithms, and feasible direction methods for estimating system state and unknown parameters of electromechanical motion devices. US Patent App. 15/360,995
226. Lei Y, Gong M, Jiao L, Zuo Y (2015) A memetic algorithm based on hyper-heuristics for examination timetabling problems. Int J Intelligent Computing and Cybernetics 8(2):139–151, URL https://doi.org/10.1108/IJICC-02-2015-0005
227. Lei Y, Gong M, Jiao L, Shi J, Zhou Y (2017) An adaptive coevolutionary memetic algorithm for examination timetabling problems. IJBIC 10(4):248–257
228. Lei Y, Shi J, Yan Z (2018) A memetic algorithm based on MOEA/D for the examination timetabling problem. Soft Computing 22(5):1511–1523
229. Leite N, Fernandes CM, Melício F, Rosa AC (2018) A cellular memetic algorithm for the examination timetabling problem. Computers & Operations Research 94:118–138
230. Li D, Pan Z, Hu G, Zhu Z, He S (2017) Active module identification in intracellular networks using a memetic algorithm with a new binary decoding scheme. BMC Genomics 18(2):209, URL https://doi.org/10.1186/s12864-017-3495-y
231. Li H, Wang L, Hei X, Li W, Jiang Q (2018) A decomposition-based chemical reaction optimization for multi-objective vehicle routing problem for simultaneous delivery and pickup with time windows. Memetic Computing 10(1):103–120
232. Li M, Liu J (2018) A link clustering based memetic algorithm for overlapping community detection. Physica A: Statistical Mechanics and its Applications 503:410–423, URL http://www.sciencedirect.com/science/article/pii/S037843711830253X
233. Li X, Ma S (2016) Multi-objective memetic search algorithm for multi-objective permutation flow shop scheduling problem. IEEE Access 4:2154–2165
234. Li Y, Liu J, Liu C (2014) A comparative analysis of evolutionary and memetic algorithms for community detection from signed social networks. Soft Comput 18(2):329–348
235. Liao CC, Ting CK (2018) A novel integer-coded memetic algorithm for the set k-cover problem in wireless sensor networks. IEEE Transactions on Cybernetics pp 1–14, https://doi.org/10.1109/TCYB.2017.2731598
236. Lim M, Gustafson SM, Krasnogor N, Ong Y (2009) Editorial to the first issue. Memetic Computing 1(1):1–2, URL https://doi.org/10.1007/s12293-009-0007-x
237. de Lima Corrêa L, Borguesan B, Krause MJ, Dorn M (2018) Three-dimensional protein structure prediction based on memetic algorithms. Computers & Operations Research 91:160–177, URL https://www.sciencedirect.com/science/article/pii/S0305054817302897

238. Lin CC, Deng DJ, Wu JC, Lu LY (2018) Detecting hierarchical and overlapping community structures in social networks using a one-stage memetic algorithm. In: Li B, Shu L, Zeng D (eds) Communications and Networking, Springer International Publishing, Cham, pp 182–188
239. Lin G, Zhu W, Ali MM (2016) An effective hybrid memetic algorithm for the minimum weight dominating set problem. IEEE Trans Evolutionary Computation 20(6):892–907
240. Lin G, Guan J, Feng H (2018) An ILP based memetic algorithm for finding minimum positive influence dominating sets in social networks. Physica A: Statistical Mechanics and its Applications 500:199–209, URLs https://doi.org/10.1016/j.physa.2018.02.119, http://www.sciencedirect.com/science/article/pii/S0378437118302218
241. Liu F, Wang S, Hong Y, Yue X (2017) On the Robust and Stable Flowshop Scheduling Under Stochastic and Dynamic Disruptions. IEEE Transactions on Engineering Management 64(4):539–553, URL https://doi.org/10.1109/TEM.2017.2712611
242. Liu K, Feng L, Dai P, Lee VCS, Son SH, Cao J (2017) Coding-Assisted Broadcast Scheduling via Memetic Computing in SDN-Based Vehicular Networks. IEEE Transactions on Intelligent Transportation Systems pp 1–12, https://doi.org/10.1109/TITS.2017.2748381
243. Liu K, Feng L, Dai P, Wu W, Lee VCS, Son SH (2017) A memetic algorithm for cache-aided data broadcast with network coding in vehicular networks. In: GLOBECOM 2017–2017 IEEE Global Communications Conference, pp 1–6, https://doi.org/10.1109/GLOCOM.2017.8254677
244. Liu M, Singh HK, Ray T (2014) A memetic algorithm with a new split scheme for solving dynamic capacitated arc routing problems. In: IEEE Congress on Evolutionary Computation, IEEE, pp 595–602
245. Liu W, Gong M, Wang S, Ma L (2018) A two-level learning strategy based memetic algorithm for enhancing community robustness of networks. Information Sciences 422:290–304
246. Liu Xp, Liu F, Wang Jj (2016) An enhanced memetic algorithm for combinational disruption management in sequence-dependent permutation flowshop. In: Huang DS, Bevilacqua V, Premaratne P (eds) Intelligent Computing Theories and Application: 12th International Conference, ICIC 2016, Lanzhou, China, August 2–5, 2016, Proceedings, Part I, Springer International Publishing, Cham, pp 548–559, URL https://doi.org/10.1007/978-3-319-42291-6_55
247. Liu Y (2008) A memetic algorithm for the probabilistic traveling salesman problem. In: Proceedings of the IEEE Congress on Evolutionary Computation, CEC 2008, June 1–6, 2008, Hong Kong, China, IEEE, pp 146–152, URL https://doi.org/10.1109/CEC.2008.4630790
248. Ljubic I, Raidl GR (2003) A memetic algorithm for minimum-cost vertex-biconnectivity augmentation of graphs. J Heuristics 9(5):401–427
249. Los M, Sawicki J, Smolka M, Schaefer R (2017) Memetic approach for irremediable ill-conditioned parametric inverse problems*. In: Koumoutsakos P, Lees M, Krzhizhanovskaya VV, Dongarra JJ, Sloot PMA (eds) International Conference on Computational Science, ICCS 2017, 12–14 June 2017, Zurich, Switzerland, Elsevier, Procedia Computer Science, vol 108, pp 867–876, URL https://doi.org/10.1016/j.procs.2017.05.007
250. Loucera C, Iglesias A, Gálvez A (2017) Memetic simulated annealing for data approximation with local-support curves. In: ICCS, Elsevier, Procedia Computer Science, vol 108, pp 1364–1373
251. Lozada CAC, Erazo C, Luna J, Mendoza M, Gaviria C, Arteaga C, Paz A (2016) Multi-objective memetic algorithm based on NSGA-II and simulated annealing for calibrating CORSIM micro-simulation models of vehicular traffic flow. In: CAEPIA, Springer, Lecture Notes in Computer Science, vol 9868, pp 468–476
252. Lu Y, Benlic U, Wu Q (2018) A memetic algorithm for the Orienteering Problem with Mandatory Visits and Exclusionary Constraints. European Journal of Operational Research 268(1):54–69
253. Lu Y, Benlic U, Wu Q (2018) A hybrid dynamic programming and memetic algorithm to the traveling salesman problem with hotel selection. Computers & Operations Research 90:193–207

254. Lü Z, Hao J (2010) A memetic algorithm for graph coloring. European Journal of Operational Research 203(1):241–250
255. Luo J, Yang Y, Liu Q, Li X, Chen M, Gao K (2018) A new hybrid memetic multi-objective optimization algorithm for multi-objective optimization. Information Sciences 448:164–186
256. Lust T, Teghem J (2008) MEMOTS: a memetic algorithm integrating tabu search for combinatorial multiobjective optimization. RAIRO - Operations Research 42(1):3–33
257. Ma A, Zhong Y, Zhang L (2014) Remote sensing imagery clustering using an adaptive bi-objective memetic method. In: IEEE Congress on Evolutionary Computation, IEEE, pp 50–57
258. Ma L, Gong M, Liu J, Cai Q, Jiao L (2014) Multi-level learning based memetic algorithm for community detection. Appl Soft Comput 19:121–133
259. Ma L, Gong M, Yan J, Liu W, Wang S (2018) Detecting composite communities in multiplex networks: A multilevel memetic algorithm. Swarm and Evolutionary Computation 39:177–191, URLs https://doi.org/10.1016/j.swevo.2017.09.012, http://www.sciencedirect.com/science/article/pii/S2210650216305156
260. Ma W, Zuo Y, Zeng J, Liang S, Jiao L (2014) A memetic algorithm for solving flexible job-shop scheduling problems. In: IEEE Congress on Evolutionary Computation, IEEE, pp 66–73
261. Majdouli MAE, Bougrine S, Rbouh I, Imrani AAE (2017) A comparative study of the EEG signals big optimization problem using evolutionary, swarm and memetic computation algorithms. In: Proceedings of the Genetic and Evolutionary Computation Conference Companion, ACM, New York, NY, USA, GECCO '17, pp 1357–1364, URL http://doi.acm.org/10.1145/3067695.3082489
262. Mandziuk J, Zychowski A (2016) A memetic approach to vehicle routing problem with dynamic requests. Appl Soft Comput 48:522–534
263. Mandziuk J, Wozniczko A, Goss M (2014) A neuro-memetic system for music composing. In: AIAI, Springer, IFIP Advances in Information and Communication Technology, vol 436, pp 130–139
264. Maric M, Stanimirovic Z, Djenic A, Stanojevic P (2014) Memetic algorithm for solving the multilevel uncapacitated facility location problem. Informatica, Lith Acad Sci 25(3):439–466
265. Marinakis Y, Politis M, Marinaki M, Matsatsinis NF (2015) A memetic-grasp algorithm for the solution of the orienteering problem. In: MCO (2), Springer, Advances in Intelligent Systems and Computing, vol 360, pp 105–116
266. Martínez LMS, Cobos CA, Corrales JC (2017) Memetic Algorithm Based on Global-Best Harmony Search and Hill Climbing for Part of Speech Tagging. In: International Conference on Mining Intelligence and Knowledge Exploration, Springer, Lecture Notes in Computer Science, vol 10682, pp 198–211
267. Matsui T, Katagiri Y, Katagiri H, Kato K (2015) Automatic feature point selection through hybrid metaheuristics based on tabu search and memetic algorithm for augmented reality. In: KES, Elsevier, Procedia Computer Science, vol 60, pp 1120–1127
268. Mavrovouniotis M, Yang S (2011) A memetic ant colony optimization algorithm for the dynamic travelling salesman problem. Soft Comput 15(7):1405–1425, URL https://doi.org/10.1007/s00500-010-0680-1
269. Mavrovouniotis M, Müller FM, Yang S (2015) An ant colony optimization based memetic algorithm for the dynamic travelling salesman problem. In: Silva S, Esparcia-Alcázar AI (eds) Proceedings of the Genetic and Evolutionary Computation Conference, GECCO 2015, Madrid, Spain, July 11–15, 2015, ACM, pp 49–56, URL http://doi.acm.org/10.1145/2739480.2754651
270. Mendes A, Franca P, Moscato P (2002) Fitness landscapes for the total tardiness single machine scheduling problem. Neural Network World 2(2):165–180
271. Mendes A, Cotta C, Garcia V, Franca P, Moscato P (2005) A new memetic algorithm for ordering datasets: Applications in microarray analysis. In: Proceedings of ICPP2005 - 34th International Conference on Parallel Processing, Oslo, Norway, pp 604–611
272. Mendes AS, Muller FM, Franca PM, Moscato P (2002) Comparing meta-heuristic approaches for parallel machine scheduling problems. Production Planning & Control 13(2):143–154, URL http://dx.doi.org/10.1080/09537280110069649

273. Mendoza M, Lozada CAC, León-Guzmán E, Lozano M, Rodríguez FJ, Herrera-Viedma E (2014) A new memetic algorithm for multi-document summarization based on CHC algorithm and greedy search. In: MICAI (1), Springer, Lecture Notes in Computer Science, vol 8856, pp 125–138
274. Meng Z, Pan J (2016) Monkey king evolution: A new memetic evolutionary algorithm and its application in vehicle fuel consumption optimization. Knowl-Based Syst 97:144–157
275. Merz P (2002) A comparison of memetic recombination operators for the traveling salesman problem. In: Langdon WB, Cantú-Paz E, Mathias KE, Roy R, Davis D, Poli R, Balakrishnan K, Honavar V, Rudolph G, Wegener J, Bull L, Potter MA, Schultz AC, Miller JF, Burke EK, Jonoska N (eds) GECCO 2002: Proceedings of the Genetic and Evolutionary Computation Conference, New York, USA, 9–13 July 2002, Morgan Kaufmann, pp 472–479
276. Merz P, Freisleben B (1998) Memetic algorithms and the fitness landscape of the graph bi-partitioning problem. In: PPSN, Springer, Lecture Notes in Computer Science, vol 1498, pp 765–774
277. Merz P, Freisleben B (1999) Fitness landscapes and memetic algorithm design. In: Corne D, Dorigo M, Glover F (eds) New Ideas in Optimization, McGraw-Hill, pp 245–260
278. Merz P, Freisleben B (2000) Fitness landscapes, memetic algorithms, and greedy operators for graph bipartitioning. Evolutionary Computation 8(1):61–91
279. Merz P, Freisleben B (2002) Memetic algorithms for the traveling salesman problem. Complex Systems 13(4), URL http://www.complex-systems.com/abstracts/v13_i04_a01.html
280. Mirsaleh MR, Meybodi MR (2015) A learning automata-based memetic algorithm. Genetic Programming and Evolvable Machines 16(4):399–453
281. Mirsaleh MR, Meybodi MR (2016) A Michigan memetic algorithm for solving the community detection problem in complex network. Neurocomputing 214:535–545
282. Mirsaleh MR, Meybodi MR (2016) A new memetic algorithm based on cellular learning automata for solving the vertex coloring problem. Memetic Computing 8(3):211–222
283. Mirsaleh MR, Meybodi MR (2018) Assignment of cells to switches in cellular mobile network: a learning automata-based memetic algorithm. Applied Intelligence URL https://link.springer.com/10.1007/s10489-018-1136-z
284. Mirsaleh MR, Meybodi MR (2018) Balancing exploration and exploitation in memetic algorithms: A learning automata approach. Computational Intelligence 34(1):282–309
285. Mirsaleh MR, Meybodi MR (2018) A Michigan memetic algorithm for solving the vertex coloring problem. Journal of Computational Science 24:389 – 401, URLs https://doi.org/10.1016/j.jocs.2017.10.005, http://www.sciencedirect.com/science/article/pii/S1877750317301680
286. Mishra KK, Tripathi A, Tiwari S, Saxena N (2017) Evolution based memetic algorithm and its application in software cost estimation. Journal of Intelligent and Fuzzy Systems 32(3):2485–2498
287. Mišković S (2017) Memetic Algorithm for the Uncapacitated Multiple Allocation p-Hub Center Problem. The IPSI BgD Transactions on Internet Research 13(1)
288. Moalic L, Gondran A (2015) The new memetic algorithm HEAD for graph coloring: An easy way for managing diversity. In: EvoCOP, Springer, Lecture Notes in Computer Science, vol 9026, pp 173–183
289. Moalic L, Gondran A (2018) Variations on memetic algorithms for graph coloring problems. J Heuristics 24(1):1–24
290. Mohammadi S, Namadchian A (2017) A New Deep Learning Approach for Anomaly Base IDS using Memetic Classifier. International Journal of Computers, Communications & Control 12(5)
291. Mohammed TA, Sahmoud S, Bayat O (2017) Efficient hybrid memetic algorithm for multi-objective optimization problems. In: 2017 International Conference on Engineering and Technology (ICET), pp 1–6, https://doi.org/10.1109/ICEngTechnol.2017.8308178
292. Moisi EV, Lukic T, Nagy B, Cretu V (2015) Comparing memetic and simulated annealing approaches for discrete tomography on the triangular grid. In: SACI, IEEE, pp 523–528

293. Molina D, Herrera F, Lozano M (2005) Adaptive local search parameters for real-coded memetic algorithms. In: Congress on Evolutionary Computation, IEEE, pp 888–895
294. Molina G, Alba E (2008) Wireless sensor network deployment using a memetic simulated annealing. In: SAINT, IEEE Computer Society, pp 237–240
295. Moll PP (2017) Construction and application of a memetic algorithm assigning catchment areas to retailers for consumer parcel flow. Master's thesis, URL http://hdl.handle.net/2105/38120
296. Moscato P (1989) On evolution, search, optimization, genetic algorithms and martial arts. towards memetic algorithms. Tech. Rep. Caltech Concurrent Computation Program, Tec. Rep. 826, California Institute of Technology, Pasadena, California, USA
297. Moscato P (1993) An introduction to population approaches for optimization and hierarchical objective functions: A discussion on the role of tabu search. Annals OR 41(2):85–121, URL https://doi.org/10.1007/BF02022564
298. Moscato P (1999) Memetic algorithms: A short introduction. In: Corne D, Dorigo M, Glover F (eds) New Ideas in Optimization, McGraw-Hill, pp 219–234
299. Moscato P (2001) Problemas de otimizacão np, aproximabilidade e computacão evolutiva: da prática à teoria. PhD thesis, Departamento de Engenharia de Sistemas (DENSIS), Universidade de Campinas (UNICAMP), UNICAMP, an optional note
300. Moscato P (2012) Memetic algorithms: The untold story. In: Neri F, Cotta C, Moscato P (eds) Handbook of Memetic Algorithms, Studies in Computational Intelligence, vol 379, Springer, pp 275–309, URL https://doi.org/10.1007/978-3-642-23247-3_17
301. Moscato P, Cotta C (2003) A gentle introduction to memetic algorithms. In: Handbook of metaheuristics, Springer, pp 105–144
302. Moscato P, Cotta C (2007) Memetic algorithms. In: Gonzalez TF (ed) Handbook of Approximation Algorithms and Metaheuristics, Chapman & Hall/CRC, Boca Raton, pp 27–1–27–12, URL https://doi.org/10.1201/9781420010749.ch27
303. Moscato P, Cotta C (2010) A modern introduction to memetic algorithms. In: Gendreau M, Potvin JY (eds) Handbook of Metaheuristics, Springer US, Boston, MA, pp 141–183, URL https://doi.org/10.1007/978-1-4419-1665-5_6
304. Moscato P, Fontanari J (1990) Stochastic versus deterministic update in simulated annealing. Physics Letters A 146(4):204–208, URLs http://dx.doi.org/10.1016/0375-9601(90)90166-L, http://www.sciencedirect.com/science/article/pii/037596019090166L
305. Moscato P, Norman MG (1992) A "memetic" approach for the travelling salesman problem: Implementation of a computational ecology for combinatorial optimization on message-passing systems. In: Valero M, Onate E, Jane M, Larriba JL, Suarez B (eds) Proceedings of PACTA '92 - International Conference on Parallel Computing and Transputer Applications, Sep. 21–25, Barcelona, IOS Press, Amsterdam, pp 177–186
306. Moscato P, Norman MG (1998) On the performance of heuristics on finite and infinite fractal instances of the Euclidean traveling salesman problem. INFORMS Journal on Computing 10(2):121–132, URL https://doi.org/10.1287/ijoc.10.2.121
307. Moscato P, Tinetti F (1992) Blending heuristics with a population-based approach: A memetic algorithm for the traveling salesman problem. Report 92–12, Universidad Nacional de La Plata, C.C. 75, 1900 La Plata, Argentina
308. Moscato P, Cotta C, Mendes A (2004) Memetic algorithms. In: Onwubolu G, Babu B (eds) New Optimization Techniques in Engineering, Studies in Fuzziness and Soft Computing, vol 141, Springer Berlin Heidelberg, Berlin, Heidelberg, pp 53–85
309. Moscato P, Mendes A, Cotta C (2004) Scheduling and production & control: Ma. In: Onwubolu G, Babu B (eds) New Optimization Techniques in Engineering, Studies in Fuzziness and Soft Computing, vol 141, Springer Berlin Heidelberg, Berlin, Heidelberg, pp 655–680
310. Moscato P, Mendes A, Linhares A (2004) VLSI design: gate matrix layout problem. In: Onwubolu G, Babu B (eds) New Optimization Techniques in Engineering, Studies in Fuzziness and Soft Computing, vol 141, Springer Berlin Heidelberg, Berlin, Heidelberg, pp 455–478

311. Moscato P, Berretta R, Mendes A (2005) A new memetic algorithm for ordering datasets: Applications in microarray analysis. In: Proceedings of MIC2005 - The 6th Metaheuristics International Conference, Vienna, Austria, pp 695–700
312. Moscato P, Mendes A, Berretta R (2007) Benchmarking a memetic algorithm for ordering microarray data. Biosystems 88(1–2):56–75, URL https://doi.org/10.1016/j.biosystems.2006.04.005
313. Moscato P, Berretta R, Cotta C (2010) Memetic algorithms. In: Cochran JJ, Cox LA, Keskinocak P, Kharoufeh JP, Smith JC (eds) Wiley Encyclopedia of Operations Research and Management Science, John Wiley & Sons, Inc., URL http://dx.doi.org/10.1002/9780470400531.eorms0515
314. Mu C, Xie J, Liu R, Jiao L (2014) A memetic algorithm using local structural information for detecting community structure in complex networks. In: IEEE Congress on Evolutionary Computation, IEEE, pp 680–686
315. Mu C, Xie J, Liu Y, Chen F, Liu Y, Jiao L (2015) Memetic algorithm with simulated annealing strategy and tightness greedy optimization for community detection in networks. Appl Soft Comput 34:485–501
316. Naeni LM, de Vries NJ, Reis R, Arefin AS, Berretta R, Moscato P (2014) Identifying communities of trust and confidence in the charity and not-for-profit sector: A memetic algorithm approach. In: BDCloud, IEEE Computer Society, pp 500–507
317. Naeni LM, Craig H, Berretta R, Moscato P (2016) A novel clustering methodology based on modularity optimisation for detecting authorship affinities in Shakespearean era plays. PLOS ONE 11(8):1–27, URL https://doi.org/10.1371/journal.pone.0157988
318. Nalepa J, Blocho M (2016) Adaptive memetic algorithm for minimizing distance in the vehicle routing problem with time windows. Soft Comput 20(6):2309–2327
319. Nalepa J, Blocho M (2017) A Parallel Memetic Algorithm for the Pickup and Delivery Problem with Time Windows. In: PDP, IEEE, pp 1–8
320. Nalepa J, Kawulok M (2016) Adaptive memetic algorithm enhanced with data geometry analysis to select training data for SVMs. Neurocomputing 185:113–132
321. Nalepa J, Cwiek M, Zak L (2018) Behind the scenes of deadline24: A memetic algorithm for the modified job shop scheduling problem. In: Gruca A, Czachórski T, Harezlak K, Kozielski S, Piotrowska A (eds) Man-Machine Interactions 5, Springer International Publishing, Cham, pp 502–512
322. Naveen N, Rao MC (2016) Bankruptcy prediction using memetic algorithm. In: MIWAI, Springer, Lecture Notes in Computer Science, vol 10053, pp 153–161
323. Neri F (2012) Diversity Management in Memetic Algorithms, Springer Berlin Heidelberg, Berlin, Heidelberg, pp 153–165. URL https://doi.org/10.1007/978-3-642-23247-3_10
324. Neri F, Cotta C (2012) Memetic algorithms and memetic computing optimization: A literature review. Swarm and Evolutionary Computation 2:1–14, URL https://doi.org/10.1016/j.swevo.2011.11.003
325. Neri F, Kotilainen N, Vapa M (2007) An adaptive global-local memetic algorithm to discover resources in P2P networks. In: EvoWorkshops, Springer, Lecture Notes in Computer Science, vol 4448, pp 61–70
326. Neri F, Cotta C, Moscato P (eds) (2012) Handbook of Memetic Algorithms, Studies in Computational Intelligence, vol 379. Springer, URL https://doi.org/10.1007/978-3-642-23247-3
327. Neto JC, Colanzi TE, Amaral AMMM (2017) Application of memetic algorithms in the search-based product line architecture design: An exploratory study. In: ICEIS (2), SciTePress, pp 178–189
328. Nguyen ML, Hui SC, Fong ACM (2017) Submodular memetic approximation for multiobjective parallel test paper generation. IEEE Trans Cybernetics 47(6):1562–1575
329. Nguyen PTH, Sudholt D (2018) Memetic Algorithms Beat Evolutionary Algorithms on the Class of Hurdle Problems. ArXiv e-prints 1804.06173
330. Nguyen QH, Ong Y, Lim M, Krasnogor N (2009) Adaptive cellular memetic algorithms. Evolutionary Computation 17(2):231–256

331. Ni J, Wang K, Cao Q, Khan Z, Fan X (2017) A memetic algorithm with variable length chromosome for robot path planning under dynamic environments. International Journal of Robotics and Automation 32(4), URL http://www.actapress.com/PDFViewer.aspx?paperId=45632
332. Nogueras R, Cotta C (2015) Studying fault-tolerance in island-based evolutionary and multimemetic algorithms. J Grid Comput 13(3):351–374, URL https://doi.org/10.1007/s10723-014-9315-6
333. Nogueras R, Cotta C (2016) A study of the performance of self-⋆ memetic algorithms on heterogeneous ephemeral environments. In: Handl J, Hart E, Lewis PR, López-Ibáñez M, Ochoa G, Paechter B (eds) Parallel Problem Solving from Nature - PPSN XIV - 14th International Conference, Edinburgh, UK, September 17–21, 2016, Proceedings, Springer, Lecture Notes in Computer Science, vol 9921, pp 91–100, URL https://doi.org/10.1007/978-3-319-45823-6_9
334. Nogueras R, Cotta C (2016) Studying self-balancing strategies in island-based multimemetic algorithms. J Computational Applied Mathematics 293:180–191, URL https://doi.org/10.1016/j.cam.2015.03.047
335. Nogueras R, Cotta C (2017) Self-healing strategies for memetic algorithms in unstable and ephemeral computational environments. Natural Computing 16(2):189–200, URL https://doi.org/10.1007/s11047-016-9560-7
336. Norman M, Moscato P (1989) A competitive and cooperative approach to complex combinatorial search. Tech. Rep. Caltech Concurrent Computation Program, Report. 790, California Institute of Technology, Pasadena, California, USA, expanded version published at the 20th Informatics and Operations Research Meeting, Buenos Aires (20th JAIIO), Aug. 1991, pp. 3.15–3.29
337. Norman MG, Moscato P (1995) The Euclidean traveling salesman problem and a space-filling curve. Chaos, Solitons & Fractals 6:389–397, URLs http://dx.doi.org/10.1016/0960-0779(95)80046-J, http://www.sciencedirect.com/science/article/pii/096007799580046J, complex Systems in Computational Physics
338. Ochoa G, Veerapen N, Whitley D, Burke EK (2015) The multi-funnel structure of TSP fitness landscapes: A visual exploration. In: Bonnevay S, Legrand P, Monmarché N, Lutton E, Schoenauer M (eds) Artificial Evolution - 12th International Conference, Evolution Artificielle, EA 2015, Lyon, France, October 26–28, 2015. Revised Selected Papers, Springer, Lecture Notes in Computer Science, vol 9554, pp 1–13, URL https://doi.org/10.1007/978-3-319-31471-6_1
339. Okonji EA, Oluwatoyin YM, Patricia OI (2017) Intelligence classification of the timetable problem: A memetic approach. International Journal on Data Science and Technology 3(2):24
340. Ong Y, Lim M, Zhu N, Wong KW (2006) Classification of adaptive memetic algorithms: a comparative study. IEEE Trans Systems, Man, and Cybernetics, Part B 36(1):141–152
341. Ong Y, Krasnogor N, Ishibuchi H (2007) Special issue on memetic algorithms. IEEE Trans Systems, Man, and Cybernetics, Part B 37(1):2–5, URL https://doi.org/10.1109/TSMCB.2006.883274
342. Ong Y, Lim M, Neri F, Ishibuchi H (2009) Special issue on emerging trends in soft computing: memetic algorithms. Soft Comput 13(8–9):739–740, URL https://doi.org/10.1007/s00500-008-0353-5
343. Ong Y, Lim M, Chen X (2010) Memetic computation - past, present & future [research frontier]. IEEE Comp Int Mag 5(2):24–31, URL https://doi.org/10.1109/MCI.2010.936309
344. Orito Y, Izawa H, Mardyla G, Okamura M (2017) Consumption loan planning by using memetic algorithm. In: Proceedings of the 2017 International Conference on Intelligent Systems, Metaheuristics & Swarm Intelligence, ACM, New York, NY, USA, ISMSI '17, pp 6–10, URL http://doi.acm.org/10.1145/3059336.3059343
345. Osaba E, Díaz F (2012) Comparison of a memetic algorithm and a tabu search algorithm for the traveling salesman problem. In: FedCSIS, pp 131–136
346. Özcan E, Parkes AJ, Alkan A (2012) The interleaved constructive memetic algorithm and its application to timetabling. Computers & OR 39(10):2310–2322, URL https://doi.org/10.1016/j.cor.2011.11.020

347. Özcan E, Drake JH, Altintas C, Asta S (2016) A self-adaptive multimeme memetic algorithm co-evolving utility scores to control genetic operators and their parameter settings. Appl Soft Comput 49:81–93
348. Paechter B, Cumming A, Norman MG, Luchian H (1995) Extensions to a memetic timetabling system. In: [62], pp 251–265, URL https://doi.org/10.1007/3-540-61794-9_64
349. Palacios JJ, Vela CR, González-Rodríguez I, Puente J (2017) A memetic algorithm for due-date satisfaction in fuzzy job shop scheduling. In: Ferrández Vicente JM, Álvarez-Sánchez JR, de la Paz López F, Toledo Moreo J, Adeli H (eds) Natural and Artificial Computation for Biomedicine and Neuroscience, Springer International Publishing, pp 135–145
350. Palacios JJ, Vela CR, Rodríguez IG, Puente J (2017) A memetic algorithm for due-date satisfaction in fuzzy job shop scheduling. In: IWINAC (1), Springer, Lecture Notes in Computer Science, vol 10337, pp 135–145
351. Palar PS, Tsuchiya T, Parks GT (2015) Comparison of scalarization functions within a local surrogate assisted multi-objective memetic algorithm framework for expensive problems. In: CEC, IEEE, pp 862–869
352. Palar PS, Dwianto YB, Zuhal LR, Tsuchiya T (2016) Framework for robust optimization combining surrogate model, memetic algorithm, and uncertainty quantification. In: ICSI (1), Springer, Lecture Notes in Computer Science, vol 9712, pp 48–55
353. Palar PS, Tsuchiya T, Parks GT (2016) A comparative study of local search within a surrogate-assisted multi-objective memetic algorithm framework for expensive problems. Appl Soft Comput 43:1–19
354. Panteleev AV, Pis'mennaya VA (2018) Application of a Memetic Algorithm for the Optimal Control of Bunches of Trajectories of Nonlinear Deterministic Systems with Incomplete Feedback. Journal of Computer and Systems Sciences International 57(1):25–36, URL https://link.springer.com/article/10.1134/S1064230718010082
355. Parsa NR, Karimi B, Husseini SMM (2017) An improved memetic algorithm to minimize the earliness–tardiness on a single batch processing machine. Journal of Industrial and Systems Engineering URL https://pdfs.semanticscholar.org/2342/25753bc213e96f7b71a990ee1299f005ea30.pdf
356. Patil S, Kulkarni S (2018) Mining social media data for understanding students' learning experiences using memetic algorithm. Materials Today: Proceedings 5(1, Part 1):693–699, URL http://www.sciencedirect.com/science/article/pii/S2214785317323817, international Conference on Processing of Materials, Minerals and Energy (July 29th–30th) 2016, Ongole, Andhra Pradesh, India
357. Pecháč P, Sága M (2017) Memetic Algorithm with Normalized RBF ANN for Approximation of Objective Function and Secondary RBF ANN for Error Mapping. Procedia Engineering 177:540–547, URLs https://doi.org/10.1016/j.proeng.2017.02.258, http://www.sciencedirect.com/science/article/pii/S187770581730766X, xXI Polish-Slovak Scientific Conference Machine Modeling and Simulations MMS 2016.September 6–8, 2016, Hucisko, Poland
358. Pecháč P, Sága M, Weis P (2017) Feasibility study of using artificial neural networks for approximation of n-dimensional objective functions in memetic algorithms for structural optimization. Procedia Engineering 192:671–676, URLs https://doi.org/10.1016/j.proeng.2017.06.116, http://www.sciencedirect.com/science/article/pii/S1877705817326620, 12th international scientific conference of young scientists on sustainable, modern and safe transport
359. Pelaez JI, Gomez-Ruiz JA, Veintimilla J, Vaccaro G, Witt P (2017) Memetic computing applied to the design of composite materials and structures. Mathematical Problems in Engineering 2017
360. Peng-jiao Z, Jian-guo L (2017) Deployment optimization of air defense force deployment based on memetic algorithm. In: 2017 29th Chinese Control And Decision Conference (CCDC), pp 5538–5543, https://doi.org/10.1109/CCDC.2017.7979481
361. Peres W, Silva VV, Coelho FC, Junior ICS, Filho JAP (2018) A memetic algorithm for power system damping controllers design. International Journal of Bio-Inspired Computation 11(2):110–122

362. Phan DH, Suzuki J (2014) R2 indicator based multiobjective memetic optimization for the pickup and delivery problem with time windows and demands (PDP-TW-D). In: BICT, ICST
363. Phu-ang A, Thammano A (2017) Memetic algorithm based on marriage in honey bees optimization for flexible job shop scheduling problem. Memetic Computing 9(4):295–309, URL https://doi.org/10.1007/s12293-017-0230-9
364. Pilát M, Neruda R (2011) ASM-MOMA: multiobjective memetic algorithm with aggregate surrogate model. In: IEEE Congress on Evolutionary Computation, IEEE, pp 1202–1208
365. Piwonska A, Koszelew J (2011) A memetic algorithm for a tour planning in the selective travelling salesman problem on a road network. In: Kryszkiewicz M, Rybinski H, Skowron A, Ras ZW (eds) Foundations of Intelligent Systems - 19th International Symposium, ISMIS 2011, Warsaw, Poland, June 28–30, 2011. Proceedings, Springer, Lecture Notes in Computer Science, vol 6804, pp 684–694, URL https://doi.org/10.1007/978-3-642-21916-0_72
366. Pop PC, Hu B, Raidl GR (2013) A memetic algorithm with two distinct solution representations for the partition graph coloring problem. In: EUROCAST (1), Springer, Lecture Notes in Computer Science, vol 8111, pp 219–226
367. Pourrahimian P (2017) A new memetic algorithm for mitigating tandem automated guided vehicle system partitioning problem. Journal of Industrial Engineering International, URL https://doi.org/10.1007/s40092-017-0247-1
368. Qi Y, Hou Z, Li H, Huang J, Li X (2015) A decomposition based memetic algorithm for multi-objective vehicle routing problem with time windows. Computers & OR 62:61–77
369. Qin Y, Liu J (2017) Emergency materials scheduling in disaster relief based on a memetic algorithm. In: Liu D, Xie S, Li Y, Zhao D, El-Alfy ESM (eds) Neural Information Processing, Springer International Publishing, Cham, pp 279–287
370. Qu X, Zhao W, Feng X, Bai L, Liu B (2018) An improved memetic algorithm with novel level comparison for constrained optimization. In: Xhafa F, Patnaik S, Zomaya AY (eds) Advances in Intelligent Systems and Interactive Applications, Springer International Publishing, Cham, pp 698–704
371. Rakshit P, Banerjee D, Konar A, Janarthanan R (2012) An adaptive memetic algorithm for multi-robot path-planning. In: SEMCCO, Springer, Lecture Notes in Computer Science, vol 7677, pp 248–258
372. Rakshit P, Konar A, Bhowmik P, Goswami I, Das S, Jain LC, Nagar AK (2013) Realization of an adaptive memetic algorithm using differential evolution and q-learning: A case study in multirobot path planning. IEEE Trans Systems, Man, and Cybernetics: Systems 43(4):814–831
373. Rakshit P, Konar A, Das S, Nagar AK (2013) ABC-TDQL: an adaptive memetic algorithm. In: HIMA, IEEE, pp 35–42
374. Ramadan RM, Gasser SM, El-Mahallawy MS, Hammad K, El Bakly AM (2018) A memetic optimization algorithm for multi-constrained multicast routing in ad hoc networks. PLOS ONE 13(3):1–17, URL https://doi.org/10.1371/journal.pone.0193142
375. Ramezani Z, Pourdarvish A, Garmabaki A, Kapur P (2017) Optimal reliability equivalence factor for reliability system improvement using memetic algorithm. International Journal of Reliability, Quality and Safety Engineering 24(06):1740,008
376. Ramírez A, Barbudo R, Romero JR, Ventura S (2016) Memetic algorithms for the automatic discovery of software architectures. In: ISDA, Springer, Advances in Intelligent Systems and Computing, vol 557, pp 437–447
377. Ramírez A, Barbudo R, Romero JR, Ventura S (2017) Memetic algorithms for the automatic discovery of software architectures. In: Madureira AM, Abraham A, Gamboa D, Novais P (eds) Intelligent Systems Design and Applications, Springer International Publishing, Cham, pp 437–447
378. Rezgui D, Siala JC, Aggoune-Mtalaa W, Bouziri H (2017) Application of a memetic algorithm to the fleet size and mix vehicle routing problem with electric modular vehicles. In: [57], pp 301–302, URL http://doi.acm.org/10.1145/3067695.3075608
379. Rezoug A, Bader-El-Den M, Boughaci D (2018) Guided genetic algorithm for the multidimensional knapsack problem. Memetic Computing 10(1):29–42

380. Rincy N (2017) Memetic Computing using Simulated Annealing for Dynamic Vehicle Routing Protocol. International Journal of Scientific Research in Science, Engineering and Technology 3, URL https://pdfs.semanticscholar.org/4a2e/b7e845ccd7d9631d211bedf52f23675cd9ff.pdf
381. Rizzi R, Mahata P, Mathieson L, Moscato P (2010) Hierarchical clustering using the arithmetic-harmonic cut: Complexity and experiments. PLOS ONE 5(12):1–8, URL https://doi.org/10.1371/journal.pone.0014067
382. Rodríguez CDR, Gomez DM, Rey MAM (2017) Forecasting time series from clustering by a memetic differential fuzzy approach: An application to crime prediction. In: 2017 IEEE Symposium Series on Computational Intelligence (SSCI), pp 1–8, https://doi.org/10.1109/SSCI.2017.8285373
383. Rogdakis I, Marinaki M, Marinakis Y, Migdalas A (2017) An island memetic algorithm for real world vehicle routing problems. In: Grigoroudis E, Doumpos M (eds) Operational Research in Business and Economics, Springer International Publishing, Cham, pp 205–223
384. Ruiz-Torrubiano R, Suárez A (2015) A memetic algorithm for cardinality-constrained portfolio optimization with transaction costs. Appl Soft Comput 36:125–142
385. Sabar NR, Aleti A (2017) An adaptive memetic algorithm for the architecture optimisation problem. In: ACALCI, Lecture Notes in Computer Science, vol 10142, pp 254–265
386. Sabar NR, Turky AM, Song A (2016) A multi-memory multi-population memetic algorithm for dynamic shortest path routing in mobile ad-hoc networks. In: PRICAI, Springer, Lecture Notes in Computer Science, vol 9810, pp 406–418
387. Sabar NR, Abawajy J, Yearwood J (2017) Heterogeneous cooperative co-evolution memetic differential evolution algorithm for big data optimization problems. IEEE Trans Evolutionary Computation 21(2):315–327
388. Sabar NR, Kieu LM, Chung E, Tsubota T, de Almeida PEM (2017) A memetic algorithm for real world multi-intersection traffic signal optimisation problems. Engineering Applications of Artificial Intelligence 63:45–53, URLs https://doi.org/10.1016/j.engappai.2017.04.021, http://www.sciencedirect.com/science/article/pii/S0952197617300854
389. Sait SM, Oughali FC, Arafeh AM (2016) Engineering a memetic algorithm from discrete cuckoo search and tabu search for cell assignment of hybrid nanoscale CMOL circuits. Journal of Circuits, Systems, and Computers 25(4)
390. Sales LdPA, Melo CS, Bonates TdOe, Prata BdA (2018) Memetic algorithm for the heterogeneous fleet school bus routing problem. Journal of Urban Planning and Development 144(2):04018,018
391. Salvini C, Monacchia S (2017) A memetic computing approach for unit commitment with energy storage systems. Energy Procedia 107:377–382, URLs https://doi.org/10.1016/j.egypro.2016.12.179, http://www.sciencedirect.com/science/article/pii/S1876610216317684, 3rd International Conference on Energy and Environment Research, ICEER 2016, 7–11 September 2016, Barcelona, Spain
392. Samanlioglu F, Jr WGF, Kurz ME (2008) A memetic random-key genetic algorithm for a symmetric multi-objective traveling salesman problem. Computers & Industrial Engineering 55(2):439–449, URL https://doi.org/10.1016/j.cie.2008.01.005
393. Samma H, Lim CP, Mohamad-Saleh J (2016) A new reinforcement learning-based memetic particle swarm optimizer. Appl Soft Comput 43:276–297
394. Sangamithra B, Neelima P, Kumar MS (2017) A memetic algorithm for multi objective vehicle routing problem with time windows. In: 2017 IEEE International Conference on Electrical, Instrumentation and Communication Engineering (ICEICE), pp 1–8, https://doi.org/10.1109/ICEICE.2017.8191931
395. Sanhueza C, Jimenez F, Berretta R, Moscato P (2017) PasMoQAP: A parallel asynchronous memetic algorithm for solving the multi-objective quadratic assignment problem. In: [3], pp 1103–1110, URL https://doi.org/10.1109/CEC.2017.7969430
396. Santos A, Santos R, Silva M, Figueiredo E, Sales C, Costa JCWA (2017) A global expectation–maximization approach based on memetic algorithm for vibration-based structural damage detection. IEEE Transactions on Instrumentation and Measurement 66(4):661–670, https://doi.org/10.1109/TIM.2017.2663478

397. Saprykina O, Saprykin O (2017) Transport infrastructure optimization method based on a memetic algorithm. In: 2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC), pp 1–6, https://doi.org/10.1109/ITSC.2017.8317869
398. Sasikala S, alias Balamurugan SA, Geetha S (2016) A novel memetic algorithm for discovering knowledge in binary and multi class predictions based on support vector machine. Appl Soft Comput 49:407–422
399. Schauer C, Raidl GR (2014) A memetic algorithm for multi layer hierarchical ring network design. In: PPSN, Springer, Lecture Notes in Computer Science, vol 8672, pp 832–841
400. Schauer C, Prandtstetter M, Raidl GR (2010) A memetic algorithm for reconstructing cross-cut shredded text documents. In: Blesa MJ, Blum C, Raidl GR, Roli A, Sampels M (eds) Hybrid Metaheuristics - 7th International Workshop, HM 2010, Vienna, Austria, October 1–2, 2010. Proceedings, Springer, Lecture Notes in Computer Science, vol 6373, pp 103–117, URL https://doi.org/10.1007/978-3-642-16054-7_8
401. Schmickl T (2017) Fundamentalism in a social learning perspective: A memetic agent model of vegetarianism, social interaction networks and food markets. In: 2017 IEEE Symposium Series on Computational Intelligence (SSCI), pp 1–8, https://doi.org/10.1109/SSCI.2017.8280876
402. Schoenberger J (2017) Scheduling constraints in dial-a-ride problems with transfers: a metaheuristic approach incorporating a cross-route scheduling procedure with postponement opportunities. Public Transport 9(1–2, SI):243–272, URL https://doi.org/10.1007/s12469-016-0139-6
403. Schönberger J, Mattfeld DC, Kopfer H (2004) Memetic algorithm timetabling for non-commercial sport leagues. European Journal of Operational Research 153(1):102–116, URL https://doi.org/10.1016/S0377-2217(03)00102-4
404. Schütze O, Alvarado S, Segura C, Landa R (2017) Gradient subspace approximation: a direct search method for memetic computing. Soft Computing 21(21):6331–6350, URL https://doi.org/10.1007/s00500-016-2187-x
405. Segredo E, Segura C, León C (2014) Fuzzy logic-controlled diversity-based multi-objective memetic algorithm applied to a frequency assignment problem. Eng Appl of AI 30:199–212, URL https://doi.org/10.1016/j.engappai.2014.01.005
406. Segredo E, Paechter B, Hart E, Gonzalez-Vila CI (2016) Hybrid parameter control approach applied to a diversity-based multi-objective memetic algorithm for frequency assignment problems. In: [2], pp 1517–1524, URL https://doi.org/10.1109/CEC.2016.7743969
407. Segura C, Segredo E, Miranda G (2017) The importance of the individual encoding in memetic algorithms with diversity control applied to large Sudoku puzzles. In: [3], pp 2152–2160, URL https://doi.org/10.1109/CEC.2017.7969565
408. Sekar PC, Mangalam H (2018) Third generation memetic optimization technique for energy efficient routing stability and load balancing in MANET. Cluster Computing URL https://link.springer.com/article/10.1007/s10586-017-1524-x
409. Semet Y, Schoenauer M (2005) An efficient memetic, permutation-based evolutionary algorithm for real-world train timetabling. In: Proceedings of the IEEE Congress on Evolutionary Computation, CEC 2005, 2–4 September 2005, Edinburgh, UK, IEEE, pp 2752–2759, URL https://doi.org/10.1109/CEC.2005.1555040
410. Sengupta A, Chakraborti T, Konar A, Kim E, Nagar AK (2012) An adaptive memetic algorithm using a synergy of differential evolution and learning automata. In: IEEE Congress on Evolutionary Computation, IEEE, pp 1–8
411. Serna-Urán CA, Arango-Serna MD, Zapata-Cortés JA, Gómez-Marín CG (2018) An Agent-Based Memetic Algorithm for Solving Three-Level Freight Distribution Problems, Springer International Publishing, Cham, pp 111–131. URL https://doi.org/10.1007/978-3-319-74002-7_6
412. Shahin AA (2015) Memetic elitist Pareto evolutionary algorithm for virtual network embedding. Computer and Information Science 8(2):73–88
413. Shan P, Sheng W (2015) An adaptive memetic algorithm for designing artificial neural networks. In: UIC/ATC/ScalCom, IEEE Computer Society, pp 320–323

414. Shang R, Wang J, Jiao L, Wang Y (2014) An improved decomposition-based memetic algorithm for multi-objective capacitated arc routing problem. Appl Soft Comput 19:343–361
415. Shang R, Yuan Y, Du B, Jiao L (2017) A memetic algorithm based on decomposition and extended search for multi-objective capacitated arc routing problem. In: Shi Y, Tan KC, Zhang M, Tang K, Li X, Zhang Q, Tan Y, Middendorf M, Jin Y (eds) Simulated Evolution and Learning, Springer International Publishing, pp 272–283
416. Shang R, Du B, Dai K, Jiao L, Esfahani AMG, Stolkin R (2018) Quantum-inspired immune clonal algorithm for solving large-scale capacitated arc routing problems. Memetic Computing 10(1):81–102
417. Shao W, Pi D, Shao Z (2017) Memetic algorithm with node and edge histogram for no-idle flow shop scheduling problem to minimize the makespan criterion. Appl Soft Comput 54:164–182
418. Sharifipour H, Shakeri M, Haghighi H (2017) Structural test data generation using a memetic ant colony optimization based on evolution strategies. Swarm and Evolutionary Computation. URLs https://doi.org/10.1016/j.swevo.2017.12.009, http://www.sciencedirect.com/science/article/pii/S2210650217303371
419. Shellshear E, Carlson JS, Bohlin R, Tafuri S (2015) A multi-threaded memetic packing algorithm for the ISO luggage packing problem. In: CASE, IEEE, pp 1509–1514
420. Shen XN, Minku LL, Marturi N, Guo YN, Han Y (2018) A Q-learning-based memetic algorithm for multi-objective dynamic software project scheduling. Information Sciences 428:1–29
421. Sheng W, Chen S, Fairhurst MC, Xiao G, Mao J (2014) Multilocal search and adaptive niching based memetic algorithm with a consensus criterion for data clustering. IEEE Trans Evolutionary Computation 18(5):721–741
422. Sheng W, Chen S, Sheng M, Xiao G, Mao J, Zheng Y (2016) Adaptive multisubpopulation competition and multiniche crowding-based memetic algorithm for automatic data clustering. IEEE Trans Evolutionary Computation 20(6):838–858
423. Sheng W, Shan P, Mao J, Zheng Y, Chen S, Wang Z (2017) An Adaptive Memetic Algorithm With Rank-Based Mutation for Artificial Neural Network Architecture Optimization. {IEEE} Access 5:18,895–18,908
424. Shi H, Tang K, Liu C, Song X, Hu C, Sun J (2017) Memetic-based schedule synthesis for communication on time-triggered embedded systems. International Journal of Distributed Sensor Networks 13(10):1550147717738,167, https://doi.org/10.1177/1550147717738167
425. Shi J, Zhang Q, Tsang E (2017) EB-GLS: an improved guided local search based on the big valley structure. Memetic Computing, URL https://doi.org/10.1007/s12293-017-0242-5
426. Shi Y, Li W, Raman A, Fan S (2017) Optimization of multi-layer optical films with a memetic algorithm and mixed integer programming. ACS Photonics URL http://pubs.acs.org/doi/abs/10.1021/acsphotonics.7b01136
427. Shi Y, Li W, Raman A, Fan S (2018) Memetic algorithm optimization of thin-film photonic structures for thermal and energy applications. In: Conference on Lasers and Electro-Optics, Optical Society of America, p SF2I.8, URL http://www.osapublishing.org/abstract.cfm?URI=CLEO_SI-2018-SF2I.8
428. da Silva Menezes M, Goldbarg MC, Goldbarg EFG (2014) A memetic algorithm for the prize-collecting traveling car renter problem. In: IEEE Congress on Evolutionary Computation, IEEE, pp 3258–3265
429. Silvestrin PV, Ritt M (2017) An iterated tabu search for the multi-compartment vehicle routing problem. Computers & Operations Research 81:192–202, URL https://doi.org/10.1016/j.cor.2016.12.023
430. Sivakumar V, Rekha D (2018) Underwater acoustic sensor node scheduling using an evolutionary memetic algorithm. Journal of Telecommunications and Information Technology (1):88–94
431. Smith JE (2007) Credit assignment in adaptive memetic algorithms. In: GECCO, ACM, pp 1412–1419

432. Smith JE (2012) Estimating meme fitness in adaptive memetic algorithms for combinatorial problems. Evolutionary Computation 20(2):165–188
433. Soncco-Álvarez JL, Muñoz DM, Ayala-Rincón M (2018) Opposition-Based Memetic Algorithm and Hybrid Approach for Sorting Permutations by Reversals. Evolutionary Computation URL https://www.mitpressjournals.org/doi/abs/10.1162/evco_a_00220
434. Soria-Alcaraz JA, Carpio JM, Puga H, Melin P, Terashima-Marín H, Cruz Reyes L, Sotelo-Figueroa MA (2014) Generic memetic algorithm for course timetabling ITC2007. In: Castillo O, Melin P, Pedrycz W, Kacprzyk J (eds) Recent Advances on Hybrid Approaches for Designing Intelligent Systems, Studies in Computational Intelligence, vol 547, Springer, pp 481–492 URL https://doi.org/10.1007/978-3-319-05170-3_33
435. Soukour AA, Devendeville L, Lucet C, Moukrim A (2013) A memetic algorithm for staff scheduling problem in airport security service. Expert Syst Appl 40(18):7504–7512
436. Starke S, Hendrich N, Zhang J (2017) A memetic evolutionary algorithm for real-time articulated kinematic motion. In: [3], pp 2473–2479, URL https://doi.org/10.1109/CEC.2017.7969605
437. Stepaniuk K (2018) Visualization of expressing culinary experience in social network, Memetic approach. Entrepreneurship and Sustainability 5(3):693–702, URL https://doi.org/10.9770/jesi.2018.5.3(21)
438. Sun J, Garibaldi JM, Krasnogor N, Zhang Q (2013) An intelligent multi-restart memetic algorithm for box constrained global optimisation. Evolutionary Computation 21(1):107–147, URL https://doi.org/10.1162/EVCO_a_00068
439. Sun J, Miao Z, Gong D (2017) A surrogate-assisted memetic algorithm for interval multi-objective optimization. In: 2017 IEEE Symposium Series on Computational Intelligence (SSCI), pp 1–6, https://doi.org/10.1109/SSCI.2017.8280977
440. Sun Y, Kirley M, Halgamuge SK (2017) A memetic cooperative co-evolution model for large scale continuous optimization. In: Wagner M, Li X, Hendtlass T (eds) Artificial Life and Computational Intelligence, Springer International Publishing, pp 291–300
441. Sun Y, Liang Y, Zhang Z, Wang J (2017) M-NSGA-II: A memetic algorithm for vehicle routing problem with route balancing. In: IEA/AIE (1), Springer, Lecture Notes in Computer Science, vol 10350, pp 61–71
442. Szwarc K, Boryczka U (2017) A comparative study of different variants of a memetic algorithm for ATSP. In: Nguyen NT, Papadopoulos GA, Jędrzejowicz P, Trawiński B, Vossen G (eds) Computational Collective Intelligence, Springer International Publishing, Cham, pp 76–86
443. Tang J, Lim M, Ong Y (2007) Diversity-adaptive parallel memetic algorithm for solving large scale combinatorial optimization problems. Soft Comput 11(9):873–888
444. Tavrov D (2015) Memetic approach to anonymizing groups that can be approximated by a fuzzy inference system. In: NAFIPS/WConSC, IEEE, pp 1–6
445. Tenne Y, Armfield SW (2009) A framework for memetic optimization using variable global and local surrogate models. Soft Comput 13(8–9):781–793
446. Tesfaldet BT (2008) Automated lecture timetabling using a memetic algorithm. APJOR 25(4):451–475, URL https://doi.org/10.1142/S021759590800181X
447. Ting CK, Liaw RT, Wang TC, Hong TP (2018) Mining fuzzy association rules using a memetic algorithm based on structure representation. Memetic Computing 10(1):15–28
448. Ting CK, Liao XL, Huang YH, Liaw RT (2017) Multi-vehicle selective pickup and delivery using metaheuristic algorithms. Information Sciences 406:146–169, URL https://doi.org/10.1016/j.ins.2017.04.001
449. Torres-Velázquez R, Estivill-Castro V (2002) A memetic algorithm guided by quicksort for the error-correcting graph isomorphism problem. In: EvoWorkshops, Springer, Lecture Notes in Computer Science, vol 2279, pp 173–182
450. Tripathy B, Sooraj T, Mohanty R (2018) Memetic algorithms and their applications in computer science. In: Handbook of Research on Modeling, Analysis, and Application of Nature-Inspired Metaheuristic Algorithms, IGI Global, pp 73–93

451. Turabieh H, Abdullah S (2009) Incorporating tabu search into memetic approach for enrolment-based course timetabling problems. In: DMO, IEEE, pp 115–119
452. Tüű-Szabó B, Földesi P, Kóczy LT (2017) An efficient new memetic method for the traveling salesman problem with time windows. In: Phon-Amnuaisuk S, Ang SP, Lee SY (eds) Multi-disciplinary Trends in Artificial Intelligence, Springer International Publishing, Cham, pp 426–436
453. Tymerski R, Greenwood G, Sills D (2017) Equity option strategy discovery and optimization using a memetic algorithm. In: Wagner M, Li X, Hendtlass T (eds) Artificial Life and Computational Intelligence, Springer International Publishing, Cham, pp 25–38
454. Valencia CE, Martínez FJZ, Pérez SLP (2017) A simple but effective memetic algorithm for the multidimensional assignment problem. In: 2017 14th International Conference on Electrical Engineering, Computing Science and Automatic Control (CCE), pp 1–6, https://doi.org/10.1109/ICEEE.2017.8108889
455. Vega-Alvarado E, Portilla-Flores E, Munoz-Hernandez GA, Mezura-Montes E, Sepúlveda-Cervantes G, Bautista-Camino P (2018) A memetic algorithm based on artificial bee colony for optimal synthesis of mechanisms. Revista Internacional de Métodos Numéricos para Cálculo y Diseño en Ingeniería 34(1), URL https://www.scipedia.com/public/Vega-Alvarado_et_al_2017a
456. Vig V, Palekar US (2008) On estimating the distribution of optimal traveling salesman tour lengths using heuristics. European Journal of Operational Research 186(1):111–119, URLs http://dx.doi.org/10.1016/j.ejor.2006.12.066, http://www.sciencedirect.com/science/article/pii/S0377221707001877
457. Vijayaraju P, Sripathy B, Arivudainambi D, Balaji S (2017) Hybrid memetic algorithm with two-dimensional discrete Haar wavelet transform for optimal sensor placement. IEEE Sensors Journal 17(7):2267–2278, https://doi.org/10.1109/JSEN.2017.2662951
458. Wang C, Ji Z, Wang Y (2017) A Novel Memetic Algorithm Based on Decomposition for Multiobjective Flexible Job Shop Scheduling Problem. Mathematical Problems in Engineering 2017, URL https://www.hindawi.com/journals/mpe/2017/2857564/abs/
459. Wang C, Tian N, Ji Z, Wang Y (2017) Multi-objective fuzzy flexible job shop scheduling using memetic algorithm. Journal of Statistical Computation and Simulation 87(14):2828–2846
460. Wang H, Wang D, Yang S (2009) A memetic algorithm with adaptive hill climbing strategy for dynamic optimization problems. Soft Comput 13(8–9):763–780
461. Wang H, Fu Y, Huang M, Huang GQ, Wang J (2017) A NSGA-II based memetic algorithm for multiobjective parallel flowshop scheduling problem. Computers & Industrial Engineering 113:185–194, URLs https://doi.org/10.1016/j.cie.2017.09.009, http://www.sciencedirect.com/science/article/pii/S0360835217304187
462. Wang J, Cui G, Xiao Y, Luo X, Kabelac S (2017) Bi-level heat exchanger network synthesis with evolution method for structure optimization and memetic particle swarm optimization for parameter optimization. Engineering Optimization 49(3):401–416, URLs 10.1080/0305215X.2016.1191803, http://www.tandfonline.com/doi/abs/10.1080/0305215X.2016.1191803
463. Wang L, Wang X, Sun D, Wang W (2017) The process optimization of train operation based on multi-objective memetic algorithm using incorporated preference information. In: 2017 36th Chinese Control Conference (CCC), pp 2812–2817, https://doi.org/10.23919/ChiCC.2017.8027791
464. Wang P, Liu R, Jiang Z (2018) Optimization of combination chemotherapy with dose adjustment using a memetic algorithm. Information Sciences 432:63–78, URLs https://doi.org/10.1016/j.ins.2017.12.002, http://www.sciencedirect.com/science/article/pii/S0020025516319818
465. Wang S, Gong M, Du H, Ma L, Miao Q, Du W (2016) Optimizing dynamical changes of structural balance in signed network based on memetic algorithm. Social Networks 44:64–73
466. Wang S, Gong M, Li H, Yang J, Wu Y (2017) Memetic algorithm based location and topic aware recommender system. Knowl-Based Syst 131:125–134, https://doi.org/10.1016/j.knosys.2017.05.030, https://doi.org/10.1016/j.knosys.2017.05.030

467. Wang X, Liu J (2017) A memetic algorithm for community detection in bipartite networks. In: Liu D, Xie S, Li Y, Zhao D, El-Alfy ESM (eds) Neural Information Processing, Springer International Publishing, pp 89–99
468. Wang X, Tang L (2017) A machine-learning based memetic algorithm for the multi-objective permutation flowshop scheduling problem. Computers & Operations Research 79:60–77, URLs https://doi.org/10.1016/j.cor.2016.10.003, http://www.sciencedirect.com/science/article/pii/S0305054816302519
469. Wang X, Liu H, Yu Z (2016) A novel heuristic algorithm for IP block mapping onto mesh-based networks-on-chip. The Journal of Supercomputing 72(5):2035–2058, URL https://doi.org/10.1007/s11227-016-1719-6
470. Wang Y, Li J, Gao K, Pan Q (2011) Memetic algorithm based on improved inver-over operator and Lin-Kernighan local search for the Euclidean traveling salesman problem. Computers & Mathematics with Applications 62(7):2743–2754, URL https://doi.org/10.1016/j.camwa.2011.06.063
471. Wang Y, Hao J, Glover F, Lü Z (2014) A tabu search based memetic algorithm for the maximum diversity problem. Eng Appl of AI 27:103–114
472. Wang Y, Chen J, Sun H, Yin M (2017) A memetic algorithm for minimum independent dominating set problem. Neural Computing and Applications, URL https://doi.org/10.1007/s00521-016-2813-7
473. Wang Y, Chen Y, Lin Y (2017) Memetic algorithm based on sequential variable neighborhood descent for the minmax multiple traveling salesman problem. Computers & Industrial Engineering 106:105–122, URL https://doi.org/10.1016/j.cie.2016.12.017
474. Wang Y, Ding Z, Zuo M, Peng L (2017) An improved memetic differential evolution for college students' comprehensive quality evaluation. International Journal of Wireless and Mobile Computing 13(3):193–199
475. Wang Z, Jin H, Tian M (2015) Rank-based memetic algorithm for capacitated arc routing problems. Appl Soft Comput 37:572–584
476. Wei K, Dinneen MJ (2014) Runtime analysis comparison of two fitness functions on a memetic algorithm for the clique problem. In: IEEE Congress on Evolutionary Computation, IEEE, pp 133–140
477. WEN T, HUA Jx, YANG Js, ZHAI Xy (2017) Memetic Differential Evolution with Baldwin Effect and Opposition-Based Learning. In: 2017 2nd International Conference on Computer Science and Technology (CST 2017), DEStech Publications, Inc., URL http://dpi-proceedings.com/index.php/dtcse/article/view/12554
478. Widl M, Musliu N (2014) The break scheduling problem: complexity results and practical algorithms. Memetic Computing 6(2):97–112
479. Wolpert D, Macready W (1997) No free lunch theorems for optimization. IEEE Transactions on Evolutionary Computation 1(1):67–82
480. Wood DA (2017) Gas and oil project time-cost-quality tradeoff: Integrated stochastic and fuzzy multi-objective optimization applying a memetic, nondominated, sorting algorithm. Journal of Natural Gas Science and Engineering 45:143–164, URLs http://dx.doi.org/10.1016/j.jngse.2017.04.033, http://www.sciencedirect.com/science/article/pii/S187551001730224X
481. Wood DA (2018) Thermal maturity and burial history modelling of shale is enhanced by use of Arrhenius time-temperature index and memetic optimizer. Petroleum 4(1):25–42, URLs https://doi.org/10.1016/j.petlm.2017.10.004, http://www.sciencedirect.com/science/article/pii/S2405656117300548
482. Wu J, Chang Z, Yuan L, Hou Y, Gong M (2014) A memetic algorithm for resource allocation problem based on node-weighted graphs [application notes]. IEEE Comp Int Mag 9(2):58–69
483. Wu J, Shen X, Jiao K (2018) Game-Based Memetic Algorithm to the Vertex Cover of Networks. IEEE Transactions on Cybernetics URL https://ieeexplore.ieee.org/abstract/document/8276561/
484. Wu K, Liu J (2017) Evolutionary game network reconstruction by memetic algorithm with l 1/2 regularization. In: Shi Y, Tan KC, Zhang M, Tang K, Li X, Zhang Q, Tan Y, Middendorf

M, Jin Y (eds) Simulated Evolution and Learning, Springer International Publishing, Cham, pp 15–26
485. Wu P, Pan L (2015) Multi-objective community detection based on memetic algorithm. PLOS ONE 10(5):1–31, URL https://doi.org/10.1371/journal.pone.0126845
486. Wu X, Che A (2018) A memetic differential evolution algorithm for energy-efficient parallel machine scheduling. Omega URL https://www.sciencedirect.com/science/article/pii/S0305048317307922
487. Xhafa F, Duran B (2008) Parallel memetic algorithms for independent job scheduling in computational grids. In: Cotta C, van Hemert JI (eds) Recent Advances in Evolutionary Computation for Combinatorial Optimization, Studies in Computational Intelligence, vol 153, Springer, pp 219–239, URL https://doi.org/10.1007/978-3-540-70807-0_14
488. Xhafa F, Alba E, Díaz BD (2007) Efficient batch job scheduling in grids using cellular memetic algorithms. In: 21th International Parallel and Distributed Processing Symposium (IPDPS 2007), Proceedings, 26–30 March 2007, Long Beach, California, USA, IEEE, pp 1–8, URL https://doi.org/10.1109/IPDPS.2007.370437
489. Xhafa F, Alba E, Dorronsoro B, Duran B (2008) Efficient batch job scheduling in grids using cellular memetic algorithms. J Math Model Algorithms 7(2):217–236, URL https://doi.org/10.1007/s10852-008-9076-y
490. Xhafa F, Duran B, Barolli L, Kolici V, Miho R, Takizawa M (2012) Tuning of operators in memetic algorithms for independent batch scheduling in computational grids. In: Barolli L, Xhafa F, Vitabile S, Uehara M (eds) Sixth International Conference on Complex, Intelligent, and Software Intensive Systems, CISIS 2012, Palermo, Italy, July 4–6, 2012, IEEE Computer Society, pp 335–342, URL https://doi.org/10.1109/CISIS.2012.22
491. Xiao C, Dong Z, Xu Y, Meng K, Zhou X, Zhang X (2016) Rational and self-adaptive evolutionary extreme learning machine for electricity price forecast. Memetic Computing 8(3):223–233
492. Xiao J, Yang Y, Ma X, Zhou J, Zhu Z (2016) Multi-objective memetic algorithm for solving pickup and delivery problem with dynamic customer requests and traffic information. In: CEC, IEEE, pp 1964–1970
493. Xu J, Yin Y, Cheng TCE, Wu C, Gu S (2014) A memetic algorithm for the re-entrant permutation flowshop scheduling problem to minimize the makespan. Appl Soft Comput 24:277–283
494. Xue X, Ren A (2017) Using memetic algorithm for matching process models. In: 2017 13th International Conference on Computational Intelligence and Security (CIS), pp 11–15, https://doi.org/10.1109/CIS.2017.00011
495. Xue X, Wang Y (2015) Optimizing ontology alignments through a memetic algorithm using both matchFmeasure and unanimous improvement ratio. Artif Intell 223:65–81
496. Xue X, Wang Y (2016) Using memetic algorithm for instance coreference resolution. IEEE Trans Knowl Data Eng 28(2):580–591
497. Xue X, Wang Y, Ren A (2014) Optimizing ontology alignment through memetic algorithm based on partial reference alignment. Expert Syst Appl 41(7):3213–3222
498. Yan J, Gong M, Ma L, Wang S, Shen B (2016) Structure optimization based on memetic algorithm for adjusting epidemic threshold on complex networks. Appl Soft Comput 49:224–237
499. Yan L, Mei Y, Ma H, Zhang M (2016) Evolutionary web service composition: A graph-based memetic algorithm. In: CEC, IEEE, pp 201–208
500. Yan S, Cai K (2017) A multi-objective multi-memetic algorithm for network-wide conflict-free 4D flight trajectories planning. Chinese Journal of Aeronautics 30(3):1161–1173, URL https://doi.org/10.1016/j.cja.2017.03.008
501. Yan S, Cai K, Zhu Y (2015) A multi-objective memetic algorithm for network-wide flights planning optimization. In: ICTAI, IEEE Computer Society, pp 518–525
502. Yang S, Huang K (2017) Combinatorial optimization and simulation for weapon system portfolio using self-adaptive Memetic algorithm. Journal of Engineering Research 5(1), URL http://kuwaitjournals.org/jer/index.php/JER/article/view/1275

503. Yang Y, Ma X, Sun Y, Zhu Z (2017) Multi-objective memetic algorithm based on three-dimensional request prediction for dynamic pickup-and-delivery problem with time windows. In: Shi Y, Tan KC, Zhang M, Tang K, Li X, Zhang Q, Tan Y, Middendorf M, Jin Y (eds) Simulated Evolution and Learning, Springer International Publishing, Cham, pp 810–820
504. Yang Y, Sun Y, Zhu Z (2017) Multi-objective memetic algorithm based on request prediction for dynamic pickup-and-delivery problems. In: 2017 IEEE Congress on Evolutionary Computation (CEC), pp 1728–1733, https://doi.org/10.1109/CEC.2017.7969510
505. Yannibelli V, Amandi A (2015) Project scheduling: A memetic algorithm with diversity-adaptive components that optimizes the effectiveness of human resources. Polibits 52:93–103
506. Yao T, Yao X, Han S, Wang Y, Cao D, Wang F (2017) Memetic algorithm with adaptive local search for capacitated arc routing problem. In: 2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC), pp 836–841, https://doi.org/10.1109/ITSC.2017.8317903
507. Ye T, Wang T, Lü Z, Hao J (2014) A multi-parent memetic algorithm for the linear ordering problem. CoRR abs/1405.4507
508. Yu Y, Gao S, Cheng S, Wang Y, Song S, Yuan F (2017) CBSO: a memetic brain storm optimization with chaotic local search. Memetic Computing, URL https://doi.org/10.1007/s12293-017-0247-0
509. Yuan Y, Xu H (2015) Multiobjective flexible job shop scheduling using memetic algorithms. IEEE Trans Automation Science and Engineering 12(1):336–353
510. Žalik KR, Žalik B (2018) Memetic algorithm using node entropy and partition entropy for community detection in networks. Information Sciences 445:38–49
511. Zeng R, Wen X (2018) Dynamic categorization of semantics of fashion language: A memetic approach. International Journal of Society, Culture & Language 6(1):101–114
512. Zeng Y, Chen X, Ong Y, Tang J, Xiang Y (2017) Structured memetic automation for online human-like social behavior learning. IEEE Trans Evolutionary Computation 21(1):102–115, URL https://doi.org/10.1109/TEVC.2016.2577593
513. Zeng ZZ, Yu XG, Chen M, Liu YY (2018) A memetic algorithm to pack unequal circles into a square. Computers & Operations Research 92:47–55
514. Zhang C, Hei X, Yang D, Wang L (2016) A memetic particle swarm optimization algorithm for community detection in complex networks. IJPRAI 30(2)
515. Zhang J, Song Y (2017) Mathematical Model and Algorithm for the Reefer Mechanic Scheduling Problem at Seaports. Discrete Dynamics in Nature and Society URL https://doi.org/10.1155/2017/4730253
516. Zhang M, Ma J, Gong M, Li H, Liu J (2017) Memetic algorithm based feature selection for hyperspectral images classification. In: 2017 IEEE Congress on Evolutionary Computation (CEC), pp 495–502, https://doi.org/10.1109/CEC.2017.7969352
517. Zhang X, Zhang X (2017) Thinning of antenna array via adaptive memetic particle swarm optimization. EURASIP Journal on Wireless Communications and Networking 2017(1):183, URL https://doi.org/10.1186/s13638-017-0968-2
518. Zhang Y, Cai Z, Wu J, Wang X, Liu X (2015) A memetic algorithm based extreme learning machine for classification. In: IJCNN, IEEE, pp 1–8
519. Zhang Y, Wu J, Cai Z, Zhang P, Chen L (2016) Memetic extreme learning machine. Pattern Recognition 58:135–148
520. Zhang Y, Mei Y, Tang K, Jiang K (2017) Memetic algorithm with route decomposing for periodic capacitated arc routing problem. Appl Soft Comput 52:1130–1142
521. Zhang Z, Che O, Cheang B, Lim A, Qin H (2013) A memetic algorithm for the multiperiod vehicle routing problem with profit. European Journal of Operational Research 229(3):573–584
522. Zhao H, Xu WA, Jiang R (2015) The memetic algorithm for the optimization of urban transit network. Expert Syst Appl 42(7):3760–3773
523. jian Zhao Z, qing He X, Liu F (2017) An improved multi-objective memetic algorithm for bi-objective permutation flow shop scheduling. In: 2017 International Conference on Service Systems and Service Management, pp 1–6, https://doi.org/10.1109/ICSSSM.2017.7996154

524. Zhou M, Liu J (2014) A memetic algorithm for enhancing the robustness of scale-free networks against malicious attacks. Physica A: Statistical Mechanics and its Applications 410:131–143, URLs http://dx.doi.org/10.1016/j.physa.2014.05.002, http://www.sciencedirect.com/science/article/pii/S0378437114003665
525. Zhou M, Cheng X, Guo X, Gu M, Zhang H, Song X (2016) Improving failure detection by automatically generating test cases near the boundaries. In: 40th IEEE Annual Computer Software and Applications Conference, COMPSAC 2016, Atlanta, GA, USA, June 10–14, 2016, pp 164–173, URL https://doi.org/10.1109/COMPSAC.2016.137
526. Zhou T, Lü Z, Ye T, Zhou K (2017) A memetic algorithm for the linear ordering problem with cumulative costs. In: Gao X, Du H, Han M (eds) Combinatorial Optimization and Applications, Springer International Publishing, Cham, pp 518–526
527. Zhou Y, Hao J, Glover F (2017) Memetic search for identifying critical nodes in sparse graphs. CoRR abs/1705.04119, URL http://arxiv.org/abs/1705.04119
528. Zhou Y, Hao JK, Duval B (2017) Opposition-based memetic search for the maximum diversity problem. IEEE Transactions on Evolutionary Computation 21(5):731–745, https://doi.org/10.1109/TEVC.2017.2674800
529. Zhou Z, Ong Y, Lim M, Lee B (2007) Memetic algorithm using multi-surrogates for computationally expensive optimization problems. Soft Comput 11(10):957–971
530. Zhu Z, Zhou J, Ji Z, Shi Y (2011) DNA sequence compression using adaptive particle swarm optimization-based memetic algorithm. IEEE Trans Evolutionary Computation 15(5):643–658
531. Zhu Z, Xiao J, He S, Ji Z, Sun Y (2016) A multi-objective memetic algorithm based on locality-sensitive hashing for one-to-many-to-one dynamic pickup-and-delivery problem. Inf Sci 329:73–89
532. Zhuang Z, Fan S, Xu H, Zheng J (2016) A memetic algorithm using partial solutions for graph coloring problem. In: CEC, IEEE, pp 3200–3206
533. Zou X, Liu J (2017) A Mutual Information based Two-phase Memetic Algorithm for Large-scale Fuzzy Cognitive Map Learning. IEEE Transactions on Fuzzy Systems pp 1–1, https://doi.org/10.1109/TFUZZ.2017.2764445
534. Zychowski A, Gupta A, Mańdziuk J, Ong YS (2017) Addressing Expensive Multi-objective Games with Postponed Preference Articulation via Memetic Co-evolution. ArXiv e-prints 1711.06763

Chapter 14
A Memetic Algorithm for the Team Orienteering Problem

Dimitra Trachanatzi, Eleftherios Tsakirakis, Magdalene Marinaki, Yannis Marinakis, and Nikolaos Matsatsinis

Abstract The Team Orienteering Problem (TOP) is an expansion of the orienteering problem. The problem's data is a set of nodes and each node is associated with a score value. The goal of the TOP is to construct a discrete number of routes in order to visit the nodes and collect their scores aiming to maximize the total collected score with respect to a total travel time constraint. In this paper we propose a Memetic algorithm with Similarity Operator (MSO-TOP) for solving the TOP. The concept of the "similarity operator" is that feasible sub-routes of the solutions are serving as chromosomes. The efficacy of MSO-TOP was tested using the known benchmark instances for the TOP. From the experiments it was concluded that "similarity operator" is a promising technique and MSO-TOP produces quality solutions.

Keywords Memetic algorithm · Mobile tourist guide · Multiple tour maximum collection problem · Museum visitor routing problem · Team orienteering problem · Similarity operator

14.1 Introduction

The Orienteering Problem (OP) was introduced by Golden et al. [10] when they described an outdoor game played in the mountains or in the forests. The game's formation is simple, a number of players start from a specific point and they have a map that informs them about the checkpoints' position and about the scores associated with each one of them. The players have to visit as many checkpoints

D. Trachanatzi · E. Tsakirakis · M. Marinaki · Y. Marinakis (✉) · N. Matsatsinis
Technical University of Crete, School of Production Engineering and Management, Chania, Greece
e-mail: dtrachanatzi@isc.tuc.gr; magda@dssl.tuc.gr; marinakis@ergasya.tuc.gr; nikos@ergasya.tuc.gr

P. Moscato, N. J. de Vries (eds.), *Business and Consumer Analytics: New Ideas*,
https://doi.org/10.1007/978-3-030-06222-4_14

as possible adding to their personal score, the score of the point they pass from. Within a specific time, their journey ends and they return to a terminal point, the winner is the player who gathered the maximum score. The above game can be transformed from single-competitor to a team-competitor version, by adding a cooperative aspect. Each member of a team has to act upon the rules. Once a team member visits a point and gathers the associated score, his/her teammates are not allowed to visit that point. Thus each team member follows a feasible path visiting a different set of points from the other, aiming to gather as much score as possible with respect to the duration limit of their journey. The later version of the game is also denoted as the Team Orienteering Problem (TOP) [5].

The earliest version of the Team Orienteering Problem was proposed by Butt and Cavalier [4] as the Multiple Tour Maximum Collection Problem (MTMCP). Chao et al. [5] named the problem as TOP. Yet real life applications of the TOP are mentioned in the earlier literature, such as the delivery of home heating fuel, where the urgency of a customer request for fuel is treated as a score by Golden et al. [10]. Other practical applications have been mentioned over the years. Butt et al. [4] described a TOP's application of high schools athlete recruitment. The sport game of orienteering was introduced by Golden and Wasil [6] and the technicians routing to serve profitable customers was proposed by Tang and Miller-Hooks [22].

The most interesting application is the exploitation of the described model in order to create tourist guides for a city. Recent example is the Mobile Tourist Guide [21]. In case that a tourist visits a city but has limited time to explore all the points of interests, he/she is forced to select only the most valuable attractions. Moreover, the visits have to be ordered in a feasible path within his/her available time span. Vansteenwegen and Van Oudheusden [27] defined this kind of problem as the tourist trip design problem (TTDP). The OP is the simplest TTDP and TOP is the variant of TTDP for a multiple day visit. Another application following the same logic is the Museum Visitor Routing Problem (MVRP) proposed by Vincent et al. [28].

The present paper is an effort to extend the previous study of Memetic-GRASP algorithm (MemGRASP) for the solution of the Orienteering Problem by Marinakis et al. [15] in a more complicated problem, such as the team orienteering problem. A new approach of the memetic algorithm for solving the TOP is proposed by the authors, a Memetic algorithm with Similarity Operator (MSO-TOP). The population in the proposed algorithm is generated through the "similarity operator", which plays the role of a crossover operator and each chromosome is a sub-route from one of the parent solutions.

The rest of this paper is organized as follows. In Sect. 14.2, a brief literature review is presented along with the mathematical formulation of the Team Orienteering Problem. Section 14.3 consists of a sort introduction to the Memetic Algorithms and the analytical description of our proposed algorithm. The computational results on the standard benchmark instances are provided in Sect. 14.4. In the last section, conclusions and future research are presented.

14.2 Team Orienteering Problem

The Team Orienteering Problem (TOP) belongs to the class of Vehicle Routing Problem with Profits (VRPPs) and constitutes an extension of the Orienteering Problem (OP), as we described above. Constructing the problem, a set of customers with specific travel times between them and a set of score associated with each one of the customers are given. Thus, the problem's goal is to cluster the customers' set in a specified number of routes and order the visits in every route, while satisfying a maximum duration constraint aiming to maximize the total collected score.

14.2.1 Brief Literature Review

Several studies for solving TOP with exact algorithms have been published, but after Chao proved that it is an NP-hard problem [5], researches tend to focus on more suitable approaches such as heuristic and metaheuristic algorithms. At first, Butt and Cavalier [4] developed a mixed integer programming (MIP) model and a simple construction algorithm using the score/distance ratio which was improved later using a column generation method. Chao et al. [5] proposed a heuristic approach for the TOP where a set of routes are created using the cheapest insertion rule and refined them using customer (1-point movement, 2-point exchange, 2-opt) and restart strategies [23].

In the twenty first century several metaheuristics were applied to the TOP. Tang and Miller-Hooks [22] proposed a MIP model along with a tabu search algorithm included in an adaptive memory procedure (AMP) that alternates between small and large neighbourhoods during the search. They, also, implemented greedy and random heuristics in order to generate solutions in the neighbourhood. Their algorithm used more computation time than Chao's but the solutions were improved. In 2008, Archetti et al. [1] developed two variants of a generalized tabu search algorithm (the second allows infeasible solutions with penalty), a Slow and a Fast Variable Neighbourhood Search algorithm (SVN and FVN). The later approach performed better than all the previous. Ke et al. [12] used an Ant Colony Optimization (ACO) approach, where each ant constructs a candidate solution and in the framework of ACO, four different methods have been developed to achieve it. More precisely, they proposed the sequential method (ASe) that constructs in sequence complete paths, the random-concurrent method (ARC) that inserts new nodes into a randomly chosen path, the deterministic-concurrent method (ADC), that implements fixed sequence of paths and the simultaneous method (ASi) that fills all paths up to their limit by inserting nodes at every iteration. The more efficient method is the ASe which is faster and reaches a higher average score. Examining it against all other methods that were mentioned above, only FVN needs less computational time than ASe and only SVN shows smaller average gap compared to the best so far known solution [24]. Vansteenwegen et al. [25] implemented a Guided Local Search (GLS) framework and later a Skewed Variable Neighbourhood Search (SVNS) framework [26]. Both algorithms apply a greedy construction phase prior to a combination of intensification and diversification procedures designed to increase the score and to decrease the travel time in a path. The SVNS algorithm is faster and more efficient than GLS algorithm. In 2009 Souffriau et al. [20] proposed two variants of a Greedy Randomized Adaptive Search Procedure (GRASP) with Path Relinking, a slow version (SPR) with excellent results and a fast version (FPR) with quality results. In this GRASP algorithm, four procedures are executed in sequence for a fixed number of iterations. The algorithm includes a construction procedure, a number of local search procedures able to reach optimal results in the searching neighbourhood and a path relinking procedure that provides a pool of elite solutions by adding a long-term memory component. The path relinking creates infeasible solutions and transforms them through local search to feasible ones. The final procedure updates the elite pool. The comparison with different approaches shows small gaps between the obtained optimal solutions. A memetic algorithm approach for team orienteering problem was proposed by Bouly et al. [3]. They actually developed a simple hybrid genetic algorithm that makes use of a procedure called Optimal Split to evaluate the chromosomes and local search techniques for mutation. Initially, their algorithm generates the population by an Iterative Destruction/Construction Heuristic (IDCH) based on the Destruct and Repair operator that removes a small part of the solution aiming to rebuild an improved solution. At each iteration, the parents are chosen by the Binary Tournament and the LOX crossover operator is used to create a child chromosome. On the purpose to create new chromosomes a giant tour (including all customers) is created and, then, the Optimal Split is performed utilizing a modified version of the program evaluation and review technique/critical path method (PERT/CPM). In addition a child chromosome has a probability of being mutated by a set of LS techniques. MA performs better than the slow VNS algorithm of Archetti [1] and the algorithm's stability is equivalent to ACO's [12].

In 2012, Lin [14] described a multi-start simulated annealing (MSA) algorithm, combining a Simulated Annealing (SA) based meta-heuristic with a multi-start hill climbing strategy which outperforms traditional single-start SA. MSA algorithm reached best solutions in the 0.85% of the benchmark problems. Dang et al. [7] proposed an effective Particle Swarm Optimization (PSO)-based Memetic Algorithm (PSOMA) for the TOP. The PSOMA includes optimal tour-splitting techniques, genetic crossover operators and, also, an intensive use of local search techniques. Experiments show that PSOMA attained best known solution gap of 0.016%. Other studies with PSO variants were published by Bonnefoy [2], who developed a PSO algorithm combined with a linear programming technique and Muthuswamy and Lam [18] suggest a discrete version of PSO (DPSO). Dang et al. [8] based on their previous research developed a PSO-inspired algorithm (PSOiA) based on an interval graph model with result to detect all of the best known solutions and additionally improved one of them. At last, Kim et al. [13] proposed an augmented large neighbourhood search method with three improvement algorithms: a local search, a shift and insertion improvement and a replacement improvement. Their algorithm detects the best known solutions for the 386 out of the 387 benchmark instances and only one of them has a 1 point divergence. To the best of our knowledge the results obtained by the last two approaches are the most efficient in terms of solution quality and computation time.

14.2.2 TOP Mathematical Formulation

The Team Orienteering Problem (TOP) can be described using a complete graph $G = (V, A)$, where $V = \{1, \cdots, N\}$ is the set of nodes and $A = \{(i, j)|i, j \in V\}$ is the set of arcs able to connect the nodes of V set. Every node i included in V set is related to a score s_i, in addition the required travel time t_{ij} between nodes is given for every pair (i, j) where the starting and the ending node has score equal to zero. A limited number of M vehicle routes have to be formed. Each route is a feasible path with respect to a pre-specified travelling duration limit $Tmax$. Each path should start from the initial node 1, while ending to the final node N, each node is visited at most once and belongs to exactly one path. The objective is to identify the set of M paths that maximize the total $Score$ obtained by the visited nodes. Using the following set of decision variables the problem can be formulated as an integer problem [12]:

- $y_{id} = 1$ if node i ($i \in \{1, \ldots, N\}$) belongs to vehicle route d ($d \in \{1, \ldots, M\}$), $y_{id} = 0$ otherwise.
- $x_{ijd} = 1$ if vehicle route d ($d \in \{1, \ldots, M\}$) includes edges i, j ($i, j \in \{1, \ldots, N\}$), $x_{ijd} = 0$ otherwise. The problem's symmetry ensures that $t_{ij} = t_{ji}$, thus, only x_{ijd} for $i < j$ are used.

The mathematical formulation of TOP is:

$$z = \max \sum_{i=2}^{N-1} \sum_{d=1}^{M} s_i y_{id} \tag{14.1}$$

subject to

$$\sum_{j=2}^{N} \sum_{d=1}^{M} x_{1jd} = \sum_{i=1}^{N-1} \sum_{d=1}^{M} x_{iNd} = M \tag{14.2}$$

$$\sum_{i<j} x_{ijd} + \sum_{i>j} x_{jid} = 2y_{jd}, \forall j = 2, \ldots, N-1; \forall d = 1, \ldots, M \tag{14.3}$$

$$\sum_{d=1}^{m} y_{id} \leq 1, \forall i = 2, \ldots, N-1 \tag{14.4}$$

$$\sum_{i=1}^{N-1} \sum_{j>i} t_{ij} x_{ijd} \leq Tmax, \forall d = 1, \ldots, M \tag{14.5}$$

$$\sum_{(i,j) \in U, i<j} x_{ijd} \leq |U| - 1, \ \forall U \subset V \backslash \{1, N\}; 2 \leq |U| \leq N - 2; \forall d = 1, \ldots, M \tag{14.6}$$

$$x_{ijd} \in \{0, 1\}, 1 \leq i < j \leq N; d = 1, \ldots, M \tag{14.7}$$

$$y_{1d} = y_{Nd} = 1, y_{jd} \in \{0, 1\}; \forall i = 2, \ldots, N - 1; \forall d = 1, \ldots, M. \tag{14.8}$$

The objective function (14.1) ensures the maximization of the score collected from the nodes in every route. The constraints (14.2) guarantee that each vehicle route starts at the first node in V set and terminates at the N node. As the vehicle route is created, the connectivity for every included node is protected by constraints (14.3). Constraint (14.4) ensures that every vehicle route is allowed to visit every node at most once and constraints (14.5) establish the total time limit. Constraint (14.6) prohibits the sub-routes. Constraints (14.7) and (14.8) set the integral requirement on each variable.

14.3 A Memetic Algorithm for the Team Orienteering Problem

In this section, we present the proposed memetic algorithm that was used to solve the TOP. The following subsections provide a short introduction to memetic algorithms and analyse the basic stages of the algorithm. Furthermore, the algorithm is presented by pseudo-code, see Algorithm 1. A combination of a Genetic Algorithm (GA) with a local search heuristic (LA) is commonly known as a Memetic Algorithm (MA), firstly introduced by Moscato [16]. A memetic algorithm is based upon the production of a *population* of solutions, the algorithm maintains stable population's size through a maximum number of iterations during the search procedure and each solution into the population is labelled as an *individual*. Initially the individuals are termed "parents", at each iteration new solutions, the "offspring", are constructed by several genetic operators, such as crossover and mutation. MA offers not only a wide spread exploration in the solution area but moreover it focuses on fixed points of the solution area, exploiting best solutions [11, 17]. Due to these characteristics, memetic algorithms succeed in solving effectively the majority of NP-hard optimization problems, such as the Team Orienteering Problem. The cornerstone feature of the presented memetic algorithm, the MSO-TOP, is the crossover operator which is referred as the **similarity operator**, which is our main contribution.

Algorithm 1: Memetic algorithm

```
    Data: $K$ : number of iterations, $E$ : number of solutions (Sect. 14.3.1)
1   Construction of parent population $P$ by using Algorithm 2, the Nearest Insertion
      Algorithm (Sect. 14.3.2)
2   if $Non \neq 0$ then
3       Improve $P$ :
4       for $Non$ nodes do
5           for Each node do            /* seeks suitable position in P */
6               $S_{node} = t_{inode} + t_{nodej} - t_{ij}$
7               Total time($e_0$) $\leftarrow$ Total time($e_0$) + $S_{node}$
8               if Total time($e_0$) $\leq Tmax$ then
9                   Total time($e_0$) $\leftarrow$ Total time($e_0$) + $S_{node}$
10                  Score($e_0$) $\leftarrow$ Score($e_0$) + $s_{node}$
11                  Update $P$
12              end if
13          end for
14      end for
15      Remove $node$
16  end if
17  for $K$ iterations do
18      Similarity Operation for creating offspring population $O$ (Sect. 14.3.3) for $E$
          solutions do
19          Initiate similarity parameter $c$
20          Select elite routes $E$
21          Considering $c$ select $M - 1$ routes
22          Combine $M$ routes & delete common nodes
23      end for
24      Improve $O$ by Mutation (Sect. 14.3.4) Begin Algorithm 3, Local Search (Sect. 14.3.5)
          Select the elites from $P$ and $O$ (Sect. 14.3.6)
25      Create new $P$ population
26  end for
```

14.3.1 Data and Parameters

For the solution of the Team Orienteering Problem a set of nodes $V = \{1, 2, \ldots, N\}$ is used. The total number of routes M, the total feasible $Tmax$, the score of each node s_i and the travel time from each node i to another node j, t_{ij}, are taken as input parameters from the benchmark instances. For the calculation of the distance t_{ij} we used the Euclidean distance. It should be noted that the travel time t_{ij} is assumed to be the transition cost from node i to node j. Moreover, all nodes N are corresponded to an exact score s_i. Additionally we give two more parameters which are essential for the algorithm, the first one is the number of the solutions E (the size of population) which will be produced and the second one is the maximum number of iterations K in which the algorithm will be executed.

14.3.2 Parent Creation and Configuration

Initially, the algorithm focuses on the construction of the parent population. This procedure is divided into two steps, the first step includes the implementation of the heuristic algorithm Nearest Insertion (NI) [19], in this part we initialize the parents by inserting nodes in their paths. In the second step, the algorithm forces the not inserted nodes from the first step to enter into a route taken into account the total time limit. In particular it configures both the feasibility and the fullness of the solutions. Both steps are analysed with details in the next paragraphs.

The first step begins with the construction of the initial solution, see Algorithm 2. Its structure depends on the number M of total routes. Each route begins with the node 1 and ends with the node N. Before the Nearest Insertion of nodes process, each route of the initial solution must contain at least three nodes (the first, one randomly picked and the last). Then, the algorithm enters the phase of node insertion. During the process the remaining nodes are checked. For each one, the algorithm seeks the route and in which position in the selected route to insert the candidate node. It calculates the total time of the route. If the total time does not exceed the total feasible time T_{max}, then the candidate node enters the appropriate route and it increases its total time and the score of the current solution. Otherwise, the candidate node cannot be inserted and the algorithm saves the node as not inserted (denoted by Non) looking forward to use it in the second step. The node insertion algorithm continues until all the remaining unselected nodes have been checked. In the end of the process the parent population will be constructed and each parent consists of M routes. All the solutions will be feasible according to the constraints of the problem.

In the second step, an effort is made to improve the initial population P, when the set of Non nodes is not empty, see Algorithm 1 (lines 3–17). The algorithm utilizes the non inserted nodes (Non) and forces them to enter a route considering not to violate the constraints. It is possible that some of the routes constructed in the first step can have enough total time until T_{max} in order to store one or more nodes. Therefore, with this process we can improve even more the fitness value of each parent. In this phase, the algorithm performs a specific number of iterations, until all the not included nodes are checked. During the iterations, it splits each solution according to the involving M routes. We seek the candidate node which satisfies all the constraints. The node is inserted in the appropriate route and the algorithm increases the route's length, the total time and the total score. The process ends when all nodes have been checked. In the end, the M routes are united constructing the parent solution. In case that the set Non is empty, the process described above is omitted, since all the available nodes have been inserted into the initial solution.

Algorithm 2: Nearest insertion

```
   Data: $V$ number of nodes, $M$ number of routes, $T_{max}$ : maximum allowed time
1  Select initial nodes from $V$
2  Generate $e_0$ initial solution
3  Remaining nodes : Remove initial nodes from $V$
4  for Remaining nodes do
5      for Each node do                /* seeks suitable position in $e_0$ */
6          $S_{node} = t_{inode} + t_{nodej} - t_{ij}$
7          Total time($e_0$) $\leftarrow$ Total time($e_0$) + $S_{node}$
8          if Total time($e_0$) $\leq Tmax$ then
9              Total time($e_0$) $\leftarrow$ Total time($e_0$) + $S_{node}$
10             Score($e_0$) $\leftarrow$ Score($e_0$) + $s_{node}$
11             Update $e_0$
12         else
13             Save $node$ in $Non$
14         end if
15     end for
16     Remove $node$
17 end for
```

14.3.3 Similarity Operator

As mentioned in the previous sections, genetic algorithms are based on natural procedures such as crossover and mutation. These are widely known operations due to their efficiency in succeeding on many scientific fields. In our memetic algorithm we differentiate the classic genetic part. Instead of using the crossover operator we decided to follow another technique for offspring production. The algorithm constructs the solutions by selecting the routes with the best total scores and combining them with other routes which have common nodes. The route combination is defined by a parameter c which designates how many similar nodes are included in each route. We introduce this parameter as a "similarity operator", which differs from the crossover operator, see Algorithm 1 (lines 19–25). This idea was inspired from the column generation technique [9]. Since each solution in our problem is divided into M different routes, we gradually create the offspring from the parent-routes. In the first step the offspring has only one route and afterwards, a second route is combined to it in an effort to solve a small manageable part of the problem (create a feasible solution from a subset of all available routes). As the process is repeated, the partial solution is expanded and, finally, it converges to a complete solution, the final offspring.

Table 14.1 Data for the example

	V set of nodes									
Node	1	2	3	4	5	6	7	8	9	10
$Score\ node$	0	10	20	10	15	25	10	15	25	0

Table 14.2 Parent solutions ($M = 2$)

	Feasible parent tours		$Score\ route\ 1$	$Score\ route\ 2$	$Total\ score$	Unvisited nodes
P1	1 3 4 6 7 10	1 2 5 9 10 –	65	50	115	[8]
P2	1 6 4 7 10	1 3 5 8 9 10 –	45	75	115	[2]
P3	1 2 5 10	1 4 3 9 8 10 – –	25	70	95	[6,7]

The proposed memetic algorithm applies this parameter at the start of the creation of offspring population. This parameter has two different utilities. It not only designates which routes will be selected but it, also, gives how many elements will be removed during the similarity operation considering the unique insertion of a node into a solution. The range of the parameter's values can be modified in order to produce alternative results. At the end of the previous phase, the algorithm saves all routes separately in order to define the common nodes among them. During this procedure we select from the current solutions' set the best score routes. The aim is to compare them with the remaining ones from the solutions' set which contain common nodes with them, taking into account the parameters' values. The selection of the best is purposeful because the algorithm forces the common nodes to be removed based on the similarity parameter from routes with lower score. After removing the similar nodes, we calculate the new length, score and total time of the selected routes. Therefore, the best routes have not been changed and the selected have partially been changed. At the end of the process, the algorithm unifies all of them and constructs new solutions in each iteration until the creation of a new population. This phase is a key point for the memetic algorithm because the variety of the parameter values can affect differently the efficiency of each solution.

In order to visualize and clarify the similarity operator we present an example next. The example's data are given in Table 14.1, i.e. the set of nodes $V = \{1, \ldots, 10\}$ and the $Score$ associated with each node. The algorithm has to use a number of the available nodes in order to form feasible solutions (tours). Each tour consists of sub-tours (routes). For example, if $M = 2$, a feasible solution has two routes, both beginning from the first node (1) and ending to the last one (10), each route has a $Score\ route$, which is the total score of the included nodes in this route and a $Total\ Score$, as the sum of all the routes' score. In favour of the example, all the constructed routes are feasible and satisfy the constraints. Table 14.2 shows an initial population of solutions, $E = 3$, along with their score values.

In the beginning of the process all the solutions are separated and $M * E = 6$ routes are sorted according to their $Score$ values, see Table 14.3. Afterwards, a table with dimensions $M * M$ is formed containing the quantity of the common

Table 14.3 Routes classification

Routes sorted by *Score*							
(1)	1	3	5	8	9	10	75
(2)	1	4	3	9	8	10	70
(3)	1	3	4	6	7	10	65
(4)	1	2	5	9	10	–	50
(5)	1	6	4	7	10	–	45
(6)	1	2	5	10	–	–	25

Table 14.4 Parameter c values

Number of common nodes						
	1	2	3	4	5	6
1	–	3	1	2	0	1
2	3	–	2	1	1	0
3	1	2	–	0	3	0
4	2	1	0	–	0	2
5	0	1	3	0	–	0
6	1	0	0	2	0	–

Table 14.5 Offspring solutions ($M = 2$)

	Feasible offspring tours		*Score route* 1	*Score route* 2	*Total score*	Unvisited Unvisited
O1	1 3 5 8 9 10	1 2 ~~5~~ 10 – –	75	10	85	[4,7]
O2	1 4 3 9 8 10	1 6 ~~4~~ 7 10 –	70	35	105	[2,5]
O3	1 3 4 6 7 10	1 ~~3~~ 5 8 9 10	65	45	110	[2]

nodes among the routes, see Table 14.4. These values are used in order to determine the parameter c. Thus, as the maximum value of common nodes from Table 14.4 is equal to 3, c will randomly selected in the interval [0, 3], for this example: $c = 1$.

In this stage of the process, the offspring solutions have to be constructed. Table 14.5 shows the offspring solutions generated by the similarity operator. The first E routes in Table 14.3 are paired with other ones, in order that the number of the routes in a solution to be equal to M. In this example, the combinations are: (1,6), (2,5), (3,1). It should be noted that if a route is part of a solution, it is not prohibited to be used in an other solution. Since $c = 1$, the paired routes have one common node which have to be removed from one of the routes in such way that the solution to be feasible. The common node will be removed from the route with the lowest *Score*. Also its *Score node* will be abstracted from the *Score route*.

The result of this process is the combination of good solution with consequence of the reduction of the *Score route* value with two possible outcomes. The first possible outcome can be shown observing the solution **O3**, where the *Total Score* value increases although the decrease of the second route's *Score* value, after the *Score* of the node number 3 is removed. The second outcome of the operator will be given after offspring's mutation. After the implementation of the similarity operator one or more nodes may be removed and along with their *Score*, the Total time in

Table 14.6 Improved offspring solutions ($M = 2$)

	Improved offspring tours		*Score route* 1	*Score route* 2	*Total score*	Unvisited nodes
O1	1 3 5 8 9 10	1 **7** **4** 2 10 –	75	30	105	–
O2	1 4 3 9 8 10	1 6 **5** 7 10 –	70	50	105	[2]
O3	1 3 4 6 7 10	1 5 8 **2** 9 10	65	65	130	–

a route will be also reduced. Thus, the gap between the Total time and the $Tmax$ of the changed route will be increased and it is possible for some of the unused nodes to be included in a feasible offspring solution, as Table 14.6 shows (the bold numbers are included into the solution after the similarity operator). As an example, the $Score$ of the solution **O1** is temporary reduced after the similarity operator and increased again in the mutation process.

14.3.4 Offspring Mutation

After the end of the similarity operation, the offspring population is formulated. However, the total score of each solution may decrease dramatically due to the deduction of similar nodes from the routes. Therefore, we implement the mutation in order to improve each solution. In this stage, the algorithm aims to modify the offspring completely by adding new nodes to the paths increasing their length and total score without violating the constraints of the problem. Furthermore, it utilizes the heuristic algorithm of Nearest Insertion such as in the parent configuration phase. Then, we check which nodes are contained in the solution. Next, the algorithm executes a sequence of iterations forcing the not included nodes to enter into the routes. At the end of the iterations, the mutation process is completed and the offspring population is reformed.

14.3.5 Local Search

At this point we proceed with the local search algorithm to intensify the solution. Generally, heuristics are implemented until they achieve a local best. Moreover, they offer a variety of methods which can be used separately or in combinations. In our approach, the algorithm utilizes the method $Replace$ in order to increase the total score of the solutions, see Algorithm 3. The $Replace$ process finds nodes with low score, removes them and inserts in their position non selected nodes with better score. During the operation it is confirmed which nodes are not included in the solution defining them as Non. Next, the algorithm compares the score of every Non node with the score of the current solution nodes. If the Non node has greater score from the one compared with it and the constraints are not violated, then it is

Algorithm 3: Local search

```
Data: $O$ offspring population, $Non$ set of non inserted nodes, $T_{max}$ : maximum allowed
      time
1  Update $Non$
2  for $each\ individual\ in\ O : o$ do
3      for $each\ node\ in\ Non$ do
4          if $s_{node} > s_{o_{node}}$ then
5              Find position in $o$ to insert $node$
6              Calculate Total time($route$)
7              if $Total\ time(route) \leq T_{max}$ then
8                  Update $o$
9                  break
10             end if
11         end if
12     end for
13     Remove $node$ from $Non$
14 end for
```

efficient to take over its position and the current solution is updated. In case that a Non node is not able to replace any of the solution's nodes, the current node is omitted and an another Non node takes its place continuing the $Replace$ process. At the end of the local search, the offspring population will be updated.

14.3.6 Elimination Process and Population Feedback

The algorithm will execute K iterations. In each iteration, a set of solutions must be given in order to produce a new set of solutions. At the end of each iteration, the algorithm creates a new offspring population from the given parent population. Both of them contain E feasible solutions with M valid routes. As mentioned previously, from both populations, each solution has a score which defines its quality. During the selection of the elite individuals we seek the E most profitable solutions among them in order to create a new finest set of solutions. Moreover, the remaining solutions are eliminated from the process. In the end, the new set of solutions will contain the elite individuals from both populations and will be the parent population for the next iteration.

14.4 Computational Results

In order to evaluate the quality of the solutions of our algorithm we use the benchmark instances for the Team Orienteering Problem that have been proposed by Chao et al. [5]. In Table 14.7, the characteristics of these problems are presented. The amount of the available TOP benchmark instances is 387 and they are divided

Table 14.7 Benchmark instances

Set	Problem set	# Problems	# Nodes (N)	# Routes (M)	$Tmax$
Set_1	p1.2	18	32	2	2.5–42.5
	p1.3	18	32	3	1.7–28.3
	p1.4	18	32	4	1.2–21.2
Set_2	p2.2	11	21	2	7.5–22.5
	p2.3	11	21	3	5–15
	p2.4	11	21	4	3.8–11.2
Set_3	p3.2	20	33	2	7.5–55
	p3.3	20	33	3	5–36.7
	p3.4	20	33	4	3.8–27.5
Set_4	p4.2	20	100	2	25–120
	p4.3	20	100	3	16.7–80
	p4.4	20	100	4	12.5–60
Set_5	p5.2	26	66	2	2.5–65
	p5.3	26	66	3	1.7–43.3
	p5.4	26	66	4	1.2–32.5
Set_6	p6.2	14	66	2	7.5–40
	p6.3	14	66	3	5–26.7
	p6.4	14	66	4	3.8–20
Set_7	p7.2	20	102	2	10–200
	p7.3	20	102	3	6.7–133.3
	p7.4	20	102	4	5–100

into seven problem sets. Each set includes three sub-problem sets, their names appear in the second column, followed by the number of their problems. In the third column, the number of nodes (N) is presented. Column 4 shows the number of routes (M). In Column 5 the range of T_{max} is given. All these benchmark instances are available via: www.mech.kuleuven.be/cib/op.

We developed our algorithm in Matlab environment using version R2013.a. The experiments were conducted using an Intel Core i7 2670QM@202 GHz processor with a 6 GB RAM, on Windows 10 Pro 64-bit operation system. For each instance the algorithm was executed 30 times, the size of the population that the algorithm generates (E) is up to 40 individuals and the similarity parameter (c) is randomly selected from the range [0,12]. Our memetic approach for the TOP is denoted by MSO-TOP (Memetic algorithm with Similarity Operator). The following Tables 14.8, 14.9, 14.10, 14.11, 14.12, 14.13, 14.14, 14.15, 14.16 are organized as mentioned below. Column 1 contains the name of each benchmark problem. For each instance the best known obtained score by PSOiA : the effective PSO-inspired algorithm of Dang et al. [8] is presented in the second column, denoted by $BestScore$. The detailed results of our algorithm are presented in the subsequent five columns, giving per instance, the minimum obtained score (z_{min}), the maximum reached score (z_{best}) with respect to the parameter selection,

Table 14.8 MSO-TOP results for Set_1

$N = 32$	$M \in \{2, 3, 4\}$								
		MSO-TOP					MA		
Instances	*Bestscore*	z_{min}	z_{best}	z_{avg}	w (%)	σ	z_{min}	z_{best}	z_{avg}
p1.2.c	20	20	20	20.00	0.00	0.00	20	20	20
p1.2.d	30	30	30	30.00	0.00	0.00	30	30	30
p1.2.e	45	45	45	45.00	0.00	0.00	45	45	45
p1.2.f	80	80	80	80.00	0.00	0.00	80	80	80
p1.2.g	90	85	90	86.83	0.00	2.41	90	90	90
p1.2.h	110	100	110	106.00	0.00	3.27	110	110	110
p1.2.i	135	130	135	130.83	0.00	1.86	135	135	135
p1.2.j	155	130	155	151.33	0.00	6.57	155	155	155
p1.2.k	175	160	175	169.33	0.00	4.42	175	175	175
p1.2.l	195	180	190	189.33	2.56	2.13	195	195	195
p1.2.m	215	200	215	206.67	0.00	4.89	215	215	215
p1.2.n	235	215	235	223.83	0.00	7.49	235	235	235
p1.2.o	240	220	240	228.83	0.00	5.11	240	240	240
p1.2.p	250	225	245	239.00	2.00	3.96	250	250	250
p1.2.q	265	250	265	258.00	0.00	4.40	265	265	265
p1.2.r	280	265	280	272.17	0.00	4.22	280	280	280
p1.3.e	30	30	30	30.00	0.00	0.00	30	30	30
p1.3.f	40	40	40	40.00	0.00	0.00	40	40	40
p1.3.g	50	50	50	50.00	0.00	0.00	50	50	50
p1.3.h	70	70	70	70.00	0.00	0.00	–	–	–
p1.3.i	105	95	105	102.67	0.00	4.03	105	105	105
p1.3.j	115	100	110	104.67	4.35	3.40	105	115	115
p1.3.k	135	125	135	131.17	0.00	2.48	135	135	135
p1.3.l	155	135	155	146.17	0.00	5.27	155	155	155
p1.3.m	175	145	175	163.17	0.00	7.58	175	175	175
p1.3.n	190	180	190	185.50	0.00	3.73	190	190	190
p1.3.o	205	190	205	192.50	0.00	3.82	–	–	–
p1.3.p	220	195	220	210.33	0.00	7.30	220	220	220
p1.3.q	230	215	230	222.50	0.00	4.61	230	230	230
p1.3.r	250	240	250	249.17	0.00	2.27	–	–	–
p1.4.f	25	25	25	25.00	0.00	0.00	25	25	25
p1.4.g	35	35	35	35.00	0.00	0.00	35	35	35
p1.4.h	45	45	45	45.00	0.00	0.00	45	45	45
p1.4.i	60	60	60	60.00	0.00	0.00	60	60	60
p1.4.j	75	75	75	75.00	0.00	0.00	75	75	75
p1.4.k	100	95	100	96.00	0.00	2.00	100	100	100
p1.4.l	120	115	120	119.33	0.00	1.70	120	120	120
p1.4.m	130	120	130	127.00	0.00	2.77	130	130	130
p1.4.n	155	135	155	151.17	0.00	5.73	155	155	155
p1.4.o	165	150	165	157.17	0.00	6.91	165	165	165
p1.4.p	175	155	175	166.50	0.00	8.77	175	175	175

(continued)

Table 14.8 (continued)

$N = 32$	$M \in \{2, 3, 4\}$								
		MSO-TOP					MA		
Instances	*Bestscore*	z_{min}	z_{best}	z_{avg}	w (%)	σ	z_{min}	z_{best}	z_{avg}
p1.4.q	190	160	190	176.00	0.00	7.68	190	190	190
p1.4.r	210	180	205	191.83	2.38	7.47	210	210	210

Table 14.9 MSO-TOP results for Set_2

$N = 21$	$M \in \{2, 3, 4\}$								
		MSO-TOP					MA		
Instances	*Bestscore*	z_{min}	z_{best}	z_{avg}	w (%)	σ	z_{min}	z_{best}	z_{avg}
p2.2.a	90	90	90	90.00	0.00	0.00	90	90	90
p2.2.b	120	120	120	120.00	0.00	0.00	120	120	120
p2.2.c	140	140	140	140.00	0.00	0.00	140	140	140
p2.2.d	160	160	160	160.00	0.00	0.00	160	160	160
p2.2.e	190	185	190	189.67	0.00	1.25	190	190	190
p2.2.f	200	200	200	200.00	0.00	0.00	200	200	200
p2.2.g	200	200	200	200.00	0.00	0.00	200	200	200
p2.2.h	230	230	230	230.00	0.00	0.00	230	230	230
p2.2.i	230	230	230	230.00	0.00	0.00	230	230	230
p2.2.j	260	260	260	260.00	0.00	0.00	260	260	260
p2.2.k	275	270	275	270.17	0.00	0.90	275	275	275
p2.3.a	70	70	70	70.00	0.00	0.00	70	70	70
p2.3.b	70	70	70	70.00	0.00	0.00	70	70	70
p2.3.c	105	105	105	105.00	0.00	0.00	105	105	105
p2.3.d	105	105	105	105.00	0.00	0.00	105	105	105
p2.3.e	120	120	120	120.00	0.00	0.00	120	120	120
p2.3.f	120	120	120	120.00	0.00	0.00	120	120	120
p2.3.g	145	140	145	144.67	0.00	1.25	145	145	145
p2.3.h	165	165	165	165.00	0.00	0.00	–	–	–
p2.3.i	200	200	200	200.00	0.00	0.00	200	200	200
p2.3.j	200	200	200	200.00	0.00	0.00	200	200	200
p2.3.k	200	200	200	200.00	0.00	0.00	200	200	200
p2.4.a	10	10	10	10.00	0.00	0.00	10	10	10
p2.4.b	70	70	70	70.00	0.00	0.00	70	70	70
p2.4.c	70	70	70	70.00	0.00	0.00	70	70	70
p2.4.d	70	70	70	70.00	0.00	0.00	70	70	70
p2.4.e	70	70	70	70.00	0.00	0.00	70	70	70
p2.4.f	105	105	105	105.00	0.00	0.00	105	105	105
p2.4.g	105	105	105	105.00	0.00	0.00	105	105	105
p2.4.h	120	120	120	120.00	0.00	0.00	120	120	120
p2.4.i	120	120	120	120.00	0.00	0.00	120	120	120
p2.4.j	120	120	120	120.00	0.00	0.00	120	120	120
p2.4.k	180	180	180	180.00	0.00	0.00	180	180	180

Table 14.10 MSO-TOP results for Set_3, where $M \in \{2, 3\}$

$N = 33$	$M \in \{2, 3\}$								
		MSO-TOP					MA		
Instances	$Bestscore$	z_{min}	z_{best}	z_{avg}	w (%)	σ	z_{min}	z_{best}	z_{avg}
p3.2.a	90	90	90	90.00	0	0.00	90	90	90
p3.2.b	150	150	150	150.00	0.00	0.00	150	150	150
p3.2.c	180	180	180	180.00	0.00	0.00	180	180	180
p3.2.d	220	220	220	220.00	0.00	0.00	220	220	220
p3.2.e	260	260	260	260.00	0.00	0.00	260	260	260
p3.2.f	300	300	300	300.00	0.00	0.00	300	300	300
p3.2.g	360	360	360	360.00	0.00	0.00	360	360	360
p3.2.h	410	390	400	390.67	2.44	2.49	410	410	410
p3.2.i	460	430	460	440.67	0.00	9.29	460	460	460
p3.2.j	510	470	510	494.67	0.00	8.84	510	510	510
p3.2.k	550	520	550	540.67	0.00	9.29	550	550	550
p3.2.l	590	560	590	577.67	0.00	8.83	590	590	590
p3.2.m	620	570	620	600.67	0.00	15.26	620	620	620
p3.2.n	660	610	660	634.33	0.00	11.46	660	660	660
p3.2.o	690	660	690	673.67	0.00	7.06	690	690	690
p3.2.p	720	700	720	710.33	0.00	3.14	720	720	720
p3.2.q	760	740	760	749.33	0.00	4.42	760	760	760
p3.2.r	790	780	790	786.67	0.00	4.71	790	790	790
p3.2.s	800	790	800	799.67	0.00	1.80	800	800	800
p3.2.t	800	800	800	800.00	0.00	0.00	800	800	800
p3.3.a	30	30	30	30.00	0	0.00	30	30	30
p3.3.b	90	90	90	90.00	0	0.00	90	90	90
p3.3.c	120	120	120	120.00	0.00	0.00	120	120	120
p3.3.d	170	170	170	170.00	0.00	0.00	170	170	170
p3.3.e	200	200	200	200.00	0.00	0.00	200	200	200
p3.3.g	270	250	270	258.33	0.00	5.22	270	270	270
p3.3.h	300	300	300	300.00	0.00	0.00	300	300	300
p3.3.i	330	320	330	328.33	0.00	3.73	330	330	330
p3.3.j	380	360	380	372.33	0.00	5.59	380	380	380
p3.3.k	440	420	440	425.67	0.00	6.67	440	440	440
p3.3.l	480	430	480	464.33	0.00	13.59	480	480	480
p3.3.m	520	500	520	512.67	0.00	7.72	520	520	520
p3.3.n	570	540	570	553.00	0.00	9.00	570	570	570
p3.3.o	590	550	590	577.67	0.00	6.67	590	590	590
p3.3.p	640	600	640	623.33	0.00	11.06	640	640	640
p3.3.q	680	660	680	673.00	0.00	5.86	680	680	680
p3.3.r	710	680	710	692.33	0.00	12.83	710	710	710
p3.3.s	720	690	710	705.33	1.39	7.18	720	720	720
p3.3.t	760	700	750	719.00	1.32	12.21	760	760	760

Table 14.11 MSO-TOP results for Set_3, where M=4

$N = 33$	$M = 4$								
		MSO-TOP					MA		
Instances	$Bestscore$	z_{min}	z_{best}	z_{avg}	w (%)	σ	z_{min}	z_{best}	z_{avg}
p3.4.c	90	90	90	90.00	0	0.00	90	90	90
p3.4.d	100	100	100	100.00	0.00	0.00	100	100	100
p3.4.e	140	140	140	140.00	0.00	0.00	140	140	140
p3.4.f	190	190	190	190.00	0.00	0.00	190	190	190
p3.4.g	220	220	220	220.00	0.00	0.00	220	220	220
p3.4.h	240	240	240	240.00	0.00	0.00	240	240	240
p3.4.i	270	260	270	264.33	0.00	4.96	270	270	270
p3.4.j	310	290	310	302.33	0.00	4.96	310	310	310
p3.4.k	350	350	350	350.00	0.00	0.00	–	–	–
p3.4.l	380	360	380	377.67	0.00	6.16	380	380	380
p3.4.m	390	380	390	389.67	0.00	1.80	390	390	390
p3.4.n	440	420	440	432.33	0.00	8.03	440	440	440
p3.4.o	500	460	490	478.00	2.00	8.33	500	500	500
p3.4.p	560	500	550	514.67	1.79	10.87	560	560	560
p3.4.q	560	540	560	551.67	0.00	8.98	560	560	560
p3.4.r	600	560	600	580.00	0.00	13.42	600	600	600
p3.4.s	670	660	670	669.33	0.00	2.49	670	670	670
p3.4.t	670	670	670	670.00	0.00	0.00	670	670	670

the average score (z_{avg}) acquired by 30 executions of the algorithm. Also, two performance measurements are given, the first one indicates the quality of the solutions in terms of relative deviation from the best known solution ($Bestscore$), that is $w = \frac{Best\,score - z_{best}}{Best\,score}\,\%$. The second measurement that is presented, denoted by σ, refers to the standard deviation of the results produced by the algorithm and demonstrates the spread of $Score$ values within a set of data, obtained over 30 executions of the algorithm. Afterwards, the details of the solutions of the MA (a memetic algorithm for the TOP [3]) are given in every table, such as the worst, the best and the average obtained $Score$. In these tables, where the symbol "-" appears it means that there are not data available from the MA algorithm.

Table 14.8 presents the solutions for the first set of the instances with total 32 nodes, the algorithm reached the best score in 39 of the 43 problems (90.7%). Table 14.9 shows that the best known score has been reached for all the problems in Set_2. Tables 14.10 and 14.11 present the solutions for the third set with 33 nodes and it is interesting to observe that the percentage of the problems with zero w is 91.2%. Thus, concerning the quality of the solutions, the algorithm behaves in similar way for problems with number of nodes in close range. Set_4 has the greatest number of nodes, $N = 100$. Tables 14.12 and 14.13 show that 40 of the 50 instances have been solved with relative deviation under 4% from the best known score value. About 67.1% of the problems in Set_5 obtained solutions equal to the

Table 14.12 MSO-TOP results for Set_4, where $M \in \{2, 3\}$

$N = 100$	$M \in \{2, 3\}$								
		MSO-TOP					MA		
Instances	*Bestscore*	z_{min}	z_{best}	z_{avg}	w (%)	σ	z_{min}	z_{best}	z_{avg}
p4.2.c	452	452	452	452	0.00	0.00	452	452	452
p4.2.d	531	459	525	498.50	1.13	17.00	531	531	531
p4.2.e	618	535	607	568.47	1.78	18.62	618	618	618
p4.2.f	687	590	669	634.73	2.62	18.83	678	681	687
p4.2.g	757	628	739	706.37	2.38	24.84	756	756.7	757
p4.2.h	835	707	808	764.40	3.23	25.49	810	822	835
p4.2.i	918	918	918	918.00	0.00	0.00	918	918	918
p4.2.j	965	780	931	870.43	3.52	30.15	962	963.3	964
p4.2.k	1022	887	990	942.83	3.13	25.08	1013	1016	1022
p4.2.l	1074	959	1032	992.13	3.91	19.09	1069	1070.3	1071
p4.2.m	1132	991	1085	1039.93	4.15	21.56	1125	1129.7	1132
p4.2.n	1174	1053	1145	1086.57	2.47	21.28	1170	1172.7	1174
p4.2.o	1218	1090	1192	1129.30	2.13	25.43	1217	1217	1217
p4.2.p	1242	1131	1219	1173.70	1.85	23.62	1237	1240	1242
p4.2.q	1268	1172	1243	1207.80	1.97	17.35	1263	1265	1267
p4.2.r	1292	1208	1251	1234.03	3.17	9.96	1291	1291.3	1292
p4.2.s	1304	1221	1280	1258.27	1.84	11.53	1304	1304	1304
p4.2.t	1306	1264	1299	1283.67	0.54	9.40	1306	1306	1306
p4.3.d	335	303	335	320.33	0.00	8.95	38	38	38
p4.3.e	468	415	459	438.30	1.92	11.11	193	193	193
p4.3.f	579	534	564	542.87	2.59	9.38	579	579	579
p4.3.g	653	577	630	603.77	3.52	15.10	653	653	653
p4.3.h	729	640	705	670.40	3.29	14.88	717	722.3	725
p4.3.i	809	683	767	718.67	5.19	19.83	807	808.3	809
p4.3.j	861	750	839	777.83	2.56	22.18	856	859	861
p4.3.k	919	766	869	829.70	5.44	21.67	919	919	919
p4.3.l	979	845	935	884.50	4.49	21.96	972	972.7	974
p4.3.m	1063	903	1004	958.73	5.55	21.23	1039	1049.3	1063
p4.3.n	1121	964	1090	1018.40	2.77	22.79	1114	1118.7	1121
p4.3.o	1172	1037	1126	1073.23	3.92	17.68	1167	1169.7	1172
p4.3.p	1222	1103	1188	1139.40	2.78	20.00	1205	1216	1222
p4.3.q	1253	1124	1226	1181.27	2.15	25.86	1245	1248.7	1252
p4.3.r	1273	1189	1243	1216.83	2.36	16.37	1269	1270.7	1273
p4.3.s	1295	1203	1282	1245.50	1.00	19.62	1295	1295	1295
p4.3.t	1305	1240	1288	1269.73	1.30	14.45	1304	1304	1304

best, as Tables 14.14 and 14.15 show, but it has to be mentioned that about 93.2% of the solved instances correspond to less than 2% relative deviation. Only 13 of the 24 instances that have been tested from Set_6 reached the best known solution, although the maximum relative deviation does not surpass the 2.56% and 21 of the

Table 14.13 MSO-TOP results for Set_4, where M=4

$N = 100$	$M = 4$								
		MSO-TOP					MA		
Instances	$Bestscore$	z_{min}	z_{best}	z_{avg}	w (%)	σ	z_{min}	z_{best}	z_{avg}
p4.4.e	183	182	182	182.00	0.55	0.00	183	183	183
p4.4.f	324	310	324	313.27	0.00	3.75	324	324	324
p4.4.g	461	418	446	440.73	3.25	6.41	461	461	461
p4.4.h	571	507	553	524.90	3.15	12.60	571	571	571
p4.4.i	657	559	626	584.10	4.72	18.40	657	657	657
p4.4.j	732	689	727	704.03	0.68	10.10	732	732	732
p4.4.k	821	716	807	757.73	1.71	25.29	821	821	821
p4.4.l	880	751	848	800.13	3.64	23.61	879	879	879
p4.4.m	919	759	878	818.80	4.46	26.58	916	916.7	918
p4.4.n	977	811	923	865.47	5.53	28.04	962	964.7	969
p4.4.o	1061	887	997	935.50	6.03	27.71	1051	1057.7	1061
p4.4.p	1124	959	1075	1016.73	4.36	30.37	1110	1114.7	1124
p4.4.q	1161	1002	1114	1050.57	4.05	25.42	1161	1161	1161
p4.4.r	1216	1097	1170	1131.00	3.78	14.33	1207	1210	1216
p4.4.s	1260	1136	1222	1170.60	3.02	16.44	1255	1257.7	1259
p4.4.t	1285	1188	1242	1216.63	3.35	14.70	1281	1282.3	1284

24 correspond to w less than 1%, see Table 14.16. In general, the relative deviation w is used to evaluate the performance of MSO-TOP and the average best obtained solution z_{avg}, in combination with the standard deviation, exhibits the stability of our algorithm.

In Figs. 14.1, 14.2, 14.3, 14.4, 14.5, 14.6, we present analytically the solutions for the instances p1.2.r, p2.3.e, p3.4.s, p4.4.e, p5.4.u, p6.3.n, respectively, which are optimal or near optimal. Table 14.17 presents the average w for all the subsets of the benchmark instances that have been tested. Observing Table 14.17, we conclude that the MSO-TOP algorithm is able to produce high quality solutions. In addition, three instances from each of the datasets are presented (one for every available M value) in Fig. 14.7. Figure 14.7 demonstrates the performance of the algorithm in terms of relative deviation from the best known solution, over 30 executions in a sample of instances. Observing that figure, the algorithm seems to be robust concerning the 18 selected instances, those are chosen to be illustrated since they are representative enough. A final conclusion, extracted from all the above tables, is that MSO-TOP behaves with two different ways, in the first way the algorithm will reach the best known solutions at the first iteration and will remain fixed to this value, i.e. $w = 0.00\%$ and $\sigma = 0.00$. In the second way, the algorithm will reach a value close to the best known solution but it is not able to surpass this local optimum solution. In order to overpass this situation the size of the population has to be increased.

Table 14.14 MSO-TOP results for Set_5, where $M \in \{2, 3\}$

$N = 66$	$M \in \{2, 3\}$								
		MSO-TOP					MA		
Instances	$Bestscore$	z_{min}	z_{best}	z_{avg}	w (%)	σ	z_{min}	z_{best}	z_{avg}
p5.2.b	20	20	20	20.00	0.00	0.00	20	20	20
p5.2.c	50	50	50	50.00	0.00	0.00	50	50	50
p5.2.d	80	80	80	80.00	0.00	0.00	80	80	80
p5.2.e	180	155	180	170.67	0.00	7.61	180	180	180
p5.2.f	240	215	240	226.67	0.00	12.47	240	240	240
p5.2.g	320	310	320	318.33	0.00	2.69	320	320	320
p5.2.h	410	385	410	397.17	0.00	9.72	410	410	410
p5.2.i	480	440	480	459.33	0.00	7.61	480	480	480
p5.2.j	580	545	580	565.17	0.00	11.72	580	580	580
p5.2.k	670	640	670	660.67	0.00	8.63	670	670	670
p5.2.l	800	755	800	776.67	0.00	14.62	800	800	800
p5.2.m	860	830	860	851.00	0.00	12.34	860	860	860
p5.2.n	925	840	925	888.17	0.00	21.23	925	925	925
p5.2.o	1020	950	1010	992.33	0.98	14.48	1020	1020	1020
p5.2.p	1150	1055	1150	1095.33	0.00	28.55	1150	1150	1150
p5.2.q	1195	1175	1195	1181.83	0.00	6.89	1195	1195	1195
p5.2.r	1260	1230	1260	1246.33	0.00	8.46	1260	1260	1260
p5.2.s	1340	1275	1315	1301.83	1.87	12.14	1325	1326.7	1330
p5.2.t	1400	1325	1380	1352.17	1.43	18.24	1390	1396.7	1400
p5.2.u	1460	1400	1440	1418.33	1.37	13.86	1455	1458.3	1460
p5.2.v	1505	1435	1500	1460.17	0.33	14.58	1500	1501.7	1505
p5.2.w	1565	1490	1555	1519.67	0.64	17.17	1560	1560	1560
p5.2.x	1610	1525	1600	1570.83	0.62	17.18	1610	1610	1610
p5.2.y	1645	1595	1640	1618.50	0.30	10.89	1645	1645	1645
p5.2.z	1680	1620	1675	1659.67	0.30	11.61	1680	1680	1680
p5.3.c	20	20	20	20.00	0.00	0.00	20	20	20
p5.3.d	60	60	60	60.00	0.00	0.00	60	60	60
p5.3.e	95	80	95	86.00	0.00	7.35	–	–	–
p5.3.f	110	110	110	110.00	0.00	0.00	110	110	110
p5.3.g	185	150	185	159.67	0.00	9.39	185	185	185
p5.3.h	260	225	260	238.00	0.00	8.12	260	260	260
p5.3.i	335	280	335	324.00	0.00	15.41	335	335	335
p5.3.j	470	400	470	424.17	0.00	21.99	470	470	470
p5.3.k	495	430	495	454.17	0.00	17.23	495	495	495
p5.3.l	595	520	595	566.33	0.00	21.17	595	595	595
p5.3.m	650	600	650	608.67	0.00	8.36	650	650	650
p5.3.n	755	700	755	728.67	0.00	16.17	755	755	755
p5.3.o	870	815	870	834.17	0.00	14.72	870	870	870
p5.3.p	990	900	990	984.00	0.00	19.21	990	990	990

(continued)

Table 14.14 (continued)

$N = 66$	$M \in \{2, 3\}$								
		MSO-TOP					MA		
Instances	*Bestscore*	z_{min}	z_{best}	z_{avg}	w (%)	σ	z_{min}	z_{best}	z_{avg}
p5.3.q	1070	995	1065	1056.83	0.47	17.77	1070	1070	1070
p5.3.r	1125	1090	1125	1114.50	0.00	7.99	1125	1125	1125
p5.3.s	1190	1115	1185	1153.67	0.42	15.60	1190	1190	1190
p5.3.t	1260	1155	1260	1211.17	0.00	26.07	1260	1260	1260
p5.3.u	1345	1275	1340	1306.50	0.37	20.09	1345	1345	1345
p5.3.v	1425	1360	1425	1383.33	0.00	17.00	1425	1425	1425
p5.3.w	1485	1395	1460	1430.00	1.68	17.03	1485	1485	1485
p5.3.x	1555	1455	1535	1500.67	1.29	21.71	1530	1545	1555
p5.3.y	1595	1515	1590	1544.67	0.31	21.91	1590	1590	1590
p5.3.z	1635	1555	1635	1581.50	0.00	18.89	1635	1635	1635

Table 14.15 MSO-TOP results for Set_5, where M=4

$N = 66$	$M = 4$								
		MSO-TOP					MA		
Instances	*Bestscore*	z_{min}	z_{best}	z_{avg}	w (%)	σ	z_{min}	z_{best}	z_{avg}
p5.4.c	20	20	20	20.00	0.00	0.00	20	20	20
p5.4.d	20	20	20	20.00	0.00	0.00	20	20	20
p5.4.e	20	20	20	20.00	0.00	0.00	20	20	20
p5.4.f	80	80	80	80.00	0.00	0.00	80	80	80
p5.4.g	140	125	140	133.13	0.00	6.88	140	140	140
p5.4.h	140	140	140	140.00	0.00	0.00	140	140	140
p5.4.i	240	210	240	223.00	0.00	14.87	240	240	240
p5.4.j	340	280	340	309.33	0.00	14.87	340	340	340
p5.4.k	340	320	340	328.67	0.00	7.63	340	340	340
p5.4.l	430	395	420	402.67	2.33	7.04	430	430	430
p5.4.m	555	465	550	501.83	0.90	23.08	555	555	555
p5.4.n	620	505	620	544.00	0.00	25.44	620	620	620
p5.4.o	690	600	690	636.50	0.00	27.27	690	690	690
p5.4.p	765	670	760	701.33	0.65	17.89	760	760	760
p5.4.q	860	780	860	800.83	0.00	28.11	860	860	860
p5.4.r	960	830	890	870.67	7.29	14.53	960	960	960
p5.4.s	1030	935	1010	971.83	1.94	26.60	1025	1026.7	1030
p5.4.t	1160	1070	1160	1130.50	0.00	26.97	1160	1160	1160
p5.4.u	1300	1300	1300	1300.00	0.00	0.00	1300	1300	1300
p5.4.v	1320	1260	1320	1308.67	0.00	16.83	1320	1320	1320
p5.4.w	1390	1315	1370	1348.17	1.44	16.61	1380	1380	1380
p5.4.x	1450	1380	1420	1406.50	2.07	11.91	1440	1445	1450
p5.4.y	1520	1430	1480	1453.67	2.63	15.05	1520	1520	1520
p5.4.z	1620	1445	1550	1503.00	4.32	25.02	1620	1620	1620

Table 14.16 MSO-TOP results for Set_6

$N = 66$	$M \in \{2, 3, 4\}$								
		MSO-TOP					MA		
Instances	*Bestscore*	z_{min}	z_{best}	z_{avg}	w (%)	σ	z_{min}	z_{best}	z_{avg}
p6.2.d	192	192	192	192.00	0.00	0.00	192	192	192
p6.2.e	360	360	360	360.00	0.00	0.00	360	360	360
p6.2.f	588	534	588	586.20	0.00	9.69	588	588	588
p6.2.g	660	636	660	657.60	0.00	6.12	660	660	660
p6.2.h	780	720	774	752.40	0.77	15.84	780	780	780
p6.2.i	888	834	888	864.20	0.00	17.42	888	888	888
p6.2.j	948	900	942	926.80	0.63	13.21	942	944	948
p6.2.k	1032	972	1032	1005.20	0.00	17.85	1032	1032	1032
p6.2.l	1116	1068	1110	1088.00	0.54	10.20	1116	1116	1116
p6.2.m	1188	1140	1170	1157.40	1.52	8.67	1188	1188	1188
p6.2.n	1260	1212	1254	1229.60	0.48	7.89	1254	1258	1260
p6.3.g	282	282	282	282.00	0.00	0.00	282	282	282
p6.3.h	444	408	444	430.20	0.00	13.43	444	444	444
p6.3.i	642	588	642	610.60	0.00	18.70	642	642	642
p6.3.j	828	822	822	822.00	0.72	0.00	828	828	828
p6.3.k	894	804	894	840.20	0.00	25.05	894	894	894
p6.3.l	1002	906	990	935.00	1.20	20.44	1002	1002	1002
p6.3.m	1080	1002	1074	1025.80	0.56	18.16	1080	1080	1080
p6.3.n	1170	1080	1140	1108.60	2.56	14.00	1170	1170	1170
p6.4.j	366	366	366	366.00	0.00	0.00	–	–	–
p6.4.k	528	456	528	473.60	0.00	13.50	–	–	–
p6.4.l	696	624	690	652.80	0.86	16.93	696	696	696
p6.4.m	912	816	912	854.20	0.00	29.41	912	912	912
p6.4.n	1068	1002	1062	1060.00	0.56	10.77	1068	1068	1068

Table 14.17 MSO-TOP w average

Set	Number of routes (M)	$\bar{w}$	Set	Number of routes (M)	$\bar{w}$
Set_1	2	0.29	Set_4	2	2.34
	3	0.31		3	2.99
	4	0.18		4	2.85
Set_2	2	0.00	Set_5	2	0.31
	3	0.00		3	0.19
	4	0.00		4	0.81
Set_3	2	0.12	Set_6	2	0.36
	3	0.14		3	0.63
	4	0.16		4	0.28

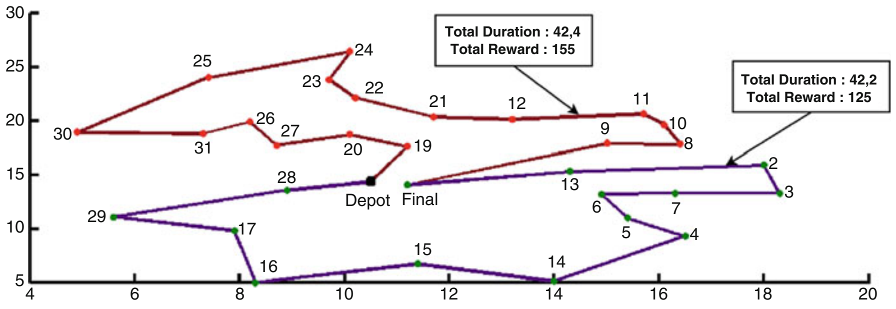


Fig. 14.1 Instance: p1.2.r, Tmax = 42.5, Score = 280

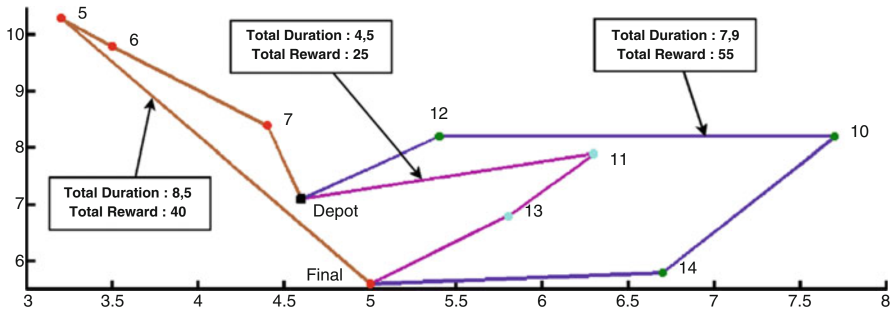


Fig. 14.2 Instance: p2.3.e, Tmax = 9, Score = 120

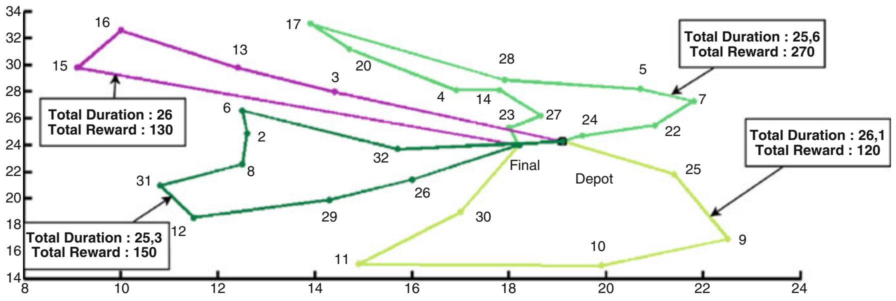


Fig. 14.3 Instance: p3.4.s, Tmax = 26.2, Score = 670

14.5 Conclusions and Future Research

Marinakis et al. [15] presented a Memetic-GRASP algorithm for the solution of the Orienteering Problem (OP), setting the base for the creation of a complete decision support system that will help a tourist to see the most important, based on his preferences, attractions of a city or a museum that he would like to visit during his vacations. We have to make the assumption that as the OP generates only one

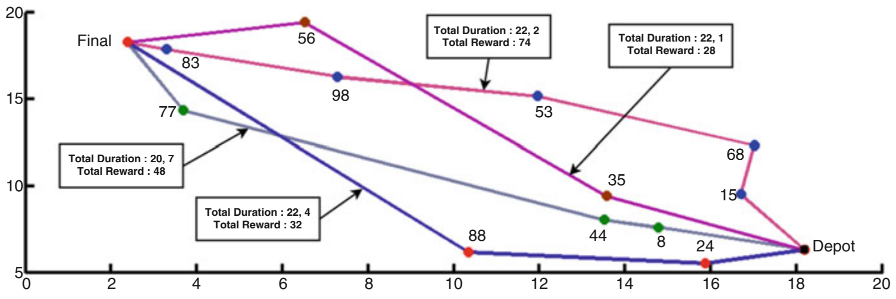


Fig. 14.4 Instance: p4.4.e, Tmax = 22.5, Score = 182

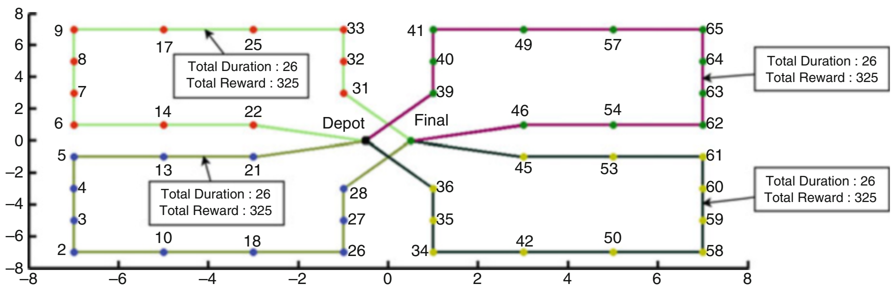


Fig. 14.5 Instance: p5.4.u, Tmax = 26.2, Score = 1300

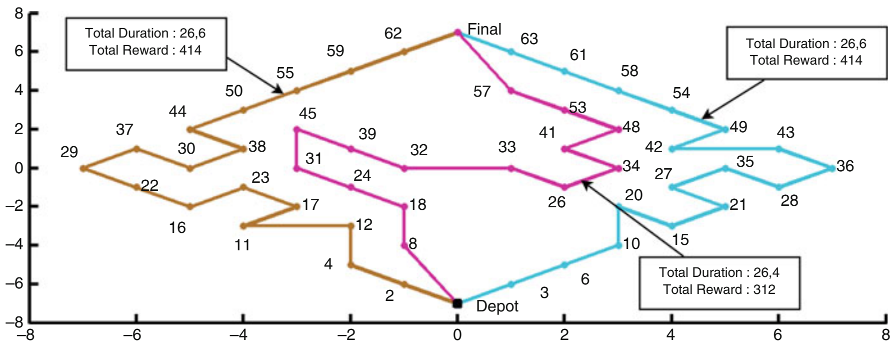


Fig. 14.6 Instance: p6.3.n, Tmax = 26.7, Score = 1140

route, the Memetic-GRASP solution is corresponding only to 1 day of visit, thus, the development of an algorithm for solving the Team Orienteering Problem was a necessity in order to simulate a multiple day visit.

The proposed algorithm in this paper that fulfils the above purpose is denoted as MSO-TOP, a Memetic algorithm with Similarity Operator. The MSO-TOP includes genetic features with the proposed “similarity operator”, combined with local search procedure. The algorithm was tested in classic sets of benchmark instances for the

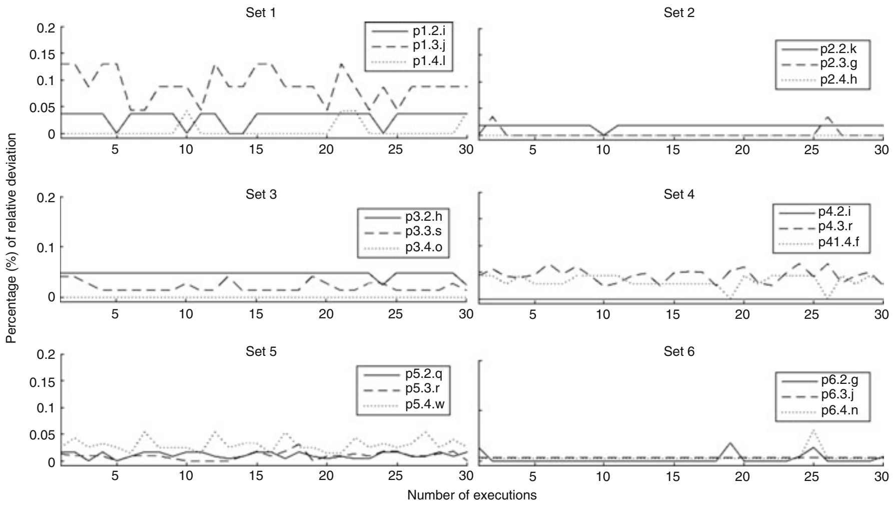


Fig. 14.7 Performance sample over 30 executions in terms of w

Orienteering Problem and in many cases it found the best known solutions. Our future research will be, initially, to change the formulation of the problem and to add a stochastic variable in each node which will correspond to the waiting time. Also, our objective is the development of a decision support system (tourist guide planner) in which the user will add his preferences.

References

1. Archetti, C., Hertz, A., & Speranza, M. G. (2007). Metaheuristics for the team orienteering problem. *Journal of Heuristics*, 13(1), 49–76.
2. Bonnefoy, L. (2010). L'optimisation par essaims particulaires appliquée au team orienteering problem. Preprint available at: http://ludovicbonnefoy.files.wordpress.com/2010/10/majecstic2010.pdf.
3. Bouly, H., Dang, D. C., & Moukrim, A. (2010). A memetic algorithm for the team orienteering problem. *4OR*, 8(1), 49–70.
4. Butt, S. E., & Cavalier, T. M. (1994). A heuristic for the multiple tour maximum collection problem. *Computers & Operations Research*, 21(1), 101–111.
5. Chao, I. M., Golden, B. L., & Wasil, E. A. (1996). The team orienteering problem. *European journal of operational research*, 88(3), 464–474.
6. Chao, I. M., Golden, B. L., & Wasil, E. A. (1996). A fast and effective heuristic for the orienteering problem. *European Journal of Operational Research*, 88(3), 475–489.
7. Dang, D. C., Guibadj, R. N., & Moukrim, A. (2011). A PSO-based memetic algorithm for the team orienteering problem. *Applications of Evolutionary Computation*, Springer Berlin Heidelberg, 471–480.
8. Dang, D. C., Guibadj, R. N., & Moukrim, A. (2013). An effective PSO-inspired algorithm for the team orienteering problem. *European Journal of Operational Research*, 229(2), 332–344.
9. Desrosiers, J., & Lübbecke, M. E. (2005). *A primer in column generation*, Springer US, 1–32.

10. Golden, B. L., Levy, L., & Vohra, R. (1987). The orienteering problem. *Naval research logistics*, 34(3), 307–318.
11. Hart, W. E., Krasnogor, N., & Smith, J. E. (Eds.). (2004). Recent advances in memetic algorithms, *Springer Science & Business Media*, 166.
12. Ke, L., Archetti, C. & Feng, Z. (2008). Ants can solve the team orienteering problem. *Computers & Industrial Engineering*, 54(3), 648–665.
13. Kim, B. I., Li, H., & Johnson, A. L. (2013). An augmented large neighborhood search method for solving the team orienteering problem. *Expert Systems with Applications*, 40(8), 3065–3072.
14. Lin, S. W. (2013). Solving the team orienteering problem using effective multi-start simulated annealing. *Applied Soft Computing*, 13(2), 1064–1073.
15. Marinakis, Y., Politis, M., Marinaki, M., & Matsatsinis, N. (2015). A Memetic-GRASP Algorithm for the Solution of the Orienteering Problem. *Modelling, Computation and Optimization in Information Systems and Management Sciences*, Springer International Publishing, 105–116.
16. Moscato, P. (1989). On evolution, search, optimization, genetic algorithms and martial arts: Towards memetic algorithms. Caltech concurrent computation program, *C3P Report*, 826, 1989.
17. Moscato, P., & Cotta, C. (2003). A gentle introduction to memetic algorithms. *Handbook of metaheuristics*, Springer US, 105–144.
18. Muthuswamy, S., & Lam, S. (2011). Discrete particle swarm optimization for the team orienteering problem. *Memetic Computing*, 3(4), 287–303.
19. Rosenkrantz, D. J., Stearns, R. E., & Lewis, II, P. M. (1977). An analysis of several heuristics for the traveling salesman problem. *SIAM journal on computing*, 6(3), 563–581.
20. Souffriau, W., Vansteenwegen, P., Berghe, G. V., & Van Oudheusden, D. (2010). A path relinking approach for the team orienteering problem.*Computers & Operations Research*, 37(11), 1853–1859.
21. Souffriau, W., Vansteenwegen, P., Vertommen, J., Berghe, G. V.,& Oudheusden, D. V. (2008). A personalized tourist trip design algorithm for mobile tourist guides. *Applied Artificial Intelligence*, 22(10), 964–985.
22. Tang, H., & Miller-Hooks, E. (2005). A tabu search heuristic for the team orienteering problem. *Computers & Operations Research*, 32(6), 1379–1407.
23. Toth, P., & Vigo, D. (Eds.). (2014). *Vehicle routing: problems, methods, and applications* (Vol. 18). Siam.
24. Vansteenwegen, P., Souffriau, W., & Van Oudheusden, D. (2011). The orienteering problem: A survey. *European Journal of Operational Research*, 209(1), 1–10.
25. Vansteenwegen, P., Souffriau, W., Berghe, G. V., & Van Oudheusden, D. (2009). A guided local search metaheuristic for the team orienteering problem. *European Journal of Operational Research*, 196(1), 118–127.
26. Vansteenwegen, P., Souffriau, W., Berghe, G. V., & Van Oudheusden, D. (2009). Metaheuristics for tourist trip planning. *Metaheuristics in the Service Industry*, Springer Berlin Heidelberg, 15–31.
27. Vansteenwegen, P., & Van Oudheusden, D. (2007). The mobile tourist guide: an OR opportunity. *OR Insight*, 20(3), 21–27.
28. Vincent, F. Y., Lin, S. W., & Chou, S. Y. (2010). The museum visitor routing problem. *Applied Mathematics and Computation*, 216(3), 719–729.

Chapter 15
A Memetic Algorithm for Competitive Facility Location Problems

Benjamin Biesinger, Bin Hu, and Günther R. Raidl

Abstract We study a memetic algorithm to solve diverse variants of competitive facility location problems. Two non-cooperating companies enter a market sequentially and compete for market share. The first decision maker, the leader, aims to choose a set of locations which maximize his market share knowing that a follower will enter the same market, lowering the leader's market share. For this bi-level combinatorial optimization problem several customer behaviour scenarios and demand models are studied. A memetic algorithm is applied to find a good set of locations for the leader and the solution evaluation consisting of finding near optimal locations for the follower is performed by greedy algorithms and the use of mixed integer linear programming models. We conclude this chapter with a case study for two hypermarket chains who both want to open stores in Vienna using real world demographic data. In this study we consider six different customer behaviour scenarios and present numerical and graphical results which show the effectiveness of the presented approach.

Keywords Facility location problem · Memetic algorithm · Competitive facility location

15.1 Introduction

We consider the competitive facility location problem (CFLP) where two decision makers, a leader and a follower, compete for market share. In order to satisfy client demands, they have to decide where to open their facilities from a finite

B. Biesinger (✉) · G. R. Raidl
Institute of Computer Graphics and Algorithms, TU Wien, Vienna, Austria
e-mail: Biesinger@ac.tuwien.ac.at; benjamin.biesinger@ait.ac.at; Raidl@ac.tuwien.ac.at

B. Hu
AIT Austrian Institute of Technology, Mobility Department - Dynamic Transportation Systems, Vienna, Austria
e-mail: bin.hu@ait.ac.at

P. Moscato, N. J. de Vries (eds.), *Business and Consumer Analytics: New Ideas*,
https://doi.org/10.1007/978-3-030-06222-4_15

set of possible positions, which correspond to *planning cells*. The leader starts to place all of his facilities, then the follower places his facilities. There are different scenarios which vary in the way customers satisfy their demands from the set of open facilities. This classification is taken from Suárez-Vega et al. [38]:

Customer Behaviour

- Binary: The demand of each customer is fulfilled by the nearest facility only.
- Proportional: Each customer splits his demand over all open facilities proportional to an attractiveness value, which depends on the distances to the facilities.
- Partially binary: This is similar to the proportional behaviour but the demand is split only between the nearest leader and nearest follower facility, again, proportional to an attractiveness value depending on the distance.

Demand Model

- Essential demand: Each customer always satisfies all of his/her demand.
- Unessential demand: A customer does not satisfy all of his/her demand but only a proportion depending on the distance to the serving facility.

Combining the three customer behaviours and the two demand models results in six different scenarios. Since demand corresponds to the buying power of the customers, the market share (or turnover) of the competing firms increases with the amount of demand they fulfil. Therefore, in order to obtain an accurate revenue value for the leader, the subproblem of finding an optimal set of Facility Locations for the follower for a given set of leader locations has to be solved. This makes the problem a Σ_2^P-hard bi-level optimization problem [31]. We model the decision problem of the leader who wants to maximize his market share knowing that a follower will enter the market subsequently under a given customer behaviour scenario.

Based on our previous work [7–9], we apply a Memetic Algorithm on a detailed case study on Vienna that uses real world demographic data and the division of 250 registration districts as planning cells, see Fig. 15.1. In this study we assume that a hypermarket chain (leader) wants to access the Viennese market, with the knowledge that a rival company (follower) will follow afterwards. The goal is to determine the optimal locations for the leader's stores so that its market share is maximal after the follower enters the market.

This chapter is structured as follows. In Sect. 15.2 we provide a short survey on location problems in the literature. Section 15.3 formalizes the considered problem and the different behaviour scenarios. In Sect. 15.4 we present the memetic algorithm and the mathematical models that are used to evaluate the solutions. Section 15.5 describes the case study and discusses the data and computational results. Finally we draw conclusions in Sect. 15.6.

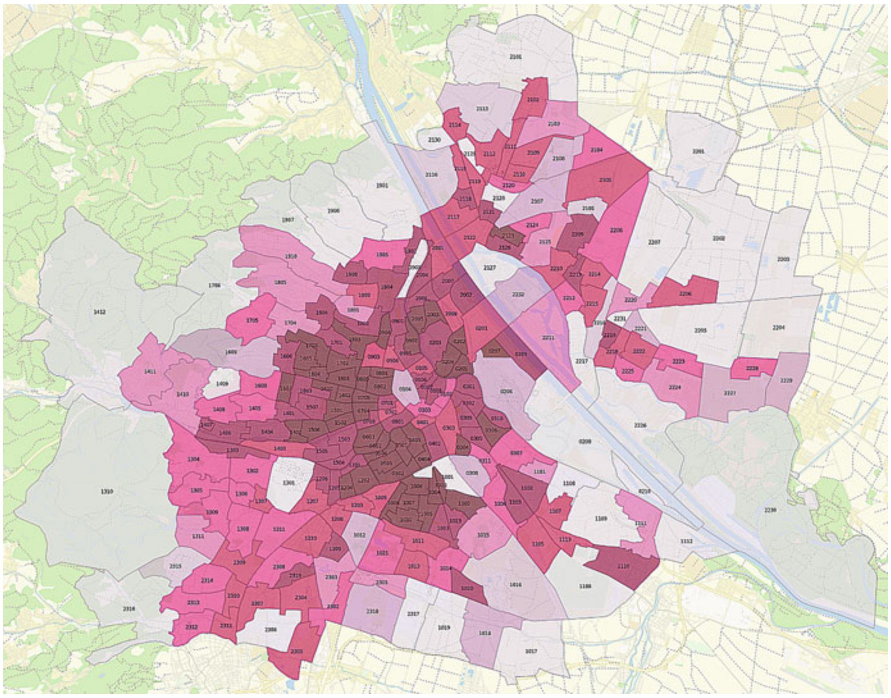


Fig. 15.1 Registration districts of Vienna and their identification numbers. Darker cells correspond to higher population density levels

15.2 Related Literature

There are a huge number of articles in existing literature dealing with location problems. While we focus here on competitive variants we refer to a recently published book by Laporte et al. [30] and two recent review articles by Farahani et al. [21] and Arabani and Farahani [4] for general overviews on recent developments in location science.

Competitive facility location problems have been introduced as early as in the late 1920s by Hotelling [24]. Hotelling originally considered two salespersons competing for market share while placing one facility each on a line. From then on many researchers within the operations research community started to embed competition in their location models, especially Drezner [17] and Hakimi [23] did important early work in this field. There are several review articles on various topics in the competitive facility location domain, such as Ashtiani [5], Eiselt and Laporte [19], Eiselt et al. [20], Kress and Pesch [27] and Eiselt [18].

Several location models have been described in the literature and whereas in the original work by Hotelling [24] the demand of each customer in the market is satisfied by the nearest facility only, Huff [25] introduced a new, for many applications more realistic notion of attraction. Instead of being served by only one facility, each customer splits its demand over all facilities in the market based on an attraction factor. This factor determines a patronizing probability of each facility. A frequent form of an attraction function on a network is $v_{ij} = \frac{a_{ij}}{(d_{ij})^{\beta}}$, where i represents the customer, j a facility, a_{ij} the attractiveness of facility j for customer i, d_{ij} the distance; β determines the influence of the distance on the overall attractiveness. An extension to the basic Huff model is the Pareto-Huff Model [32], in which the attractiveness of facilities with the same or worse quality than a nearer facility is set to zero. Modified versions of these three models are covered by this article for a specific kind of CFLP, which is described in Sect. 15.3, and we denote these models as *binary*, *proportional* and *partially binary* customer behaviour.

Another choice in the area of CFLPs is whether to allow the placement only on a discrete set of predetermined points or arbitrary positions, e.g., on the plane. The solution approaches of these discrete or continuous models are usually significantly different, and most articles focus on one of these two models only. Campos-Rodríguez et al. [10], Drezner [14, 15], Drezner et al. [16], Bhadury et al. [6], Fernández et al. [22], Suárez-Vega et al. [38] focus on continuous location models and Hakimi [23], Laporte and Benati [29], Serra and Revelle [37], Alekseeva and Kochetov [1], Alekseeva et al. [2, 3], Roboredo and Pessoa [34] concentrate on the discrete variant. Here we only consider the discrete variant and model the CFLP as a combinatorial optimization problem.

In CFLPs one can further distinguish the case where a competitor already exists in the market and the case where he has not yet entered the market, i.e., competition with foresight. In the static competition model the competition is assumed to be known and fixed, it is therefore a simpler and easier model to solve [33]. Nevertheless, this is an important case to consider since more complex competition models often use methodologies for the static competition model as a basis. When the competitor only enters the market after the first decision maker fixed his preferred locations, we obtain the leader-follower model. This form of sequential model is a Stackelberg-type model, in which the evaluation of the choice of locations for the leader requires the solution of finding the optimal locations of the follower based on the leader's locations, which is often a non-trivial optimization subproblem on its own. Hakimi [23] was among the first to consider the discrete variant with binary customer behaviour and essential demand and called the resulting problem $(r|p)$-centroid problem.

In practical applications the locations may not be the only decisions to make but also the specifics of the facilities, e.g., quality or size, might need to be decided. The optimal design of the facilities then depends on their location and influences the attractiveness to customers. Also, different opening costs for each possible location and a limited budget can be of interest and considered in the model. Such location and design problems are approached, e.g., by Küçükaydin et al. [28], Saidani et al. [35] and Sáiz et al. [36].

The complexity of the discrete CFLP model with the different customer behaviour scenarios which is considered in this work was shown to be Σ_2^P-

hard by Noltemeier et al. [31]. Even if the locations of the leader are fixed, the remaining problem of finding the optimal locations for the follower, which is also called the $(r|X_p)$-medianoid problem, is NP-hard as proven by Hakimi [23]. These complexity results are strengthened by Davydov et al. [13], who looked at special network structures and showed that the $(r|p)$-centroid problem remains Σ_2^P-hard in particular also for Euclidean distances.

As the presented CFLP models are hard to solve exactly, most solution methods for discrete CFLPs are metaheuristics. Laporte and Benati [29] and Davydov [11] both developed a Tabu search to solve the $(r|p)$-centroid problem which estimates the solution quality by approximating the market share of the follower. The first memetic algorithm for discrete competitive facility location problems was developed by Alekseeva et al. [3]. It combines rather simple genetic operators with a tabu search heuristic utilizing a probabilistic swap neighbourhood structure. The authors solve the linear programming relaxation of a mixed integer linear programming model of the follower problem to approximate the leader's market share. In the same work they propose an exact method based on iteratively solving a single level binary integer model. This was later improved by Alekseeva and Kochetov [1] by introducing strengthening inequalities which consider alternative locations for the follower if a location is already occupied by the leader. Roboredo and Pessoa [34] proposed another exact algorithm for the $(r|p)$-centroid problem based on branch-and-cut. A variable neighbourhood search and another tabu search are described by Davydov et al. [12]. Biesinger et al. [7, 8, 9] developed another memetic algorithm for discrete CFLPs with various customer behaviour and demand models which constitutes today's state-of-the-art solution algorithm from a practical point of view. This chapter describes this memetic algorithm and shows how it is able to solve important class of problems in the area of business and consumer analytics.

15.3 Problem Definition

In this section we will formally state the competitive facility location problem with different customer behaviour scenarios based on the definitions of Biesinger et al. [9]. Given are the numbers $p \geq 1$ and $r \geq 1$ of facilities to be opened by the leader and follower, respectively, and a weighted complete bipartite graph $G = (I, J, E)$ where $I = \{1, \ldots, m\}$ represents the set of potential facility locations, $J = \{1, \ldots, n\}$ represents the set of customers and $E = I \times J$ is the set of edges indicating corresponding assignments. Let $w_j > 0, \forall j \in J$, be the demand of each customer, which corresponds to the (maximal) market share (or turnover) to be earned by the serving facilities, and $d_{ij} > 0, \forall (i, j) \in E$, be the distances between customers and potential facility locations. The goal for the leader is to choose exactly p locations from I for opening facilities in order to maximize his market share under the assumption that the follower in turn chooses r locations for his facilities optimally, maximizing his market share.

We consider six different scenarios which differ in the market share distribution of the customers and we will give a formal description of the market share computation of all these scenarios. The definitions for the binary essential case is taken from [8] and for the other cases from [9]. In the following let (X, Y) be a candidate solution to our competitive facility location problem, where $X \subseteq I,\ |X| = p$, is the set of locations chosen by the leader and $Y \subseteq I,\ |Y| = r$, is the associated set of follower locations. Note that X and Y do not have to be disjoint in general. Further, let $D(j, V) = \min\{d_{ji} \mid i \in V\},\ \forall j \in J,\ V \subseteq I$ be the minimum distance from customer j to all facility locations in set V. Following Kochetov et al. [26] we define the attractiveness of a facility location to a customer by $v_{ij} = \frac{a_{ij}}{(d_{ij})^\beta}$ and define analogously to the minimum distance the maximum attractiveness from customer j to all facility locations in the set V as $A(j, V) = \max\{v_{ji} \mid i \in V\},\ \forall j \in J,\ V \subseteq I$.

In the next sections we follow the classification of the different customer behaviour scenarios from [38] and give definitions of the market share computation of each of these scenarios.

15.3.1 Binary Essential

In this scenario, each customer $j \in J$ chooses a closest facility, hence the owner of this closest facility gains the complete market share w_j. The leader is preferred in case of equal distances. From a practical point of view this scenario is reasonable when both competitors offer the same goods for the same price. Then, the distance is the main criterion where to obtain the goods. As long as $r + p \leq |I|$ the follower will never want to place a facility at a location occupied by the leader and therefore we can assume that $X \cap Y = \emptyset$ for this scenario. Ties among different facilities of the leader (follower) can be broken arbitrarily. The set of customers which are served by one of the follower's facilities is $U^{\mathrm{f}} = \{j \in J \mid D(j, Y) < D(j, X)\}$ and the customers served by the leader is given by $U^{\mathrm{l}} = J \setminus U^{\mathrm{f}}$. Consequently, the total market share of the follower is $p^{\mathrm{f}} = \sum_{j \in U^{\mathrm{f}}} w_j$ and the total market share of the leader is $p^{\mathrm{l}} = \sum_{j \in J} w_j - p^{\mathrm{f}}$. Note that this problem variant is also known as *$(r|p)$-centroid problem* [23].

15.3.2 Proportional Essential

Each customer j splits all of his demand over all opened facilities proportional to their attractiveness. This is a more general scenario which introduces a probability for satisfying a demand based on the distance of the facility. Let $x_i = 1$ if $i \in X$ and $x_i = 0$ otherwise, and $y_i = 1$ if $i \in Y$ and $y_i = 0$ otherwise, $\forall i \in I$. Then, the market share of the follower is

$$p^{\mathrm{f}} = \sum_{j \in J} w_j \frac{\sum_{i \in I} v_{ij} y_i}{\sum_{i \in I} v_{ij} x_i + \sum_{i \in I} v_{ij} y_i}$$

and the market share of the leader is

$$p^{\mathrm{l}} = \sum_{j \in J} w_j - p^{\mathrm{f}}.$$

15.3.3 Partially Binary Essential

Each customer j splits all of his demand over a nearest leader and a nearest follower facility proportional to their attractiveness. This scenario is applicable for products which share the same type but differ in their specifics like brand or price. The potential customer may want to obtain both goods but prefers the one which he can obtain in a nearer store. Let $v_j^L = A(j, X)$, i.e., the highest attraction value from any leader facility to customer j and $v_j^F = A(j, Y)$. Then, the market share of the follower is

$$p^{\mathrm{f}} = \sum_{j \in J} w_j \frac{v_j^F}{v_j^F + v_j^L}$$

and the market share of the leader is

$$p^{\mathrm{l}} = \sum_{j \in J} w_j - p^{\mathrm{f}}.$$

15.3.4 Unessential Demand

In the unessential demand scenarios we need a function that reflects how much the demand of a customer decreases with the distance to a nearest facility. In contrast to the essential demand model, which applies to goods everyone needs like basic foods, with unessential demand we can model other goods like luxury items where consumption is nonobligatory. The consumer might purchase such articles more often if he has an easier access to it, i.e., in our case has a store close nearby. We define this demand reduction function like the attractiveness function as $f(d) = \frac{b}{d^{\gamma}}$, where parameters b and γ control the decrease of demand. Further, we note that when the demand is unessential $\sum_{j \in J} w_j \geq p^{\mathrm{l}} + p^{\mathrm{f}}$, i.e., the total demand satisfied by the leader and the follower together is no longer necessarily equal to the total demand of all customers. In the following we present the profit computation for the unessential scenarios under the different customer choice rules:

- Binary Unessential

$$p^{\mathrm{f}} = \sum_{j \in U^{\mathrm{f}}} w_j f(D(j, Y)) \quad \text{and} \quad p^{\mathrm{l}} = \sum_{j \in U^{\mathrm{l}}} w_j f(D(j, X))$$

- Proportional Unessential

$$p^{\mathrm{f}} = \sum_{j \in J} w_j \frac{\sum_{i \in I} v_{ij} f(d_{ij}) y_i}{\sum_{i \in I} v_{ij} x_i + \sum_{i \in I} v_{ij} y_i} \quad \text{and} \quad p^{\mathrm{l}} = \sum_{j \in J} w_j \frac{\sum_{i \in I} v_{ij} f(d_{ij}) x_i}{\sum_{i \in I} v_{ij} x_i + \sum_{i \in I} v_{ij} y_i}$$

- Partially Binary Unessential

$$p^{\mathrm{f}} = \sum_{j \in J} w_j \frac{v_j^F}{v_j^F + v_j^L} f(D(j, Y)) \quad \text{and} \quad p^{\mathrm{l}} = \sum_{j \in J} w_j \frac{v_j^L}{v_j^F + v_j^L} f(D(j, X)).$$

15.4 Memetic Algorithm

The algorithmic framework for the memetic algorithm (MA) to solve all variants of the problem is described in Biesinger et al. [7, 8, 9]. The objective of the MA is to find optimal locations for the leader so that his market share is a maximum. The MA uses an incomplete solution representation only storing the facilities of the leader represented by a binary vector $x = (x_1, \dots, x_m) \in \{0, 1\}^m$ indicating whether or not a location is chosen. The augmentation of a leader solution with corresponding follower locations is done by solving the follower's subproblem within the solution evaluation. The follower's problem is the problem of finding optimal locations maximizing his market share knowing the locations of the leader who is already in the market. The specific follower's problem depends on the customer behaviour and demand model which have been defined before. For solving the different variants, we will consider mixed integer linear programming (MILPs) models in Sect. 15.4.1. They can be solved directly using a general purpose integer programming solver. For some of the scenarios, however, solving the model exactly for each generated solution candidate is too time-consuming, and therefore a linear programming (LP) relaxation based heuristic is used for the binary customer behaviour and a greedy algorithm for the others. After termination of the MA, the best found solution candidate is always evaluated exactly using the respective MILP model.

The MA consists of a steady-state genetic algorithm starting from a randomly initialized population and a binary tournament selection with replacement. A new solution is always created via recombination and mutation and always replaces a worst solution in the population. The recombination operator creates one new offspring out of two parent solutions by taking all locations equally chosen by both parents and fills the remaining ones by randomly picking a location which is open in either parent. Mutation is based on a series of *swap*-moves, where in each move a facility is closed at one location and re-opened at another so far unoccupied position. For intensification a local search using the swap neighbourhood structure is used. This MA is extended by several advanced techniques which are described in Biesinger et al. [8]:

- An important component is a trie-based complete solution archive, which is a data structure used for identifying duplicate solution candidates and converting them into new, but usually similar yet unvisited solutions. Especially in our case, where the solution evaluation is time-expensive unnecessary re-evaluations are avoided, resulting in a typically overall faster optimization. Additionally, the solution archive ensures higher diversity in the population and the risk of premature convergence is reduced as well.
- The local search is extended further into a tabu search that considers all solutions in the archive as tabu and can therefore escape local optima.
- For the binary customer behaviour the MA also uses a multi-level evaluation scheme that utilizes the greedy and the MILP-based solution evaluation in a combined way to reduce the overall run-time of the MA.

A flowchart of the MA is given in Fig. 15.2. In this bi-level approach, all blue parts (genetic algorithm, solution archive and local search) operate on the leader solution, while the red part (evaluation) considers the follower solution. The genetic algorithm part follows the standard steps in the above description. After a new solution is created via recombination and mutation, it is passed to the solution archive in order to check if it was already considered before. If so, it is converted into a new one before inserting it into the archive. After that, the search in the area of this solution is intensified via local search. Also this part interacts with the solution archive: For every generated move, we check if the emerging solution has been considered before and the move is only evaluated if it leads to a new neighbour solution. Note that the local search step is optional, i.e., only applied to promising new solutions. If it is skipped, the solution directly enters the evaluation step. For the binary customer behaviour the evaluation itself is divided into two steps—a greedy procedure for the upper bound (UB) and an LP procedure for the lower bound (LB), i.e., the MILP model for the follower (see next section) is solved via linear relaxation. Only if the UB is good enough, i.e., the hypothetical maximum revenue for the leader is higher than the so far best solution, the LB is evaluated. If the LB is better than the current solution, the move is applied. After the local search terminates, the improved solution is inserted into the population and replaces the worst solution.

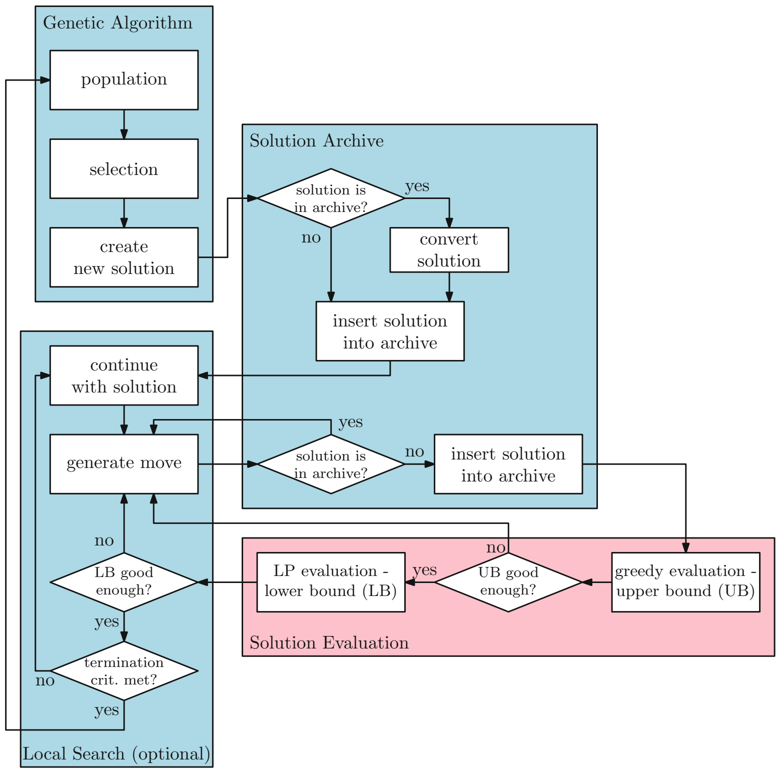


Fig. 15.2 Flowchart of the memetic algorithm

15.4.1 MILP Models for the Follower's Subproblem

The following mixed integer programming models are taken from Alekseeva et al. [2], Kochetov et al. [26] and Biesinger et al. [9]. All models are again based on binary decision variables x_i and y_i, $i \in I$, indicating whether or not the leader/follower opens a facility and location i, respectively.

15.4.1.1 Binary Essential

The model of the follower's problem for the binary essential case uses an additional type of variables:

$$z_j = \begin{cases} 1, & \text{if customer } j \text{ is served by the leader} \\ 0, & \text{if customer } j \text{ is served by the follower} \end{cases} \qquad \forall j \in J.$$

We define the set of facilities that allow the follower to capture customer j if the leader uses solution x:

$$I_j(x) = \{i \in I \mid d_{ij} < \min_{l \in I \mid x_l = 1} d_{lj}\} \quad \forall j \in J.$$

Then the follower's model is defined as follows:

$$\max \sum_{j \in J} w_j(1 - z_j) \tag{15.1}$$

$$\text{s.t.} \sum_{i \in I} y_i = r \tag{15.2}$$

$$1 - z_j \leq \sum_{i \in I_j(x)} y_i \qquad \forall j \in J \tag{15.3}$$

$$x_i + y_i \leq 1 \qquad \forall i \in I \tag{15.4}$$

$$z_j \geq 0 \qquad \forall j \in J \tag{15.5}$$

$$y_i \in \{0, 1\} \qquad \forall i \in I, \forall j \in J. \tag{15.6}$$

The objective function (15.1) maximizes the follower's satisfied demand. Equality (15.2) ensures that the follower places exactly r facilities. Inequalities (15.3) together with the objective function ensure the u_j variables to be set correctly, i.e., they indicate for each customer $j \in J$ from which competitor he is served. Inequalities (15.4) guarantee that the follower does not choose a location where the leader has already opened a facility. Note that all x_i variables are considered as constants here. Variables z_j are not explicitly restricted to binary values because in an optimal solution they will automatically become 0 or 1.

15.4.1.2 Proportional Essential

For the proportional essential case we use a linearization of a model initially proposed by Kochetov et al. [26] who introduced two new types of variables:

$$z_j = \frac{1}{\sum_{i \in I} v_{ij} x_i + \sum_{i \in I} v_{ij} y_i} \qquad \forall j \in J \tag{15.7}$$

and

$$y_{ij} = w_j z_j v_{ij} y_i \qquad \forall i \in I, j \in J. \tag{15.8}$$

Variables y_{ij} have the intuitive meaning that they are the demand of customer j that is supplied by the follower facility at location i, and the z_j variables are basically the denominator of the fractional objective function for a fixed j. This results in the following linear model:

$$\max \sum_{j \in J} \sum_{i \in I} y_{ij} \tag{15.9}$$

s.t.

$$\sum_{i \in I} y_i = r \tag{15.10}$$

$$\sum_{i \in I} y_{ij} + w_j z_j \sum_{i \in I} v_{ij} x_i \leq w_j \qquad \forall j \in J \tag{15.11}$$

$$y_{ij} \leq w_j y_i \qquad \forall i \in I, j \in J \tag{15.12}$$

$$y_{ij} \leq w_j v_{ij} z_j \leq y_{ij} + W(1 - y_i) \qquad \forall i \in I, j \in J \tag{15.13}$$

$$y_{ij} \geq 0, z_j \geq 0 \qquad \forall i \in I, j \in J \tag{15.14}$$

$$y_i \in \{0, 1\} \qquad \forall i \in I. \tag{15.15}$$

Objective function (15.9) maximizes the satisfied demand by the follower. Constraints (15.10) set the number of open facilities to r and constraints (15.11) set the variables y_{ij} by restricting them to not exceed the total demand of customer j minus the demand captured by the leader. The fact that a facility location i can only satisfy a demand of customer j when the follower opens a facility there is ensured by constraints (15.12). Finally, equations (15.8) are fulfilled because of constraints (15.13).

15.4.1.3 Partially Binary Essential

The model for the partially binary essential scenario is similar to the model for the proportional case. The difference is that for each customer we only have to model the ratio of the nearest leader and the nearest follower facility, so we introduce three new kinds of variables:

$$z_j = \frac{1}{v_j^{\mathrm{L}} + v_j^{\mathrm{F}}} \qquad \forall j \in J \tag{15.16}$$

$$\hat{y}_{ij} = \begin{cases} 1, & \text{if } i \text{ is the nearest follower facility to customer } j \\ 0, & \text{else} \end{cases}$$

and

$$y_{ij} = w_j z_j v_{ij} \hat{y}_{ij} \qquad \forall i \in I, j \in J. \tag{15.17}$$

Then, the MILP model is defined as follows:

$$\max \sum_{j \in J} \sum_{i \in I} y_{ij} \tag{15.18}$$

$$\text{s.t.} \sum_{i \in I} y_i = r \tag{15.19}$$

$$\sum_{i \in I} y_{ij} + w_j z_j v_j^{\mathrm{L}} \le w_j \qquad \forall j \in J \tag{15.20}$$

$$y_{ij} \le w_j \hat{y}_{ij} \qquad \forall i \in I, j \in J \tag{15.21}$$

$$y_{ij} \le w_j v_{ij} z_j \le y_{ij} + W(1 - \hat{y}_{ij}) \qquad \forall i \in I, j \in J \tag{15.22}$$

$$\hat{y}_{ij} \le y_i \qquad \forall i \in I, j \in J \tag{15.23}$$

$$\sum_{i \in I} \hat{y}_{ij} = 1 \qquad \forall j \in J \tag{15.24}$$

$$y_i \ge 0, y_{ij} \ge 0, z_j \ge 0 \qquad \forall i \in I, j \in J \tag{15.25}$$

$$\hat{y}_{ij} \in \{0, 1\} \qquad \forall i \in I, j \in J. \tag{15.26}$$

Objective function (15.18) maximizes the demand satisfied by the follower. Constraints (15.20) set the variables y_{ij} by restricting them to not exceed the total demand of customer j minus the demand captured by the leader. The fact that a facility location i can only satisfy a demand of customer j when it is the nearest open follower facility is ensured by constraints (15.21). Equations (15.17) are fulfilled because of constraints (15.22). Constraints (15.23) and (15.24) guarantee that there is exactly one nearest follower facility to each customer and that this location has to be chosen by the follower.

15.4.1.4 Binary Unessential

For the unessential demand models, we can identify two different goals for the follower. He can either aim to minimize the leader's market share (we denote this variant as LMIN) or to maximize his own market share (FMAX), which does have an influence on the choice of locations.

In the binary unessential LMIN scenario only a change in the objective function is needed, which is now as follows:

$$\min \sum_{j \in J} w_j z_j f(D(j, X)) \tag{15.27}$$

and Eqs. (15.2)–(15.6).

For the FMAX scenario, however, further adaptions are necessary because variables indicating which location i hosts a follower facility that is nearer to a customer j are needed. Similar to the previous cases decision variables $\hat{y}_{ij}$ are introduced.

$$\hat{y}_{ij} = \begin{cases} 1, & \text{if } i \text{ is the nearest follower facility to customer } j \\ & \text{and nearer than all leader facilities} \\ 0, & \text{else} \end{cases} \qquad \forall i \in I, \forall j \in J.$$

The MILP model can be stated as follows:

$$\max \sum_{j \in J} w_j \sum_{i \in I} \hat{y}_{ij} f(d_{ij}) \tag{15.28}$$

$$\text{s.t.} \sum_{i \in I} y_i = r \tag{15.29}$$

$$1 - z_j \leq \sum_{i \in I_j(x)} y_i \qquad \forall j \in J \tag{15.30}$$

$$x_i + y_i \leq 1 \qquad \forall i \in I \tag{15.31}$$

$$\hat{y}_{ij} \leq y_i \qquad \forall i \in I, \forall j \in J \tag{15.32}$$

$$\hat{y}_{ij} \leq 1 - z_j \qquad \forall i \in I, \forall j \in J \tag{15.33}$$

$$\sum_{i \in I} \hat{y}_{ij} \leq 1 \qquad \forall j \in J \tag{15.34}$$

$$z_j \geq 0 \qquad \forall j \in J \tag{15.35}$$

$$y_i \in \{0, 1\} \qquad \forall i \in I, \forall j \in J \tag{15.36}$$

$$\hat{y}_{ij} \in \{0, 1\} \qquad \forall i \in I, \forall j \in J. \tag{15.37}$$

There are three new types of constraints to set the $\hat{y}_{ij}$ variables correctly: Constraints (15.32) ensure that if one of these variables is set to one then there must be a follower facility on this location. Furthermore, a $\hat{y}_{ij}$ variable only is set to one if customer j is served by the follower according to constraints (15.33). Last but not least, only one follower facility can be the selected nearest to a customer, and this is guaranteed by constraints (15.34). The change in the objective function models the unessential demand by reducing the demand satisfied by each customer by our demand reduction function f.

15.4.1.5 Proportional Unessential

In contrast to the binary unessential case, in the proportional customer behaviour scenario for both LMIN and FMAX a change in the objective function is needed and for LMIN additionally a change of constraints (15.11):

$$\text{LMIN:} \quad \min \sum_{j \in J} w_j z_j \sum_{i \in I} v_{ij} x_i f(d_{ij}) \tag{15.38}$$

$$\sum_{i \in I} y_{ij} + w_j z_j \sum_{i \in I} v_{ij} x_i = w_j \qquad \forall j \in J \tag{15.39}$$

and Eqs. (15.10), (15.12)–(15.15).

$$\text{FMAX:} \quad \max \sum_{j \in J} \sum_{i \in I} y_{ij} f(d_{ij}) \tag{15.40}$$

and Eqs. (15.10)–(15.15).

15.4.1.6 Partially Binary Unessential

Also for the partially binary case, the objective function changes and for LMIN the constraints (15.20) as well:

$$\text{LMIN:} \quad \min \sum_{j \in J} w_j z_j v_j^{\mathrm{L}} f(V^{-1}(v_j^{\mathrm{L}})) \tag{15.41}$$

$$\sum_{i \in I} y_{ij} + w_j z_j v_j^{\mathrm{L}} = w_j \qquad \forall j \in J \tag{15.42}$$

and Eqs. (15.19), (15.21)–(15.26).

$$\text{FMAX:} \quad \max \sum_{j \in J} \sum_{i \in I} y_{ij} f(d_{ij}) \tag{15.43}$$

and Eqs. (15.19)–(15.26).

In the objective function, $V^{-1}(a)$ is the inverse function for computing the attractiveness, i.e., it yields the distance for a given attractiveness value.

15.5 Case Study

For evaluating the algorithm and studying the differences of the different demand considerations, we perform a case study on Vienna, Austria using real world demographic data from Stadt Wien[1] from 2014. In our hypothetical scenario, a hypermarket chain wants to access the Viennese market, with the knowledge that a rival company has the same intention in the near future.

[1] https://www.data.gv.at/auftritte/?organisation=stadt-wien.

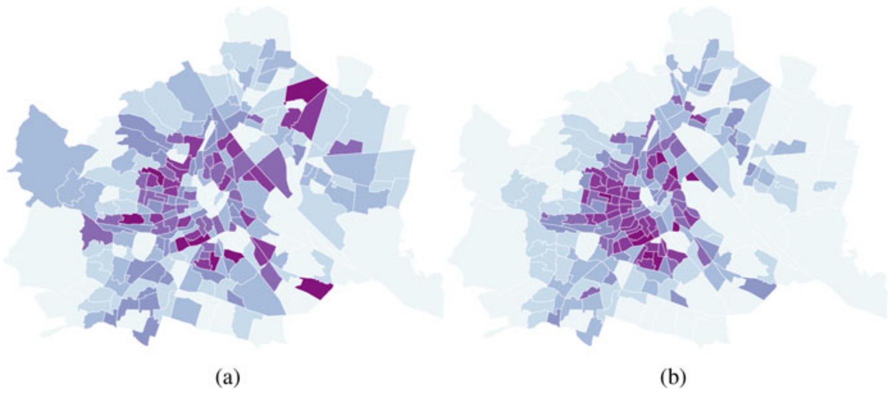


Fig. 15.3 Registration districts of Vienna: (**a**) absolute population and (**b**) density

Vienna has a total population of 1,716,635 (year 2014) and consists of 23 districts. In order to achieve a reasonable level of detail for the strategic planning, we consider the further subdivision into 250 registration districts as planning cells. We assume that the demand of each such district is proportional to its population. In average these cells have a size of around 1.5 km^2 where those in the inner districts are smaller (0.2–0.8 km^2) and the peripheral ones are larger (up to 26 km^2). Though the latter cells would be too large for a detailed planning, they are rather unproblematic since they cover large weakly populated areas and therefore are less interesting for the case study anyway.

Figure 15.3 shows the 250 registration districts of Vienna and the population distribution. The left figure shows the absolute population and the right figure shows the density. Cells with darker and more purple colours indicate a higher population or a higher density.

The leader has to decide in which cells he opens hypermarket stores. Recall that the attractiveness of a store is determined by the distance to the potential customers, i.e., $v_{ij} = \frac{a_{ij}}{(d_{ij})^{\beta}}$ with $a_{ij} = 10{,}000$ for $i \in I, j \in J$ and $\beta = 2$. For the unessential cases we set the parameter b of the demand reduction function to 10,000. We distinguish between two situations for approximating the distance:

- On the cell where a store is opened, we approximate the cell's shape by a circle and take half of its radius as average distance for the inhabitants. This includes the assumption that within the cell, the store is located at a promising location.
- On the other cells, we use the distances between the geometric centres[2] to that of the store's cell as approximation.

For the computational experiments we investigate all six scenarios for the customer behaviour: binary essential, binary unessential, proportional essential,

[2] http://www.wu.ac.at/inst/iir/datarchive/dist_zbez.html.

Table 15.1 Values of the used parameters

Parameter	Value
Population size	100
Mutation rate	μ mutations per iteration, where μ is determined anew at each iteration by a random sample from a Poisson distribution with mean one
Local search rate	Perform a local search for each solution whose quality is within 1% of the best solution found so far
Termination criterion tabu search	Five iterations without improvement
Termination criterion	1800 CPU seconds or 30,000 iterations without improvement
Time limit for solving final model	86,400 CPU seconds

proportional unessential, partially binary. While some of these scenarios are more suited than others for the hypermarket scenario, we want to provide evaluations for all of them. Also, in a real world scenario, a combination of these elementary behaviours is most likely to happen.

In our case study we assume that the leader knows that the follower has a budget to open five stores in Vienna, i.e., $r = 5$. To compete over the market share, the leader considers different number of stores and their optimal placement.

15.5.1 Results

For obtaining the computational results, we used the same parameter setting as in Biesinger et al. [9], except the time limit, and all the runs were executed on a single core of an Intel Xeon Quadcore with 2.53 GHz and 3 GB RAM. The used parameters are shown in Table 15.1.

As the instance is substantially larger than the instances in Biesinger et al. [9], especially for the non-binary scenarios, the termination criterion for the memetic algorithm and for solving the final model for the follower's problem is set to an increased run-time. The models for the solution evaluation are solved using the IBM ILOG CPLEX Optimizer in version 12.6.2.

Tables 15.2, 15.3 and 15.4 show the numerical results of the tests for fixed $r = 5$ and different values for p for all scenarios. The values in the tables under column *ms* are the average market shares over 30 runs for the leader and the follower, respectively, expressed relative to the total demand of the market (Vienna's population). Next to the market shares also the corresponding standard deviations (sd) and the median run-times in seconds (t) are given. The results are shown for each scenario separately and Table 15.2 shows the results for binary customer behaviour, Table 15.3 for proportional customer behaviour and Table 15.4 for the

Table 15.2 Results for *binary* customer behaviour, $r = 5$, and different values for p

Leader										Follower	
	Essential			Unessential LMIN			Unessential FMAX			LMIN	FMAX
p	ms (%)	sd (%)	t (s)	ms (%)	sd (%)	t (s)	ms (%)	sd (%)	t (s)	ms (%)	ms (%)
2	5.35	0.00	289	0.43	0.04	1800	0.56	0.05	1839	1.09	1.68
3	23.17	0.00	223	0.59	0.06	1800	0.75	0.06	1859	1.00	1.67
4	37.37	0.00	111	0.76	0.07	1800	0.91	0.08	1884	1.13	1.64
5	46.10	0.25	133	0.92	0.11	1800	1.04	0.08	1925	1.12	1.65
6	**53.97**	0.11	121	1.06	0.08	1800	1.21	0.09	1939	1.09	1.63
7	**59.77**	0.90	123	**1.19**	0.10	1800	1.34	0.08	1964	1.13	1.61
8	**63.08**	0.52	136	**1.33**	0.08	1800	1.48	0.09	1963	1.11	1.59
9	**66.24**	0.55	185	**1.49**	0.11	1800	**1.61**	0.10	1986	1.13	1.59
10	**69.10**	0.61	216	**1.60**	0.10	1800	**1.76**	0.12	1996	1.13	1.58

Table 15.3 Results for *proportional* customer behaviour, $r = 5$, and different values for p

Leader										Follower	
	Essential			Unessential LMIN			Unessential FMAX			LMIN	FMAX
p	ms (%)	sd (%)	t (s)	ms (%)	sd (%)	t (s)	ms (%)	sd (%)	t (s)	ms (%)	ms (%)
2	(27.54)	0.21	88,200	0.31	0.00	12,463	0.67	0.00	3435	0.85	1.47
3	(37.05)	0.21	88,200	0.43	0.02	4241	0.87	0.05	3899	0.80	1.44
4	(43.92)	0.63	88,200	0.55	0.03	3302	1.04	0.07	3866	0.73	1.41
3	(37.05)	0.21	88,200	0.43	0.02	4241	0.87	0.05	3899	0.80	1.44
5	(49.23)	0.58	88,200	**0.67**	0.04	3170	1.21	0.07	3758	0.67	1.38
3	(37.05)	0.21	88,200	0.43	0.02	4241	0.87	0.05	3899	0.80	1.44
6	**(53.51)**	0.75	88,200	**0.82**	0.05	3122	1.36	0.09	3988	0.67	1.37
3	(37.05)	0.21	88,200	0.43	0.02	4241	0.87	0.05	3899	0.80	1.44
7	**(57.25)**	0.58	88,200	**0.96**	0.08	3292	**1.48**	0.10	4040	0.68	1.35
3	(37.05)	0.21	88,200	0.43	0.02	4241	0.87	0.05	3899	0.80	1.44
8	**(60.37)**	0.69	88,200	**1.09**	0.08	3246	**1.59**	0.08	3800	0.66	1.35
3	(37.05)	0.21	88,200	0.43	0.02	4241	0.87	0.05	3899	0.80	1.44
9	**(62.57)**	0.59	88,200	**1.22**	0.09	3075	**1.69**	0.10	4138	0.65	1.32
3	(37.05)	0.21	88,200	0.43	0.02	4241	0.87	0.05	3899	0.80	1.44
10	**(64.51)**	0.50	88,200	**1.33**	0.10	2979	**1.84**	0.12	4292	0.65	1.29

partially binary customer behaviour. The values for the proportional essential case are listed in parentheses indicating that CPLEX could not solve the model within the time limit and the values being therefore only upper bounds on the leader's market share. Bold values indicate the values for p for which the leader could achieve a greater market share than the follower. We have to keep in mind, however, that the presented solution method solves the problem only heuristically and therefore the values for the leader's market share represent only lower bounds to the actual optimal value.

Table 15.4 Results for *partially binary* customer behaviour, $r = 5$, and different values for p

Leader										Follower	
	Essential			Unessential LMIN			Unessential FMAX			LMIN	FMAX
p	ms (%)	sd (%)	t (s)	ms (%)	sd (%)	t (s)	ms (%)	sd (%)	t (s)	ms (%)	ms (%)
2	33.88	0.00	33,527	0.36	0.00	6318	0.71	0.00	2271	0.35	1.53
3	40.91	0.00	22,428	0.54	0.00	6059	1.03	0.00	2683	0.55	1.44
4	45.32	0.36	19,564	**0.72**	0.00	4108	1.32	0.00	2567	0.70	1.37
5	49.26	0.60	17,914	**0.89**	0.00	4829	**1.56**	0.00	2688	0.89	1.34
6	**52.35**	0.44	15,280	**1.16**	0.00	3365	**1.82**	0.01	2689	0.88	1.31
7	**55.18**	0.49	14,930	**1.41**	0.00	3260	**2.04**	0.02	2937	0.88	1.28
8	**57.34**	0.61	12,975	**1.64**	0.02	3304	**2.24**	0.04	2972	0.87	1.26
9	**59.25**	0.60	12,546	**1.83**	0.04	3319	**2.45**	0.04	2932	0.88	1.26
10	**61.14**	0.58	11,514	**1.99**	0.04	3251	**2.61**	0.07	2656	0.84	1.27

For the essential demand model we see that the leader's market share is naturally steadily increasing with increasing values of p. We also see that if $r = p$ in this demand model the follower has always a greater market share. It is clear that for proportional and partially binary customer behaviour the leader can have at most 50% market share because the follower can choose the same locations as the leader and the demand then always splits equally between them. For the binary customer behaviour, however, the follower's advantage of knowing the locations of the leader outweighs the leader's advantage of choosing his locations first.

When looking at the results for the unessential demand model we observe that the total market share of the leader and the follower is much lower than for the essential demand model and always less than 3% for each individual player. This is due to the assumption that attractiveness of a facility decreases quadratically with respect to the distance. In the LMIN scenarios the follower can always achieve his goal to lower the market share of the leader, as it is always lower than in the FMAX scenarios for the same value of p. However, the follower's market share is also always significantly lower in the LMIN cases and therefore this behaviour is not beneficial for the overall penetration of the market.

The tables further show that for both LMIN and FMAX in the partially binary behaviour scenario the leader needs the least number of facilities to have a greater market share than the follower and in the binary behaviour he needs the most. Especially in the partially binary LMIN case the leader needs only four stores, which means that the advantage of being the first in the market is larger in this case.

Regarding the variance of the results over the 30 runs we observed that the standard deviation increases with increasing p but is always less than 0.91 for the essential demand model and less than 0.13 for unessential demand. In 17 cases for low values of p the standard deviation was even zero which means that the algorithm stopped with a solution of the same quality in each run. Overall, the MA therefore is quite robust.

The run-time of the tests differs in each scenario, and in all but the binary cases, in which solving the model needs less than 1 s, the run-time of exactly solving the follower's subproblem model for the final solution is significantly high and in the most difficult cases even dominates the total run-time of the MA. In general the essential models are harder to solve than the unessential models. We conclude from Tables 15.3 and 15.4 that the models for the proportional cases are the hardest to solve followed by the models for the partially binary cases. Also, for proportional and partially binary customer behaviour the LMIN scenario needs more time to solve than the FMAX scenario. While the run-times of the FMAX scenarios do not differ much for different values of p, in the LMIN cases the run-time tends to decrease with increasing p.

To get a better understanding of the found solutions, Figs. 15.4, 15.5 and 15.6 show best found solutions for $r = p = 5$ within a map of Vienna for different customer behaviours and demand models. For the unessential demand always the FMAX variant is illustrated. In these figures the leader is marked with blue colour

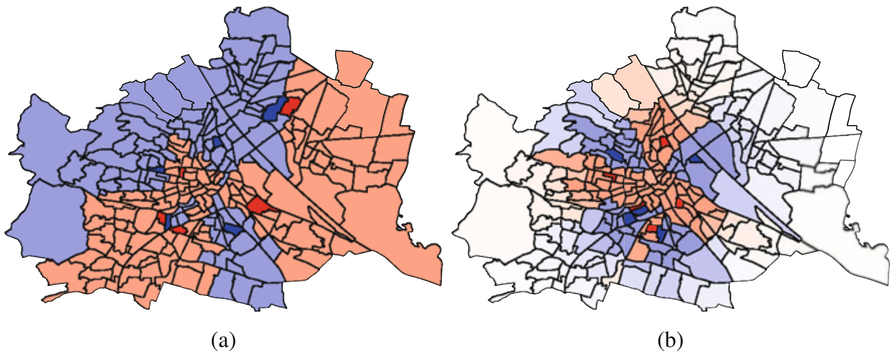


Fig. 15.4 Best found solution for *binary* customer behaviour and (**a**) essential, and (**b**) unessential demand models with $r = p = 5$

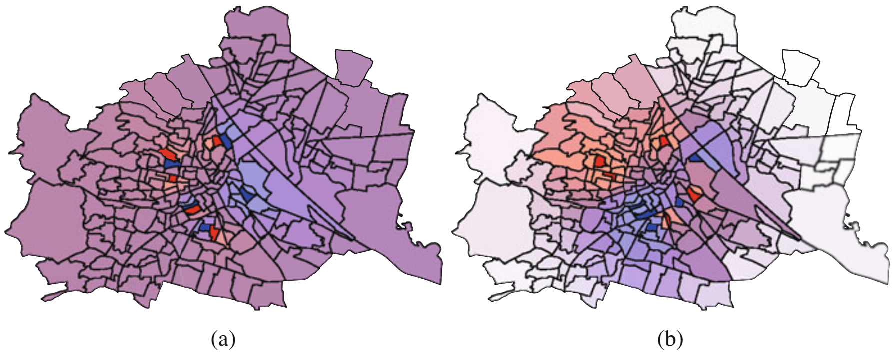


Fig. 15.5 Best found solution for *proportional* customer behaviour and (**a**) essential, and (**b**) unessential demand models with $r = p = 5$

and the follower with red and the actual chosen locations are the dark districts. If the leader and the follower choose the same planning cell for their locations, it is marked with purple. The colour of the remaining districts indicates the amount of demand that is captured by the leader and the follower, respectively. The more blue an area is, the more demand is satisfied by the leader. The same holds for the follower with red. Purple areas are served by both the leader and the follower.

In Fig. 15.4 left we see that in case of a binary behaviour, each cell is either served by the leader or the follower. All cells are roughly equally divided between both competitors. We also see that in the lower left area, the follower is able to significantly reduce the influence of a leader's facility by placing two facilities next to it. In the unessential case on the right, there are large white areas where demand is hardly fulfilled because of their large distances to the facilities.

In Figs. 15.5 and 15.6 we see that basically all districts are served by both leader and follower facilities. In the essential model, especially for the proportional behaviour in Fig. 15.5, the facilities are placed very centrally since each customer has a probability to visit all facilities. For the partially binary behaviour in Fig. 15.6, the facilities are more spread out, but some profitable districts are occupied by both competitors.

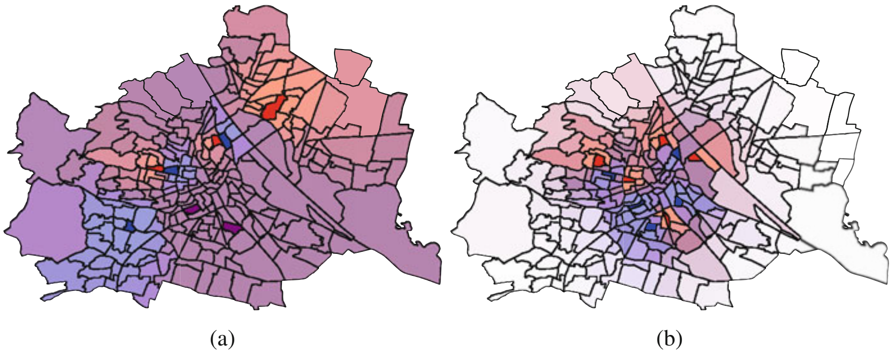


Fig. 15.6 Best found solution for *partially binary* customer behaviour and (**a**) essential, and (**b**) unessential demand models with $r = p = 5$

Overall we observe that in the unessential demand models the facility locations tend to be more central than in the essential demand models, which corresponds to our intuition of choosing the profitable districts first. We also see in all six figures that some locations are often chosen regardless of the considered scenario. For essential demand two locations were chosen for two different customer behaviour models and for unessential demand four locations are chosen twice. These indicate promising, robust choices which could also be used for real-world applications, where the actual customer behaviour is not so easy to determine. For practical applications of these results one also has to decide on the demand model one wants to assume, which highly depends on the offered goods. While some goods like basic foods are obviously essential, other goods are not so easily categorized. Especially in our case, as hypermarkets offer a variety of essential and unessential goods, the demand model is not totally accurate. Nevertheless, the practitioner could simulate all kinds of different customer behaviour and demand models and make a decision based on the results of all these scenarios.

15.6 Conclusions

Competitive Facility Location Problems are an important topic in business and consumer analytics. They constitute a huge class of problems and there exist many different variants, each focusing on different aspects. This chapter shows a specific and rather basic discrete variant, where the only decisions are the locations of the facilities. Due to the complexity of CFLPs, exact methods are limited to small instances and therefore metaheuristics and especially memetic algorithms are promising solution algorithms. Our memetic algorithm combines genetic operators with local search and underlying exact methods based on mixed integer linear programming for the solution evaluation. The modularity of the presented two-level approach is shown in six different customer behaviour and demand models. We performed a case study concerning two competitors who both want to open hypermarkets in Vienna. It is assumed that both are new to the market and that one has to choose his locations first. The possible locations are simplified by assuming them to lie in the geometric centres of the existing 250 registration districts, and the demand is assumed to correspond to the actual population in these districts. We analysed several scenarios simulating different customer behaviours and showed the resulting market share for different numbers of facilities which to be opened. Although we considered more realistic variants of facility location problem still we neglected several other factors. Usually, two companies do not enter a completely fresh market and therefore at least some stores already exist, which could be included in the model. Another extension could be the inclusion of the decision on the design of the facilities. This would introduce costs for opening a facility and a limited budget for the competitors which would lead to a variable value of r and p.

Acknowledgement This work is supported by the Austrian Science Fund (FWF) under grant P24660-N23.

References

1. Alekseeva E, Kochetov Y (2013) Metaheuristics and Exact Methods for the Discrete $(r|p)$-Centroid Problem. In: Talbi EG (ed) Metaheuristics for Bi-level Optimization, Studies in Computational Intelligence, vol 482, Springer Berlin Heidelberg, pp 189–219
2. Alekseeva E, Kochetova N, Kochetov Y, Plyasunov A (2009) A Hybrid Memetic Algorithm for the Competitive p-Median Problem. In: Bakhtadze N, Dolgui A (eds) Information Control Problems in Manufacturing, vol 13, International Federation of Automatic Control, pp 1533–1537

3. Alekseeva E, Kochetova N, Kochetov Y, Plyasunov A (2010) Heuristic and Exact Methods for the Discrete $(r|p)$-Centroid Problem. In: Cowling P, Merz P (eds) Evolutionary Computation in Combinatorial Optimization, Lecture Notes in Computer Science, vol 6022, Springer Berlin Heidelberg, pp 11–22
4. Arabani AB, Farahani RZ (2012) Facility location dynamics: An overview of classifications and applications. Computers & Industrial Engineering 62(1):408–420
5. Ashtiani MG (2016) Competitive location: a state-of-art review. International Journal of Industrial Engineering Computations 7(1):1–18
6. Bhadury J, Eiselt H, Jaramillo J (2003) An alternating heuristic for medianoid and centroid problems in the plane. Computers & Operations Research 30(4):553–565
7. Biesinger B, Hu B, Raidl GR (2014) An evolutionary algorithm for the leader-follower facility location problem with proportional customer behavior. In: Conference Proceedings of Learning and Intelligent Optimization Conference (LION 8), Springer, LNCS, vol 8426, pp 203–217
8. Biesinger B, Hu B, Raidl G (2015a) A hybrid genetic algorithm with solution archive for the discrete $(r|p)$-centroid problem. Journal of Heuristics 21(3):391–431
9. Biesinger B, Hu B, Raidl G (2015b) Models and algorithms for competitive facility location problems with different customer behavior. Annals of Mathematics and Artificial Intelligence pp 1–27
10. Campos-Rodríguez C, Moreno-Pérez J, Noltemeier H, Santos-Peñate D (2009) Two-Swarm PSO for Competitive Location Problems. In: Krasnogor N, Melián-Batista M, Pérez J, Moreno-Vega J, Pelta D (eds) Nature Inspired Cooperative Strategies for Optimization (NICSO 2008), Studies in Computational Intelligence, vol 236, Springer Berlin Heidelberg, pp 115–126
11. Davydov I (2012) Tabu search for the discrete $(r|p)$-centroid problem. Diskretn Anal Issled Oper 19(2):19–40
12. Davydov I, Kochetov Y, Mladenovic N, Urosevic D (2014a) Fast metaheuristics for the discrete $(r|p)$-centroid problem. Automation and Remote Control 75(4):677–687
13. Davydov I, Kochetov Y, Plyasunov A (2014b) On the complexity of the $(r|p)$-centroid problem in the plane. Top 22(2):614–623
14. Drezner T (1994) Optimal continuous location of a retail facility, facility attractiveness, and market share: An interactive model. Journal of Retailing 70(1):49–64
15. Drezner T (1998) Location of multiple retail facilities with limited budget constraints — in continuous space. Journal of Retailing and Consumer Services 5(3):173–184
16. Drezner T, Drezner Z, Salhi S (2002) Solving the multiple competitive facilities location problem. European Journal of Operational Research 142(1):138–151
17. Drezner Z (1981) On a modified one-center model. Management Science 27(7):848–851
18. Eiselt HA (2011) Foundations of Location Analysis, Springer US, Boston, MA, chap Equilibria in Competitive Location Models, pp 139–162
19. Eiselt HA, Laporte G (1997) Sequential location problems. European Journal of Operational Research 96(2):217–231
20. Eiselt HA, Marianov V, Drezner T (2015) Location Science, Springer International Publishing, chap Competitive Location Models, pp 365–398
21. Farahani RZ, Hekmatfar M, Arabani AB, Nikbakhsh E (2013) Hub location problems: A review of models, classification, solution techniques, and applications. Computers & Industrial Engineering 64(4):1096–1109
22. Fernández J, Pelegrı B, Plastria F, Tóth B, et al. (2007) Solving a Huff-like competitive location and design model for profit maximization in the plane. European Journal of Operational Research 179(3):1274–1287
23. Hakimi S (1983) On locating new facilities in a competitive environment. European Journal of Operational Research 12(1):29–35
24. Hotelling H (1929) Stability in competition. The Economic Journal 39(153):41–57
25. Huff DL (1964) Defining and estimating a trading area. The Journal of Marketing pp 34–38
26. Kochetov Y, Kochetova N, Plyasunov A (2013) A metaheuristic for the leader-follower facility location and design problem. In: Lau H, Van Hentenryck P, Raidl G (eds) Proceedings of the 10th Metaheuristics International Conference (MIC 2013), Singapore, pp 32/1–32/3

27. Kress D, Pesch E (2012) Sequential competitive location on networks. European Journal of Operational Research 217(3):483–499
28. Küçükaydin H, Aras N, Altınel IK (2011) Competitive facility location problem with attractiveness adjustment of the follower: A bilevel programming model and its solution. European Journal of Operational Research 208(3):206–220
29. Laporte G, Benati S (1994) Tabu Search Algorithms for the $(r|X_p)$-medianoid and $(r|p)$-centroid Problems. Location Science 2:193–204
30. Laporte G, Nickel S, da Gama FS (2015) Location science, vol 145. Springer
31. Noltemeier H, Spoerhase J, Wirth HC (2007) Multiple voting location and single voting location on trees. European Journal of Operational Research 181(2):654–667
32. Peeters PH, Plastria F (1998) Discretization results for the Huff and Pareto-Huff competitive location models on networks. Top 6(2):247–260
33. Plastria F (2001) Static competitive facility location: An overview of optimisation approaches. European Journal of Operational Research 129(3):461–470
34. Roboredo M, Pessoa A (2013) A branch-and-cut algorithm for the discrete $(r|p)$-centroid problem. European Journal of Operational Research 224(1):101–109
35. Saidani N, Chu F, Chen H (2012) Competitive facility location and design with reactions of competitors already in the market. European journal of operational research 219(1):9–17
36. Sáiz ME, Hendrix EM, Pelegrín B (2011) On Nash equilibria of a competitive location-design problem. European Journal of Operational Research 210(3):588–593
37. Serra D, Revelle C (1994) Competitive location in discrete space. Economics Working Papers 96, Dep. of Economics and Business, Univ. Pompeu Fabra, Spain, Technical Report
38. Suárez-Vega R, Santos-Peñate D, Pablo DG (2004) Competitive Multifacility Location on Networks: the $(r|X_p)$-Medianoid Problem. Journal of Regional Science 44(3):569–588

Chapter 16
Visualizing Products and Consumers: A Gestalt Theory Inspired Method

Claudio Sanhueza Lobos, Natalie Jane de Vries, Mario Inostroza-Ponta, Regina Berretta, and Pablo Moscato

Abstract Motivated by the ability that visualizations have for explaining complex relationships, we revisit an alternative and efficient algorithm for visualizing relationships between objects. *QAPgrid* was proposed to solve the problem of allocating objects in a grid. The algorithm uses as its mathematical model the NP-hard Quadratic Assignment Problem. We implemented an efficient Memetic Algorithm for solving the layout optimization problem. The algorithm has been previously tested on a variety of datasets with good results. In this chapter, we explore the algorithm's potential for analysing social networks. In particular, we examined the collaboration network created around the artificial world of the Marvel Universe comic books. We show how the algorithm can generate accurate and informative visualizations for analysing complex graphs. Furthermore, to demonstrate an alternative use of the algorithm, we analyse and visualize products (wines) and customers (telecom clients). In doing so, we show how the algorithm is suitable for the analysis of different types of objects organized as a network.

Keywords Memetic algorithm · Customer Churn · Effective visualizations · Quadratic assignment problem

C. Sanhueza Lobos (✉) · N. J. de Vries · R. Berretta · P. Moscato
School of Electrical Engineering and Computing, The University of Newcastle, Callaghan, NSW, Australia
e-mail: claudio.sanhuezalobos@uon.edu.au; claudio.sanhuezalobos@newcastle.edu.au; natalie.devries@newcastle.edu.au; regina.berretta@newcastle.edu.au; Pablo.Moscato@newcastle.edu.au

M. Inostroza-Ponta
Universidad de Santiago de Chile, Santiago, Chile
e-mail: mario.inostroza@usach.cl

P. Moscato, N. J. de Vries (eds.), *Business and Consumer Analytics: New Ideas*,
https://doi.org/10.1007/978-3-030-06222-4_16

16.1 The Visual Dimension in Business and Customer Analytics

Nothing is more revitalizing than being greeted by your favourite drink just as you walk through the door of your local cafe. This is one example of what we perceive as high-quality customer service. Although the barista knows you take an espresso every Tuesday at 8 a.m., in a digital space it is harder for your preferred brands to personalize your experience. Imagine for a moment that a mobile data platform wants to understand everyday choices and behaviours to identify who we are and what we value. We might develop predictive models based on mobile phone daily usages. In doing so, we could help to better understand the clients and how they interacted with each other via the platform. This is just an example of how advertisers around the world pursue data-driven efforts based on relevant users' preferences. In this context, modern analysis tools are continuously launched to help in the understanding of the gathered data. In particular, visualization platforms are well suited for these tasks, where a summary or simplification is useful to get basic patterns of the context.

It is well-known to experienced data science practitioners that visualization techniques can be both a curse and a blessing. The reason for this is that the data we deal with in many fields, and in particular in business and consumer analytics, is naturally embedded in a high dimensional space. For example, in our introductory case each user behaviour might be represented by a finite, but large, set of features and/or activities (e.g. number of calls, frequency of text messages, and so on). However, any attempt to bring the data into 2D (e.g. the computer screen, perhaps with the possibility of zooming in and out), 3D (again a computer screen with the possibility of interacting with the data via rotations and virtual movements in the space), and 4D (e.g. movies, including the possibility to go forward or backward in time and at the same time moving in 3D space) is, no matter how powerful, an intrinsically limited approach. We are in a kind of "hopeless situation" as the data's true nature belongs in a space where visualization is complex. Any attempt to "visualize" means that it would require some sort of "information preserving mapping" into a lower-dimensional space. However, true progress can be achieved when useful projections can be found [1].

In some sense, visualization is still considered by many both an art and a science. No book that includes something about visualization can really escape a reference to Tufte's classic work [2]; we highlight a particular passage from the epilogue:

> Design is choice. The theory of the visual display of quantitative information consists of principles that generate design options and that guide choices among options. The principles should not be applied rigidly or in a peevish spirit; they are not logically or mathematically certain; and it is better to violate any principle than to place graceless or inelegant marks on paper. Most principles of design should be greeted with some skepticism, for word authority can dominate our vision, and we may come to see only through the lenses of word authority rather than with our own eyes.

Is our quest to find some underlying guidance thus hopeless? We do not think so. We generally consider that good visualizations of data are those that bring some new insights about patterns of relationships that are present in the data. These patterns can become apparent, thanks to a lower-dimensional space projection and some sort of exploitation of "gestalt" properties. "Gestaltism" is a theory of the mind attributed to the Berlin School of experimental psychology and aims to understand the general laws that explain our ability to acquire and maintain some perceptions in a world that, otherwise, would look apparently chaotic. The so-called *"Gestalt Effect"* relates to our capacity to generate "higher level forms". For example, the visual recognition of complex figures instead of just juxtapositions of unrelated simple elements.

16.1.1 The Gestalt Laws of Grouping

Our human visual system has processes that allow us to recognize foreground from background, identify objects presented in different orientations, and accurately interpret spatial signs. Our visual information processing skills are developed from birth [3].

Visual information processing refers to a group of visual cognitive skills used for extracting and organizing information. This sensory information is integrated with higher cognitive functions. Writing books, riding a bike, or internally analysing a problem all require visual processing skills. Visual-analysis skills are a group of abilities for recognition, recall, and manipulation of visual information. Activities such as judging similarities or differences amongst forms and symbols, and remembering them are part of visual-analysis skills. Visualization is one of the types of visual-analysis abilities. Visualization is the capacity to recall and manipulate visual information by using images. The author in [4] outlined four sub-processes in visualization. The first two processes involve image generation and maintenance. The last two processes involve manipulation of the images by scanning or mental rotation.

For business analytics, we should exploit the inherent abilities of these systems to infer information from data presented to us. Data visualization can be then viewed as an interplay between two different perspectives: the computational perspective and the user perspective. Computer scientists attempt to build tools, algorithms, and strategies to optimally present high-dimensional data in lower-dimensional spaces. When we talk about using automatic approaches for creating visualizations typically the focus is on its *efficiency*, i.e. drawing a large amount of data quickly. On the other hand, from a user perspective, the aim is to create *effective visualizations*, i.e. whether they can *transmit* a meaningful message to the observer. We refer the reader to [5] for a complete review of visualization concepts.

We have interesting theories that could guide our selection of principles to bring together these two perspectives. We refer to the Gestalt principles [6] for such a guidance. The Gestalt laws of grouping were first introduced[1] by Wertheimer in 1923 [7–9].

Some of these Laws/Principles are:

- **Law of Proximity**: When individuals perceive a set of objects, they recognize those that are close to each other as forming a group.
- **Law of Similarity** or **Law of Prägnanz**: Elements within an assortment of objects are perceptually grouped together if they are similar in shape, shading, colour, or other clearly identifiable qualities.
- **Law of Good Gestalt**: Elements of objects tend to be perceptually grouped together if they form patterns that are either orderly, regular, or "simple". Perception tends to "eliminate complexity" and things that could be unfamiliar and, therefore, we mentally prioritize regularity over spatial relations.
- **Law of Closure**: We perceive objects as being "a whole" when they are actually "not complete". The idea is that we have a tendency to try to complete a relatively regular figure as a way to increase the regularity of the surrounding stimuli that may not be complete.

Other Gestalt laws are the *Laws of Common Fate*, *Symmetry*, *Continuity*, and *Past Experience*. We will return to these later on, but we have selected the above four first to be discussed so that they can guide our introductory discussion. These laws might influence many aspects of the User Experience Design (UX); thus, many Information and Communications Technology (ICT) professionals are aware of them and use them for a variety of reasons.

There is a clear interest in the ICT community to employ Gestalt Laws for designing new products [10, 11], user interfaces [12, 13], and websites [14]. It has also influenced the area of diagram design [15], web page segmentation [16], web page classification [17, 18], product graph segmentation and demand prediction [19], image segmentation [20], the location of services [21], user engagement and interaction [22], sample classification [23], information visualization [24], multimodal data representation, [25], building brands [26], and graph drawing [27].

Gestalt Theory can help us to understand high-dimensional datasets [28, 29] by exploiting our selective attention [30]. We will present an example of the application of these ideas on a dataset that allows some intuitive understanding of the method even though the dataset is big enough to show some of the challenges that high-dimensional datasets present. The QAPgrid algorithm for visualization has already been successfully applied for an Indo-European language instance, ranking of universities instance, and in a Gene Ontology-based study showing the scalability and precision of the method as a novel visualization tool [31].

[1]http://psychclassics.yorku.ca/Wertheimer/Forms/forms.htm.

16.2 Preliminaries

We provide here some preliminaries about the methodology used in each case study. First, we present how the well-known Quadratic Assignment Problem can be adapted for visual analyses. Later, we describe the details of the Memetic Algorithm we have used to address the visualization problem for the keen learners and more advanced computer science researchers.

16.2.1 The Quadratic Assignment Problem

In 1957, Koopmans et al. [32] introduced the Quadratic Assignment Problem (QAP) for a similar problem. The problem is NP-hard [33], and it has proven to be difficult to be solved to optimality even for tiny instances. However, good heuristic and metaheuristic methods exist for it and it certainly fits well with our interest as a mathematical model.

The QAP can be informally described as the problem of allocating a set of n objects to a set of m locations ($m \geq n$). For each pair of allocated objects, a cost is associated to this decision and it is proportional to the distance between the two positions to which the objects have been allocated and a certain measure of "flow" between the pair of objects. The final objective is to minimize the global cost function which is the total sum of all these costs. The Quadratic Assignment Problem is formally stated in Eq. (16.1):

$$\underset{p \in P_n}{\text{minimize}} \;\; C(p) = \sum_{i=1}^{n} \sum_{j=1}^{n} d_{ij} f_{p(i)p(j)}, \tag{16.1}$$

where $D = (d_{ij})$ is the distance between locations i and j, P is a function that assigns a unique location to each object, and $F = \{f_{p(i)p(j)}\}$ is the matrix of flow between the objects $p(i) = k$ and $p(j) = l$ that have been allocated to positions i and j, respectively. The product $d_{ij}\ f_{p(i)p(j)}$ is the cost of locating facility $p(i) = k$ to the location i and facility $p(j) = l$ to the location j.

As an example, let us consider a hospital planning problem. New buildings must be created, and the objective is to minimize the walking distance of the staff members. Also, suppose there are m available sites and n facilities to locate. Let d_{ij} represent the walking distances between the location i and j where the new facilities should be placed. Moreover, let f_{kl} represent the number of staff members per week who walk between the facilities k and l. Therefore, the problem is to locate the hospital's facilities in order to minimize the total accumulated walking distance (per week). A solution to the planning problem can be represented by a permutation p of size n, where $p(i) = k$ means that the site i ($1 \leq i \leq m$) is occupied by the facility k ($1 \leq k \leq n$). The product $d_{ij} f_{p(i)p(j)}$ describes the weekly walking distance of people who move between facilities $p(i) = k$ and $p(j) = l$. In this example, we assume that the cost to locate a facility does not depend upon the location. In the case where it does, we will denote it by $b_{p(i)i}$ the cost of assigning facility $p(i) = k$ to the location i and this would also play a role in determining the optimal assignment.

16.2.2 Creating an QAPgrid Instance

Solving the visualization problem requires the generation of a QAP instance. This condition means we have to create the matrices D and F. Let $D = (d_{ij})$ be the Euclidean distance between the locations i and j on a grid with m locations, and let $S = (s_{kl})$ be the distance between objects k and l. Then, the flow matrix $F = (f_{kl})$ can be defined as:

$$f_{kl} = \begin{cases} \dfrac{1}{s_{kl}} & , \forall k \neq l, \ \forall 1 \leq k \leq n, \ 1 \leq l \leq n \\ \\ 0 & , k = l. \end{cases} \tag{16.2}$$

The transformation method ensures that small distances between objects lead to large flows between them. Consequently, near optimal solutions of such an instance of the QAP will have objects that are very close in nearby locations. The perceptual Law of Proximity will not be misleading as indeed similar objects are likely to be located in close proximity.

16.2.2.1 Organization of the Whole Dataset as a Cluster of Clusters

A top-level QAP instance can also be defined to enhance a hierarchical understanding of the dataset. We consider a partition of the set of objects (clusters) as returned by the MST-kNN algorithm [34]. Let $C = \{C_1, C_2, \ldots, C_c\}$ be a set of c disjoint clusters returned by MST-kNN. Then, a flow matrix for the Layout of clusters can be stated as shown in Eq. (16.3)

$$f_{C_{pq}} = \begin{cases} \dfrac{\sum\limits_{\forall k \in C_p} \sum\limits_{\forall l \in C_q} f_{kl}}{|C_p||C_q|} & , \forall p \neq q, \ \forall\, 1 \leq p, q \leq c, \ |C_p|, |C_q| > 0 \\ \\ 0 & , p = q, \end{cases} \tag{16.3}$$

where f_{kl} represents the flow between objects k and l as defined in Eq. (16.2). With this definition, it is expected that clusters that share similarities between their objects will be assigned closer in the final Layout.

In this way, assuming we have as input a set of objects and a partition for them, the algorithm performs the assignment of the objects in a two-phase procedure. First, for each cluster, the layout of the objects is computed using the values f_{kl} as defined in Eq. (16.2). Second, we perform the computation of the layout of clusters using $f_{C_{pq}}$ as defined in Eq. (16.3).

16.2.3 Proximity Graph Integration

A *proximity graph* $(G(V, E), |V| = n)$ is a graph in which two vertices are connected by an edge when the vertices satisfy particular proximity conditions. Typically, the term *proximity* refers to a *spatial* distance. Although the graphs can be formulated based on different metrics, the most commonly used is the Euclidean distance. Examples of well-known proximity graphs are Minimum Spanning Trees (MST) [35], and k-Nearest Neighbours [36, 37]. The reader is referred to Chap. 4 for a comprehensive revision of proximity graphs.

The QAPgrid algorithm can integrate the information given for a proximity graph in order to indicate that certain adjacency preferences in the final Layout should be enforced. Including this extra information, the definition of Eq. (16.2) can be modified as:

$$f_{kl} = \begin{cases} 0 & , \; k = l \\ \dfrac{M}{s_{kl}}, & \text{if } e_{kl} \in E, \;\; s_{kl} \neq 0 \\ \dfrac{1}{s_{kl}}, & \text{otherwise}, \;\; s_{kl} \neq 0, \end{cases} \tag{16.4}$$

where e_{kl} represents an edge between objects k and l in the proximity graph, and the parameter M controls the flow magnitude between pairs of objects that have an edge in the proximity graph. This modification will lead to objects that will have an even higher flow, imposing the requirement to keep them closer in the final Layout.

16.2.4 Memetic Algorithm for QAPgrid

Memetic Algorithms (MAs) [38] are computational methods which combine Evolutionary Algorithms (EAs) and local search heuristic techniques to efficiently address optimization problems. They were born as a modification of the Genetic Algorithms; hence, it is also possible to find in the literature the name *hybrid genetic algorithm* to denote this class of algorithms. Their central philosophy, which is individual improvement plus population cooperation, comes from the

Algorithm 1: QAPgrid algorithm

```
1 procedure QAPGRID($D, C, G$)
2 begin
3     $Q \leftarrow createInstances(D, C, G)$
                  // $Q_0$: QAP instance for grid layout of clusters
                 // $Q_1, \ldots, Q_c$: QAP instances for each cluster in $C$
4     for $i \leftarrow 1$ to $c$ do
5         $SOL_i \leftarrow MA(Q_i.F, Q_i.D)$
6     end for
7     $createLayout(SOL)$
8 end
```

term *meme* introduced by R. Dawkins to denote the analogue of the gene in the context of cultural evolution [39]. Memetic Algorithms have been successfully applied for solving optimization problems [40, 41]. A particular feature of MAs is that they are intrinsically concerned with exploiting the available knowledge about the problem under study. Exploiting this available knowledge can be done via two elements: (1) incorporating heuristics, local search techniques, approximation algorithms, truncated exact methods, and (2) by using an *adequate* representation of the problem.

In Algorithm 1 we present the QAPgrid algorithm for solving the grid Layout problem. The algorithm receives as input a distance matrix D, a clustering C, and a graph G. In the algorithm, Q represents the set of QAP instances, where $Q_i.F$ and $Q_i.D$ represent the flow and the location matrices of the instance Q_i, respectively. The instance Q_0 corresponds to the instance for the grid layout of clusters. Using the distance matrix, the clustering result, and the graph, function $createInstances(D, C, G)$ creates a set of $c + 1$ QAP instances as explained in Sect. 16.2.2. Then we solve each of the QAP instances using the Memetic Algorithm presented in Algorithm 2. Finally, the function $createLayout(SOL)$ is used to create a unique grid Layout, using the individual layouts of each cluster (SOL_1 to SOL_c) and the layout of clusters (SOL_0).

In the algorithm, a solution is represented as permutation p of size n, where each position stores the location of an object. For example, $p(i) = k$ means that object k is assigned to location i ($1 \leq i \leq m, 1 \leq k \leq n$). The algorithm uses a *ternary tree* to organize *agents* in the population. Every agent contains a list of both *high quality* and *diverse* solutions. The list of solutions in each agent is updated using the procedure *updateAgent*. The ternary tree structure defines overlapped sub-populations. Every sub-population is organized with a *leader* and three *supporters*. With this structure, the best solution of each subpopulation in each generation will be always stored in the leader (using the procedure *udpatePop*). In order to initialize the population, in each agent one solution is randomly generated. Later, the local search procedure *swapTS* improves each starting solution and *updatePop* re-organizes the population. The selection of parent for recombination (*selectParents*) is based on the tree structure. For each agent, the $parent_1$ is selected randomly from

Algorithm 2: Memetic algorithm

```
1  procedure MA($F$, $D$)
2  begin
3      $n \leftarrow dim(F)$                              // Number of objects
4      $m \leftarrow dim(D)$                            // Number of locations
5      $pop \leftarrow initializePop()$
6      $updatePop(pop)$
7      while not $stoppingCriteria(maxGenerations)$ do
8          for $i \leftarrow 1$ to 12 do
9              $selectParents(i, parent_1, parent_2)$
10             $offspring \leftarrow crossover(parent_1, parent_2)$
11             $swapTS(offspring)$
12             $updateAgent(offspring, i)$
13         end for
14         $updatePop(pop)$
15         if $m > n$ then    // do we have more locations than objects?
16             $8 - neighborLS(pop)$
17         end if
18         if $popConverged(pop)$ then
19             $pop \leftarrow restartPop(pop)$
20         end if
21     end while
22 end
```

its list of solutions. Then we choose $parent_2$ according to a measure of diversity of the agent's subpopulation. If there is no loss of diversity, $parent_2$ is selected randomly from the solutions of the leader agent of the subpopulation; otherwise, it is randomly selected from the supporter agents of the sibling subpopulations. A modified version of *cycle crossover* [42] is used as recombination operator which supports more locations than objects. The operator produces an offspring with no additional mutation from the parents, preserving the original information from the parents. Two local search procedures are used in the algorithm. The first one (*swapTS*) is based on Tabu Search [43] which, by using a memory structure, stores a set of movements that are forbidden in a future modification of a solution.

The second local search, called *8-neighbour*, uses the grid structure to compare a current solution with its eight possible assignments (i.e. the eight neighbour positions in the grid). If a movement produces a better solution in terms of fitness, the process is repeated until no further improvements are found. Two methods for keeping a diverse population are implemented. The first one is the restrictions applied for the recombination that we already explained above. The second one is applied when the Memetic Algorithm is unable to find better solutions for a number of generations. In this case, the algorithm restarts the entire population, just keeping the best individual stored in the root agent. Thus, the algorithm generates a new starting point without losing the computational effort for finding the best solution so far. The detailed explanation of the components of the Memetic Algorithm can be consulted in the supplementary material of [31].

16.2.4.1 Memetic Algorithm Settings

In our experiment, we use an ad hoc Memetic Algorithm, of which the structure had been used previously in the literature [44–47]. As we explained, the algorithm uses a ternary tree. In this case, we set our tree with 13 agents in the population. This structure defines four overlapped sub-populations used for the recombination procedure. The algorithm triggers the restarting procedure when the MA has not found any better solution for $\frac{n}{4}$ generations. The maximum number of iterations that the algorithm was allowed to run was 100.

16.3 Case Study: The Marvel Universe

Marvel Comics, originally called Timely Comics Inc., has been publishing comic books for several decades [48]. "The Golden Age of Comics", a name given due to the popularity of the books during the early years, was later followed by a period of decline of interest in superhero stories due to World War II. In 1961, Marvel relaunched its superhero comic books publishing line. This new era started what has been known as the *Marvel Age of Comics*. Characters created during this period such as Spider-Man, the Hulk, the Fantastic Four, and the X-Men, together with those created during the "Golden Age" such as Captain America, are known worldwide and have become cultural icons during the last decades. Latterly, Marvel's characters' popularity has been revitalized even more due to the release of several recent movies which recreate the comic books using spectacular modern special effects. Nowadays, it is possible to access the content of the comic books via the digital platform created by Marvel,[2] where it is possible to subscribe monthly or yearly to get access to the comics.

This network takes place in the artificial Marvel Comics world. Even though this is an artificial network, it was used to compare the characteristics of this universe to real-world collaboration networks, such as the Hollywood network, or the one created by scientists who work together in producing research papers [49]. This dataset gives us the opportunity to present the method in an interesting "human interactions-like" network with 6418 different characters (each one represented by a different node of a weighted undirected graph). These characters appear in 12,942 comic books. In a way, this dataset can be thought of as a social network just in the way real social networks and relations can be downloaded from social media and networking sites such as Facebook and Twitter.

In these networks, the relationships are represented in the form of *followers/followees* (e.g. Twitter) or *friendship* (e.g. Facebook). These real social networks are interesting to marketers and business managers as they can show a network of social influence which could be a valuable asset for a company to, for example, communicate a message, target advertisement, or simply understand their customer base.

[2]http://marvel.com/comics.

In the following sections, we present an application of the QAPgrid algorithm on this Marvel Universe "social network". In particular, we applied the method to the Marvel Universe co-appearance network.

16.3.1 Generating a Dissimilarity Matrix: Helping the Law of Proximity

Our method requires that a certain Dissimilarity Matrix to be computed between each pair of objects that we want to visualize. Our final objective is that the characters that appear together repeatedly will be "closer" in the graph. We thus expect to benefit from the *Law of Proximity* to help the interpretation of the data.

We start by creating a weighted undirected graph. A weighted undirected graph $G(V, E, W)$ contains objects (typically called vertices or nodes) which are connected by edges, where all the edges are bidirectional. In this case, the edges are associated with a numerical value (called weight) which represents extra information about a pair of vertices. Typically, this weight is a non-negative value which can represent costs, distances, probabilities, or flows depending on the application. In our application, the vertices of the graph represent the Marvel characters, and the edges correspond to the co-appearances in the comic books. An edge then is created if two characters co-appear in at least one comic book. The weight of the edge is inversely proportional to the total number of co-appearances (the number of comics that feature both characters). The final graph thus constructed contains a total of 224,106 edges.

The next step is to build a distance matrix using as input this weighted undirected graph. In order to do so, we must compute a distance value between each pair of characters. The distance between two characters represented by two nodes in G is defined as the *shortest path* in G between these two nodes. Other choices are also possible. Intuitively, our choice will ultimately also work well together with the Law of Proximity, as there we can bias the visualization to avoid laying together characters that have no co-occurrences or that are only linked by a path of relatively low number of co-occurrences in the Marvel Universe.

With this information we create the Dissimilarity Matrix $S = \{s_{ij}\}$ which stores the lengths of the shortest paths between each pair of characters i and j

$$s_{ij} = \{cost(shortestPath(v_i, v_j)), \forall\, v_i, v_j \in V, 1 \leq i, j \leq |V|\}. \qquad (16.5)$$

16.3.2 Layout on a Grid and the Law of Good Gestalt

We use this distance matrix to assign each of Marvel's characters to a particular position in a lower dimensional space. According to the Law of Good Gestalt, we have a perceptual bias to form patterns that are either orderly, regular, or "simple" in some sense. Some visualization methods would allow objects to be place in arbitrary positions in the lower dimensional space where they are embedding the objects. Following the Law of Good Gestalt, and its prioritization on regularity over spatial relations, we think a valuable compromise is achieved by assigning objects to grid positions in a two-dimensional space.

16.3.3 Layout of Clusters and the Laws of Proximity and Closure

It is possible to now use the information given by the $D = (s_{i,j})$ and some objective function to try to assign the objects to positions on the 2D grid. In our experience, however, humans would prefer some hierarchical arrangement of the information that still makes use of the Gestalt principles, particularly when the number of objects exceeds a few hundred elements.

Towards this end, we use a clustering algorithm. In this case study, we have used the MST-*k*NN method. This is a graph-based clustering algorithm that, given a Dissimilarity Matrix $D = \{d_{i,j}\}$, constructs a disjoint graph by computing the intersection of the edge sets of two proximity graphs: the Minimum Spanning Tree (MST) and *k*-Nearest Neighbours graph (*k*-NN). The algorithm automatically computes the number of nearest neighbours to consider in each cluster according to the number of vertices in it. The MST-*k*NN has been successfully used and previously presented in [34, 50–52]. Chapter 3 has provided a greater level of detail of the MST-*k*NN method.

According to the Law of Proximity, we will aim at placing elements of the same cluster near to each other. In our visualization results, no two objects that belong to different clusters are going to be closer to each other than the minimum distance between a pair of elements in any cluster. Thus, we will strategically prevent that the Law of Closure could work against us, by perceiving an object that "is not there", such as what can we see by linking elements that belong to different clusters.

The MST-*k*NN algorithm found 69 clusters using the distance matrix that we created as is described in Sect. 16.3.1. Fifty-nine clusters contain less than 100 characters, with the smaller clusters containing only three characters. On the other hand, just ten clusters contain more than 100 Marvel characters. The largest cluster has 2019 characters and the smallest clusters in the collection contain 3, 4, and 5 characters.

16.3.4 The Marvel Characters QAPgrid Layout

The final assignment Layout of Marvel characters is quite a large image in which the main characters (that are of most interest) are hidden amongst all others.

Therefore, to facilitate the visualization, we created an extended version that we called *Organic-QAPgrid* layout. Basically, we separately applied an *organic Layout* which is implemented in the package *yEd*[3] to each cluster. An organic layout is based on the force directed Layout paradigm [53]. In particular, the nodes are considered to be *physical objects* with mutually repulsive forces. Also, the edges follow a physical analogy and are considered to be *metal springs* attached to pairs of nodes. These springs produce repulsive or attractive forces between their endpoints. The Layout simulates these forces and rearranges the nodes such that the sum of both nodes and edges forces reaches a local minimum. The resulting layout can expose the inherent symmetric and clustered structure of a graph, a well-balanced distribution of nodes, and a few edge crossings. The resulting organic Layout of our Marvel grid layout is depicted in Fig. 16.1.

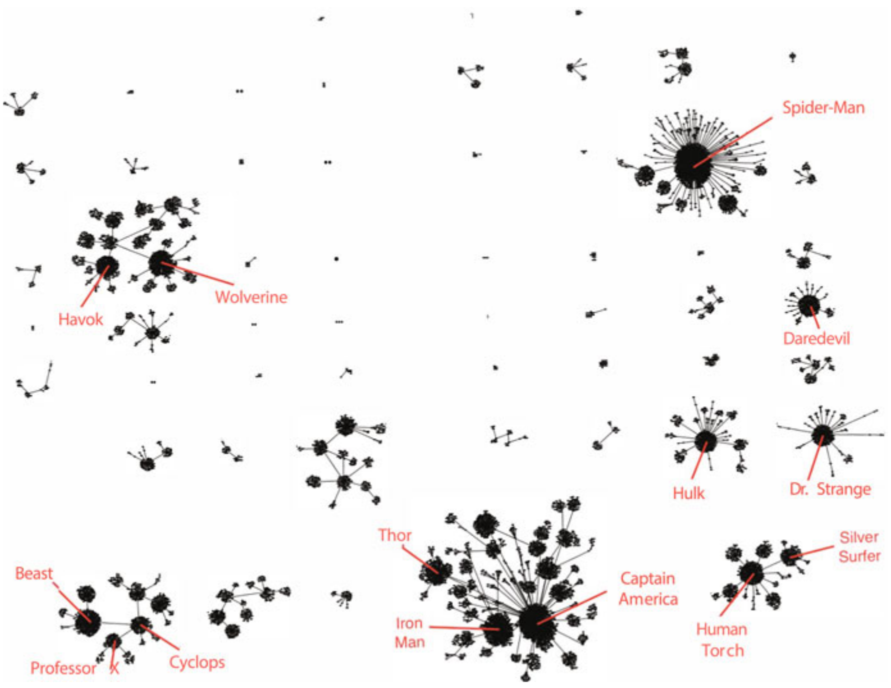


Fig. 16.1 Organic-QAPgrid layout of the Marvel Universe. We include labels in some of the most known characters of series of comic books to characterize the clusters

It is interesting to see how the groups are located by the QAPgrid in different regions of the grid Layout (e.g. top left, top right, and middle bottom). Remember that the algorithm performs the assignment in two phases. First, organizing objects of each cluster and, later, arranging the clusters in the grid. In this case, the

[3]https://www.yworks.com/products/yed.

algorithm can distinguish that the three major characters groups should be located in separated and distant regions. Figure 16.1 shows the Organic-QAPgrid layout of the Marvel co-appearance network. Some of the main characters are highlighted for easier interpretation. The first interesting fact we can notice is that the MST-kNN clustering method can clearly identify the three bigger groups that we can found in the original *ForceAtlas2* Layout.

In the biggest cluster, located in the mid-bottom of the grid, the characters are mainly connected with Iron Man and Captain America. The second largest group (756 characters) situated in the top right region of the grid layout. The central character of this cluster is Spider-Man, which concentrates the collaboration in the group. The third group is located in the top left region, having as the most collaborative characters to Wolverine and Havok. Moreover, in the right region, a few smaller groups were located which have collaboration networks for characters such as Daredevil, Hulk, and Dr. Strange. The algorithm clearly identified *The Fantastic Four* network which was positioned at the bottom right. Thus, some of the members of this network are the four heroes Human Torch, Thing, Invisible Woman, and Mr. Fantastic. Moreover, the group includes to the two biggest enemies Silver Surfer and Dr. Doom. The algorithm found two main groups of X-Men characters (left region). In one group, we can see Wolverine, Havok, and Storm. The other cluster includes Beast, Cyclops, and Professor X. Presumably, this gives us some insights on how the writers gave specific emphasis in stories talking about Wolverine. In fact, Wolverine is second (following only Spider-Man) in the huge amount of stories that are told about a single character in the modern Marvel Universe comic series.

An advantage of this approach is that we can see how every cluster has been positioned in different regions of the grid. For example, it is evident that the collaboration network led by Spider-Man is far from the network with Wolverine and Havok. Another example is the Fantastic Four network which is closer to Captain America and Iron Man group. All in all, the organization of the groups gives some insights to understand the big picture of the Marvel Universe characters, and their interaction through the stories in the comic books. Finally, this is the first time that the QAPgrid algorithm has been used to analyse social networks. As we show here, further applications of the algorithm might deliver novel insights of real-life collaboration and social networks.

16.4 Case Study Two: White Wine Quality Prediction and Visualization

For our second case study in this chapter, we look at products, namely: wine. What is interesting about wine is that it is a highly complex product, particularly when it comes to evaluating wine quality quantitatively, organizing wines by "quality", and subsequently predicting wine quality based on its physical properties. Similarly to the Marvel Universe dataset, visualizing wines may provide some insights that could help inform marketers. For instance, visualizing several wines on a grid that represents quality as well as computationally defined clusters based on their physicochemical properties could help target marketing by identifying niche markets or new product opportunities.

Furthermore, what also makes wine an attractive product to study is that it is one with many different physical aspects and chemical properties that may all be interdependent and collectively affect the product's quality. There has been a growing demand for wine. Its certification and quality assessment process are key aspects for the industry [54]. As Cortez et al. [54] explain, by identifying the most influential factors of quality, the wine-making process might be improved. Not only that, we believe that by investigating existing products (in this case wines), and understanding the underlying factors that could impact quality, new product niches could be uncovered or conversely, "product regions" on the QAPgrid that are not so successful could be identified so that the company could divert or discontinue them.

16.4.1 The Wine Quality Dataset

The Wine Quality Dataset we have used for this second case study is publicly available and can be found through the UCI Machine Learning Repository.[4] This dataset was first generated and published by Cortez et al. [54] who explain that previously, only rather scarce datasets were available that contained *both* quality and physicochemical properties. Cortez et al. [54] actually produced two datasets: red and white wine. In the datasets there are 1599 red, and 4898 white wines and 12 variables. Eleven variables that relate to the physicochemical properties of the wine and one "Quality" variable which is distributed on a Likert Scale from 0 to 7 which has been attributed to each wine by a panel of experts in blind taste tests (the median of the quality scores of each evaluation was attributed to the wine). In this case study, we use the data from the white wines only (as they cannot be studied together as one dataset).

16.4.2 Visualizing Wine on the QAPgrid

Figure 16.2 shows the result of computing the visualization with QAPgrid of the white wine dataset. The algorithm first creates wine groups which are later organized in a two-dimensional Layout. Two regions have been highlighted to indicate the concentration of the bigger clusters. As we mentioned before, wine is a remarkably challenging product when it is evaluated in terms of quality. In fact, MST-*k*NN

[4] http://archive.ics.uci.edu/ml/datasets/Wine+Quality.

struggles to find clearly separable groups. A diverse mixture of wines of different qualities are grouped together by the method as depicted in Fig. 16.4. In order to identify the quality of the wines, we used a colour scale. High-quality wines are identified with dark green, while low-quality ones are denoted with red. The colours are more clearly visual in the zoomed-in Figs. 16.3 and 16.4.

Despite the complexity of the dataset and the mixture of wine qualities found by the method in the clusters, the QAPgrid algorithm can organize similar wines together. For example, the group located to the left of Fig. 16.4 shows a high concentration of both low- and medium-quality wines. However, a small sub-group of high-quality wines are organized together in the bottom of the cluster. The similar pattern can be observed in the cluster located in the centre of the image. High-quality wines are grouped in both left and bottom regions of the cluster.

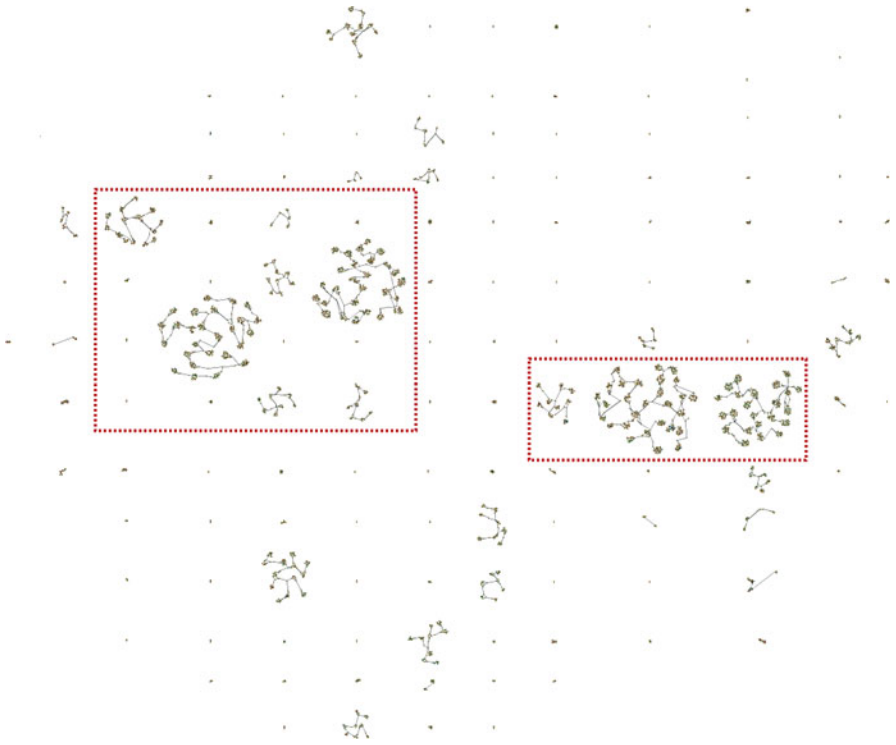


Fig. 16.2 Organic-QAPgrid layout of the wine dataset. In this figure, we highlighted two regions which concentrate the bigger clusters. These bigger clusters are used to perform a detailed analysis. They can be seen in Figs. 16.3 and 16.4, respectively

In this case, the visualization alone cannot be used to obtain meaningful insights. Therefore, we have proposed a specific analysis on each cluster to fill this gap which we detailed in the next sections.

16.4.3 Predicting Quality in the Ten Largest Wine Clusters Through Symbolic Regression

We have used the symbolic regression analysis tool *Eureqa* to predict wine quality as a function of the physicochemical properties available in the dataset. More information on Eureqa can be found in Chap. 5. Eureqa is based on genetic

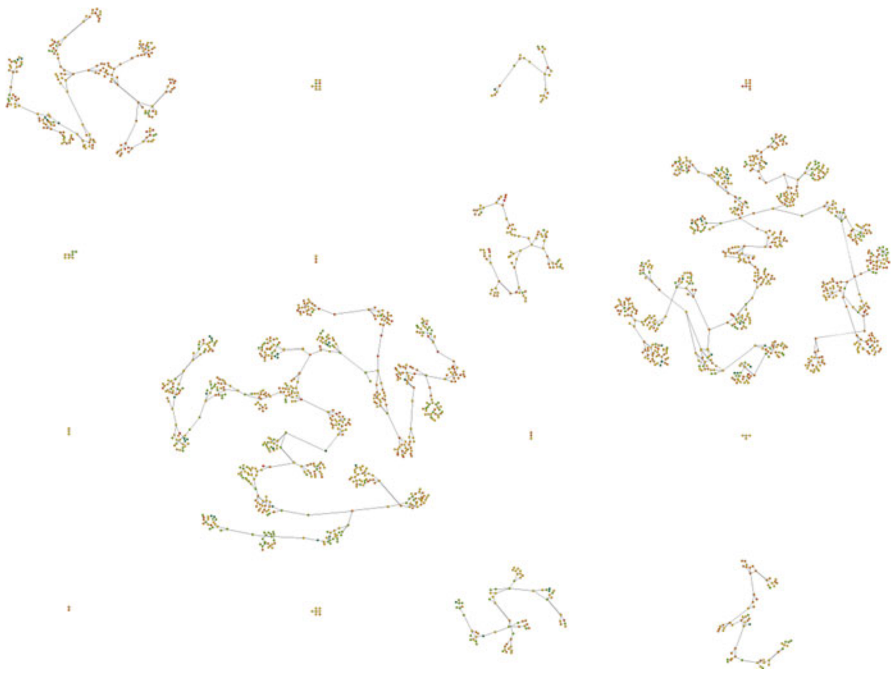


Fig. 16.3 Zoomed-in section from the left region highlighted in Fig. 16.2. These are the largest clusters in the results which is why we use them for further inspection

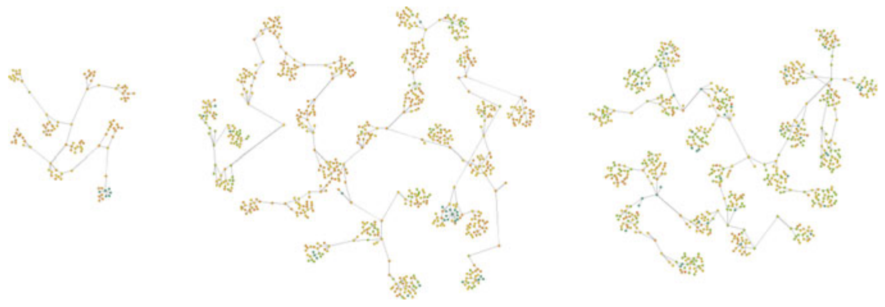


Fig. 16.4 Zoomed-in section from the right region highlighted in Fig. 16.2. Three groups with a variety of wine qualities were found using MST-*k*NN

programming which was introduced in Chap. 1. According to the authors of [54], the variables in this dataset are in fact the most common physicochemical properties to be measured with wine which is why it would be valuable to be able to predict quality using just these variables [54].

In this section, we take the ten largest wine clusters for further inspection. We first look at the problem of identifying a mathematical model that could fit the subjective value of "Quality" that was attributed to each wine as a function of the other physicochemical characteristics. This has been a subject of intense debate amongst wine researchers and data scientists. It is our expectation that there would not be a clear "highly predictive" model of user appreciation because of a variety of motives which include subject variability, experimental design, and the existence of latent variables for which no information has been provided.

Nevertheless, we conducted this experiment as a baseline for comparison of the findings we have when we "segment" the products into different clusters. By employing the ten largest clusters and comparing the variables that enter into the different models, we can have some insights on the physicochemical variations that need to be accounted for to relate to the experts' perceived quality. Equation (16.6) shows the best fitting model found for the whole (unsegmented) dataset that still only uses two variables

$$quality = alcohol - \frac{14.87}{free\ sulphur\ dioxide} - 4.04. \tag{16.6}$$

As it is the best fitting two-variable model, it is likely to be a good starting baseline for further comparisons. The Pearson correlation of this model with the actual quality values is close to 0.7 (0.68597066). We have found this model by evolving the model using the "Squashed Logarithmic Error" which is defined as the logarithm of one plus the Mean Absolute Error. This simple model returns values that are always between ±2 of the value of quality used as input for the training set. A more complex model, however, has also been found:

$$\begin{aligned} quality = 46.12 &+ (0.40 \times pH \times rt\ sulphur\ dioxide) + (0.40 \times alcohol^2) - \\ &(0.12 \times volatile\ acidity \times free\ sulphur\ dioxide) - \\ &(1.43 \times chlorides\ free\ sulphur\ dioxide) - (8.50 \times density \times alcohol) \end{aligned}$$

which although still not been able to narrow down the uncertainty interval on quality as the previous model has a better Pearson correlation, close to 0.8 (0.78756671). Using the variable sensitivity analysis tool, we observe that *rt sulphur dioxide* and pH are the two variables for which there is a 100% likelihood that increasing this variable will increase the target variable (positive effect), while four variables *free sulphur dioxide*, *chlorides*, *volatile acidity*, and *density* have a 100% negative effect (on the target variable: Quality). Interestingly, *alcohol* which had a role of being a point of reference in the first model now shows a split positive (63%) and negative effect (37%), thus indicating that there might be some clusters of wines for which

Table 16.1 This table shows the characteristics of the best fitting models found for each of the largest 10 clusters of the wine dataset

Cluster name	*Size*	Correlation	Fit
Cluster 2	299	0.32	0.76
Cluster 17	158	0.33	0.76
Cluster 3	1977	0.42	0.75
Cluster 31	127	0.69	0.55
Cluster 7	168	0.76	0.41
Cluster 8	473	0.80	0.51
Cluster 1	390	0.84	0.37
Cluster 14	94	0.89	0.20
Cluster 39	69	0.92	0.15
Cluster 32	55	0.94	0.23

The "Size" refers to the number of wines within that cluster, the "Correlation" refers to the Pearson correlation of the model with the actual data, and the "Fit" refers to the error metric

this variable may have distinct associations with quality. Remember that this is the whole, non-clustered dataset of white wines.

After investigating these models found using the whole dataset, we conducted the same process for each of the ten largest clusters. When we conducted an analysis of the relationship between the "fitting" of the best models (and the observed Pearson correlation) with the size of the cluster we did not observe any significant trend. The size of the cluster does not seem to be related to the ability of the software to find a good fitting model. However the results varied with Pearson correlation ranging between 0.32 and 0.94, including five clusters having models with correlations above 0.8 (seven clusters above 0.7). This said, model fitting for each individual clusters, using the same technique as before had led to an increased predictive capacity than what we had before (Table 16.1).

Only the two largest clusters still have a model of the functional form $quality = A * alcohol + B/(free\ sulphur\ dioxide) + C$ (with A, B, and C being some cluster-dependent constants) which was the same functional form shown in Eq. (16.6). This is perhaps not surprising since they are clusters 3 and 8, the largest clusters of them all and therefore likely to be the most representative of the whole dataset. Once again, this model seems to be a baseline when we look at a large group of white wines.

When we look at some clusters in more detail, starting with the two smallest clusters (Cluster 32 and 39), some interesting characteristics come up to light. For instance, in cluster 39 the variable $alcohol$ does not appear in any model and in cluster 32 it appears only once but *free sulphur dioxide* is one of the most frequently appearing variable in all models given by Eureqa for these two clusters. In Cluster 39, *free sulphur dioxide* has mixed positive and negative effects in the model, whereas in Cluster 32 it is completely positive. The variables pH and $density$ do not appear in any model for Cluster 32, while they are present in Cluster 39 as some of the most commonly used variables. Both variables always appear as 100%

Table 16.2 In this table we can see the physicochemical variables that are either present or absent in the best models that fit the user-defined values for quality in the 10 largest clusters of wine in the dataset

Cluster id	3	14	8	2	39	1	17	31	7	32
Variable										
Free sulphur dioxide	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Total sulphur dioxide	✓	✗	✗	✗	✗	✗	✗	✗	✗	✓
Density	✗	✓	✗	✗	✓	✓	✗	✓	✓	✗
pH	✓	✗	✓	✗	✓	✓	✓	✗	✗	✗
Chlorides	✗	✗	✗	✓	✓	✗	✓	✓	✓	✓
Sulphates	✗	✗	✓	✓	✗	✓	✓	✗	✓	✓
Fixed acidity	✓	✗	✗	✓	✓	✗	✓	✓	✓	✓
Citric acid	✗	✓	✓	✓	✗	✓	✓	✓	✓	✓
Residual sugar	✗	✓	✓	✓	✓	✓	✗	✓	✓	✓
Volatile acidity	✓	✓	✓	✗	✓	✓	✓	✓	✓	✓
Alcohol	✓	✓	✓	✓	✗	✓	✓	✓	✓	✓

positive. In contrast, in Cluster 32 *residual sugar* is the second most common variable in the models (with 100% positive effect); however, in Cluster 39 *residual sugar* is one of the least used ones.

Table 16.2 shows which variables are both present and absent in each cluster's predictive model. We use the symbol (✓) to indicate whenever the variable is present in the model of the cluster, and we used (✗) otherwise.

It is clear from Table 16.2 that some variables like *total sulphur dioxide* do not appear in models, while the *free sulphur dioxide* still is present in all clusters' predictive models in the Pareto front. The table also shows the diversity as clusters have divergences in the variables that are used and those not used. For example, the best fitting model for Cluster 39 is shown in the equation below where the constants found by Eureqa are written as alphabetical symbols:

$$\begin{aligned} quality =& A + (B \times \textit{fixed acidity}) + (C \times \textit{free sulphur dioxide}) - \\ & (D / \textit{free sulphur dioxide}) + (E \times \textit{volatile acidity} \times pH) - \\ & (F \times pH) - (G \times \textit{volatile acidity}) - (H \times density) - \\ & (I \times \textit{volatile acidity} \times \textit{free sulphur dioxide}). \end{aligned}$$

We use this model to see how much improvement we can get from running the modelling quality in a specific cluster, as opposed to the whole dataset. We used the best model found using the whole set to observe how it relates to the user-defined Quality for the wines in Cluster 39. Figure 16.5 shows some observed vs. "predicted" plots when we compare the results of the model found in the entire set in Cluster 39 alone, the results of the best model within Cluster 39 as well as the average of the two. We can clearly see the improvement in fitting the user-defined

Quality when we take the average of the best model of the whole dataset together with the best model of cluster 39. We can see that the image on the right depicts the averaged results, showing a trend that is closer to a regressed diagonal line.

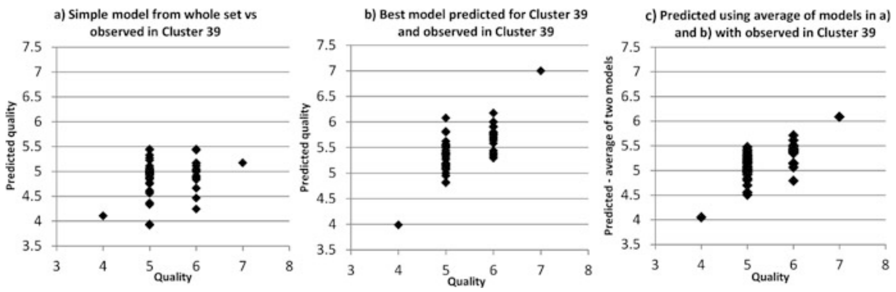


Fig. 16.5 Observed vs. "predicted" plots of the wine dataset predictive models. (**a**) The values obtained via Eq. (16.6) for Quality in Cluster 39 and the user-defined Quality in Cluster 39, (**b**) values from the best model found in Cluster 39 and the user-defined values in Cluster 39, and finally, (**c**) the averages taken from (**a**) and (**b**) against the user-defined Quality values in Cluster 39. We can see that the "spread" of the variability of the results becomes smaller from the images going left to right. Taking a model found from the whole dataset expectedly gives the most variable results, a model found specifically for Cluster 39 does a lot better job, but then interestingly, the variability in the values reduces again when taking the average of the two

16.5 Case Study 3: Investigating Customer Churn

So far we have clustered and visualized a (substitute) social network and products (wine) using the QAPgrid approach. Finally, we will now use this method on a consumer behaviour-based dataset: Customer Churn. This dataset is used in several other chapters in this book and is also featured in the final section on dataset sharing. As stated, this dataset contains 20 attributes (variables) of 3333 customers along with their *churn* information.

The variables include information about consumers and their usage of a Telecom service (e.g. daily charge, number of day or nights minutes used) and whether or not they left (churned) the company. Here, we aim to investigate whether clearly separating clusters appear in this dataset and subsequently, whether they show different characteristics as we "move" across the QAPgrid. Figure 16.6 shows the QAPgrid for the whole Customer Churn dataset. Although, unlike the Marvel Comics universe dataset, we do not have easily identifiable and popular consumers, this QAPgrid may prove useful to business managers of this service company who may want a clear picture of their customers and the spread of their churning behaviours.

As an example of the kind of exploration business managers could do using this visualization, in Fig. 16.7, we show a zoomed in inset of the top right QAPgrid. Here, it is more clear to see which clusters have more churners than others.

As with the wine dataset case study, in service of generating a greater level of insight, we have conducted several symbolic regression experiments in *Eureqa* for some of the clusters found by QAPgrid. Naturally, the variable that we are trying to

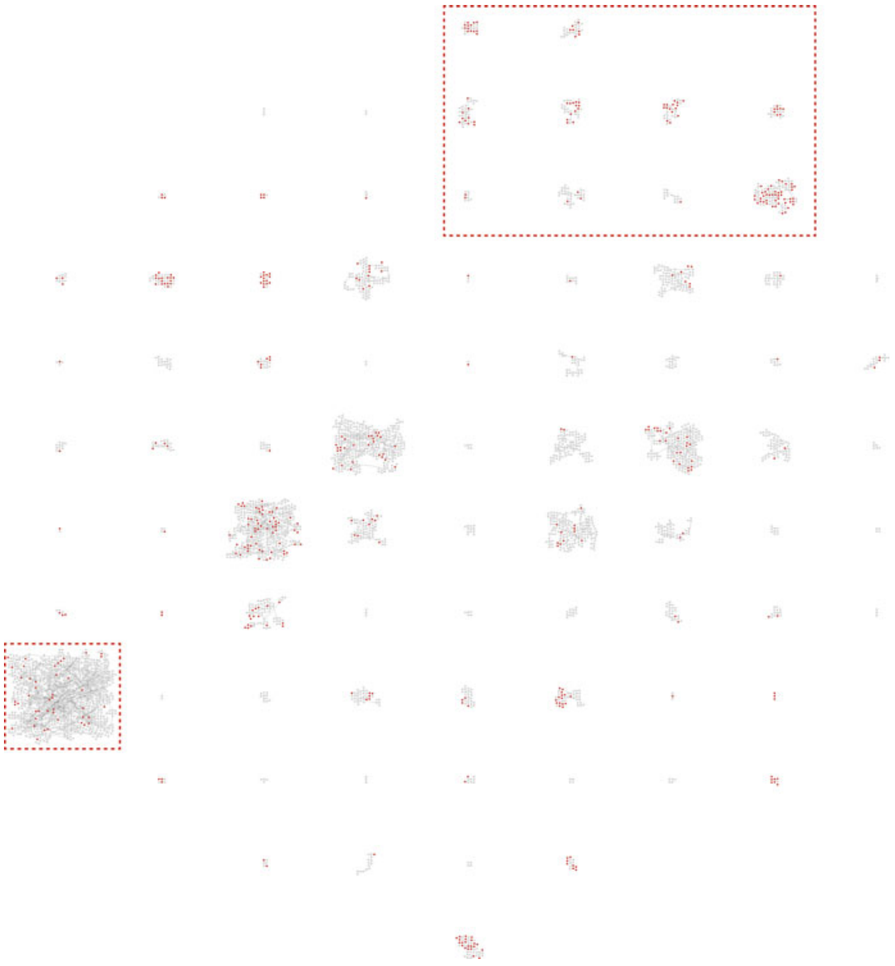


Fig. 16.6 This image shows the QAPgrid of the whole churn dataset. Coloured in red are churners (i.e. consumers who left the company) and for a better visual we have provided a "zoomed view" inside the dashed boxes shown in Figs. 16.7 and 16.4. What the QAPgrid does for this application is organize the clusters of consumers in such a way that we inspect them easily using simply our "naked eye". We can easily identify regions of more red nodes (churners) and those who are not. We selected the inset in a red dashed line in the top right of this image to inspect it a bit further as there are more red nodes in this area

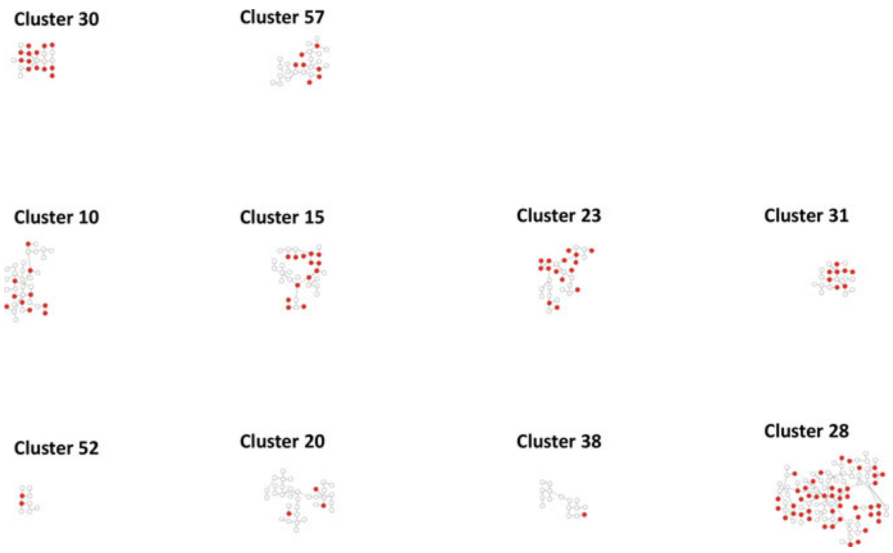


Fig. 16.7 This image shows the top right zoomed in inset of the QAPgrid of the churn dataset. The number of each cluster is shown as some of these are further discussed in the text

predict is Customer Churn, as a function of the other available variables. Rather than simply presenting the complex models that we computed with *Eureqa*, we analyse the most common variables used in each prediction experiment per cluster. These are shown in Table 16.3.

In Table 16.3, we can see that there are some differences per cluster in terms of which types of variables predict churners in that cluster. Although certain clusters still have a varied mixture of variables (e.g. Cluster 30 and 28), others have a very clear overarching type. Cluster 10, for example, has three of its top used variables as evening and night related variables. On the other hand, Cluster 52 has mostly day-related variables (such as calls, charge, or minutes) for predicting churners. Furthermore, we can see that Cluster 23 has almost only international-related variables predicting churn (including *Day Minutes* as well). Finally, some clusters have the variable *Customer Service Calls* included to predict churn, whereas others do not. This brief analysis indicates the wide differences that exist between consumers in different clusters and what may affect them to abandon (churn) the company.

In order to see whether we find a different outcome on the opposite end of the QAPgrid, we also look at Cluster 1, shown in Fig. 16.8 taken from the bottom left of the entire grid. Considering this is a much larger cluster (containing 830 consumers) it is more likely that heterogeneous behaviours exist within this cluster. However, these are the four most commonly used variables by *Eureqa* to predict churners for Cluster 1: *Customer Service Calls*, *Voicemail Message*, *Evening Calls*, and *Night Calls*.

Table 16.3 Most commonly used variables for prediction Customer Churn in selected clusters from the QAPgrid

Cluster	Top 4 used variables in prediction models	Cluster	Top 4 used variables in prediction models
Cluster 30	*Evening Calls*	Cluster 31	*International Calls*
	International Minutes		*Account Length*
	Night Calls		*International Charge*
	Day Minutes		*International Minutes*
Cluster 57	*International Calls*	Cluster 28	*International Calls*
	Day Charge		*Customer Service Calls*
	Night Charge		*International Charge*
	Evening Calls		*International Minutes*
Cluster 10	*Night Charge*	Cluster 52	*Day Calls*
	Evening Calls		*Day Charge*
	Evening Charge		*Day Minutes*
	Customer Service Calls		*International Calls*
Cluster 15	*Evening Minutes*	Cluster 20	*Day Charge*
	Day Minutes		*Customer Service Calls*
	International Minutes		*Evening Charge*
	Night Charge		*Night Calls*
Cluster 23	*International Minutes*	Cluster 38	*Night Calls*
	International Charge		*Account Length*
	International Calls		*Day Minutes*
	Day Minutes		n/a

The first thing that is interesting to note about this is that the variable *Voicemail Plan* was in fact not used once in any of the predictions for the clusters in Fig. 16.7. Furthermore, considering that *Customer Service Calls* here is the top appearing variable in *Eureqa*'s churn predictions is also interesting as it was a lot less common for the previous clusters. What this result indicates is that the QAPgrid algorithm has separated clusters that are possibly highly distinctive from one another, and placed them on opposite ends of the grid. If the managers of this Telecom service were analysing their customer base through this approach, they could potentially highlight those consumers on the grid that generated the most revenue for them (using additional information that was not available in the dataset used in this study) and inspect these clusters of consumers at a greater detail. They would be able to identify in which regions of the grid churn is predicted with variables that fall under their control (e.g. *Customer Service Calls*) or variables they could possibly influence (e.g. whether or not customers are subscribed to a voicemail plan).

Finally, we can see that the QAPgrid approach has been able to separate clusters of different types, and organize them in such a way that can be interpreted by the user with a few extra steps of analysis to provide real business insights.

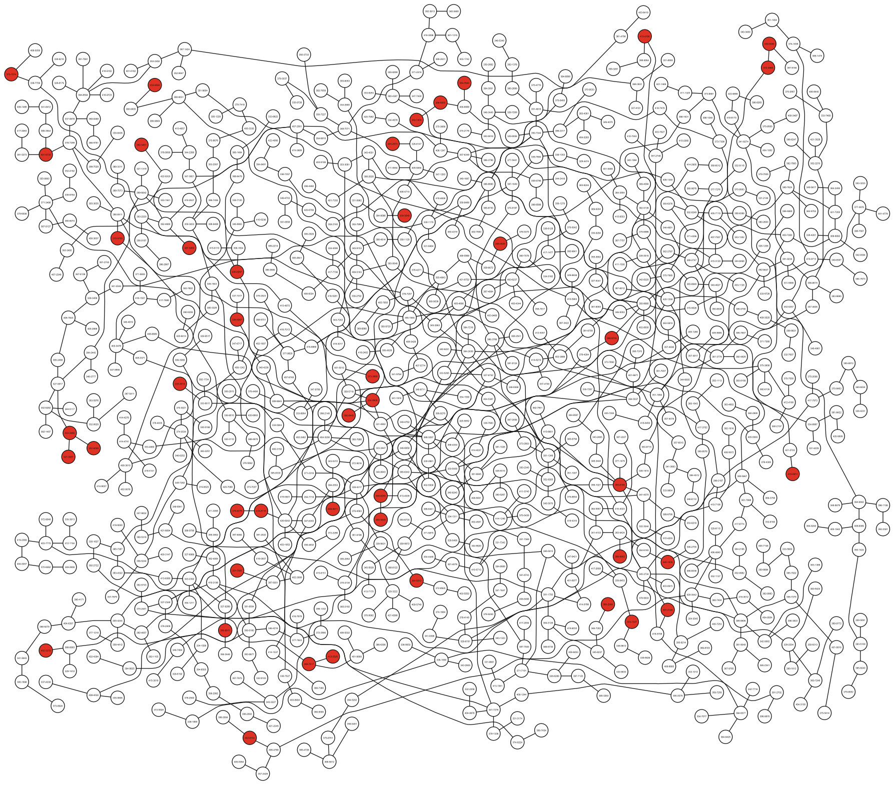


Fig. 16.8 Cluster 1 zoomed in from the bottom left highlighted section of the QAPgrid in Fig. 16.6

16.6 Final Remarks

Current technologies are giving us the unprecedented advantage to collect massive amounts of data about our environment and our behaviour. This means that there are also unprecedented advantages for the business and consumer analytics professionals to better tailor products and services to the final users. At the same time, we increasingly have access to a relatively inexpensive computer power and sophisticated algorithms. Contrary to the popular belief, it is not only web-based technologies that are responsible for this data generation. Several sectors of the economy, for instance, in education, the healthcare industry, finance, and social welfare, are good examples of this "datafication"[5] trend. The list keeps growing with the rise of biometrics and the introduction of personal gear that could track

[5]The "datafication" concept is discussed by Kennet Neil Cukier and Viktor Mayer-Mayer-Schoenberger in "The Rise of Big Data", Foreign Affairs, 2013.

movement and behaviour. Therefore, novel and powerful tools for analysing these and others industries are encouraged.

Visualization is an intuitive method that can reveal our behaviours in an era in which "datafication" shapes our lives. Here, we have presented a visualization algorithm which is based on a classical combinatorial optimization problem (QAP). Based on computed similarities between the objects (characters, products, etc.), and established principles of design that guide mathematical modelling, the computer becomes a powerful tool to generate layouts and place the objects according to meaningful relationships. The data, visualized in a way that is based on perceptual guiding principles, would allow better insights of customer and product spaces. This is a win–win scenario in which companies can provide a better user experience and they benefit from the potential of algorithmics to power their decision-making.

Acknowledgements Pablo Moscato acknowledges previous support from the Australian Research Council Future Fellowship FT120100060 and Australian Research Council Discovery Projects DP120102576 and DP140104183.

References

1. Dirk J. Lehmann and Holger Theisel. Optimal sets of projections of high-dimensional data. *IEEE Trans. Vis. Comput. Graph.*, 22(1):609–618, 2016.
2. Edward R. Tufte. *The Visual Display of Quantitative Information, (Second Edition)*. Graphics Press, 2001.
3. Mitchell Scheiman and Michael W Rouse. *Optometric management of learning-related vision problems*. Elsevier Health Sciences, 2006.
4. Stephen M Kosslyn, Jonathan A Margolis, Anna M Barrett, Emily J Goldknopf, and Philip F Daly. Age differences in imagery abilities. *Child development*, 61(4):995–1010, 1990.
5. Will J Schroeder, Bill Lorensen, and Ken Martin. *The visualization toolkit*. Kitware, 2004.
6. Dejan Todorovic. Gestalt principles. *Scholarpedia*, 3(12):5345, 2008.
7. Agnex Desolneux, Lionel Moisan, and Jean-Michel Morel. *From Gestalt Theory to Image Analysis: A Probabilistic Approach*. Springer, 2008.
8. Yonggang Qi, Jun Guo, Yi Li, Honggang Zhang, Tao Xiang, Yi-Zhe Song, and Zheng-Hua Tan. Perceptual grouping via untangling gestalt principles. In *2013 Visual Communications and Image Processing, VCIP 2013, Kuching, Malaysia, November 17–20, 2013*, pages 1–6. IEEE, 2013.
9. Jin-Gang Yu, Gui-Song Xia, Changxin Gao, and Ashok Samal. A computational model for object-based visual saliency: Spreading attention along gestalt cues. *IEEE Trans. Multimedia*, 18(2):273–286, 2016.
10. Dempsey Chang, Keith V. Nesbitt, and Kevin Wilkins. The gestalt principles of similarity and proximity apply to both the haptic and visual grouping of elements. In Wayne Piekarski and Beryl Plimmer, editors, *User Interfaces 2007, Eighth Australasian User Interface Conference, AUIC 2007, Ballarat, Victoria, Australia, January 30 - February 2, 2007*, volume 64 of *CRPIT*, pages 79–86. Australian Computer Society, 2007.
11. Youn-Kyung Lim, Erik Stolterman, Heekyoung Jung, and Justin Donaldson. Interaction gestalt and the design of aesthetic interactions. In Ilpo Koskinen and Turkka Keinonen, editors, *Proceedings of the 2007 International Conference on Designing Pleasurable Products and Interfaces, 2007, Helsinki, Finland, August 22–25, 2007*, pages 239–254. ACM, 2007.

12. Juan M. Gómez Reynoso and Lorne Olfman. The impact of combining gestalt theories with interface design guidelines in designing user interfaces. In *18th Americas Conference on Information Systems, AMCIS 2012, Seattle, Washington August 9–11, 2012*. Association for Information Systems, 2012.
13. Debaleena Chattopadhyay. Exploring perceptual and motor gestalt in touchless interactions with distant displays. In Bill Verplank, Wendy Ju, Alissa Nicole Antle, Ali Mazalek, and Florian 'Floyd' Mueller, editors, *Proceedings of the Ninth International Conference on Tangible, Embedded, and Embodied Interaction, TEI '15, Stanford, California, USA, January 15–19, 2015*, pages 433–436. ACM, 2015.
14. Birte Möller, Cornelia Brezing, and Dagmar C. Unz. What should a corporate website look like? the influence of gestalt principles and visualisation in website design on the degree of acceptance and recommendation. *Behaviour & IT*, 31(7):739–751, 2012.
15. Krystle Lemon, Edward B. Allen, Jeffrey C. Carver, and Gary L. Bradshaw. An empirical study of the effects of gestalt principles on diagram understandability. In *Proceedings of the First International Symposium on Empirical Software Engineering and Measurement, ESEM 2007, September 20–21, 2007, Madrid, Spain*, pages 156–165. ACM / IEEE Computer Society, 2007.
16. Peifeng Xiang, Xin Yang, and Yuanchun Shi. Web page segmentation based on gestalt theory. In *Proceedings of the 2007 IEEE International Conference on Multimedia and Expo, ICME 2007, July 2–5, 2007, Beijing, China*, pages 2253–2256. IEEE, 2007.
17. Zhen Xu and James Miller. A new webpage classification model based on visual information using gestalt laws of grouping. In Jianyong Wang, Wojciech Cellary, Dingding Wang, Hua Wang, Shu-Ching Chen, Tao Li, and Yanchun Zhang, editors, *Web Information Systems Engineering - WISE 2015 - 16th International Conference, Miami, FL, USA, November 1–3, 2015, Proceedings, Part II*, volume 9419 of *Lecture Notes in Computer Science*, pages 225–232. Springer, 2015.
18. Xin Yang and Yuanchun Shi. Enhanced gestalt theory guided web page segmentation for mobile browsing. In *Proceedings of the 2009 IEEE/WIC/ACM International Conference on Web Intelligence and International Conference on Intelligent Agent Technology - Workshops, Milan, Italy, 15–18 September 2009*, pages 46–49. IEEE Computer Society, 2009.
19. Vasant Dhar, Tomer Geva, Gal Oestreicher-Singer, and Arun Sundararajan. Prediction in economic networks: Using the implicit gestalt in product graphs. In *Proceedings of the International Conference on Information Systems, ICIS 2012, Orlando, Florida, USA, December 16–19, 2012*. Association for Information Systems, 2012.
20. Yuan Ren, Huixuan Tang, and Hui Wei. A Markov random field model for image segmentation based on gestalt laws. In Bao-Liang Lu, Liqing Zhang, and James T. Kwok, editors, *Neural Information Processing - 18th International Conference, ICONIP 2011, Shanghai, China, November 13–17, 2011, Proceedings, Part III*, volume 7064 of *Lecture Notes in Computer Science*, pages 582–591. Springer, 2011.
21. Jeni Paay and Jesper Kjeldskov. A gestalt theoretic perspective on the user experience of location-based services. In Bruce Thomas, editor, *Proceedings of the 2007 Australasian Computer-Human Interaction Conference, OZCHI 2007, Adelaide, Australia, November 28–30, 2007*, volume 251 of *ACM International Conference Proceeding Series*, pages 283–290. ACM, 2007.
22. Robert Fraher and James Boyd-Brent. Gestalt theory, engagement and interaction. In Elizabeth D. Mynatt, Don Schoner, Geraldine Fitzpatrick, Scott E. Hudson, W. Keith Edwards, and Tom Rodden, editors, *Proceedings of the 28th International Conference on Human Factors in Computing Systems, CHI 2010, Extended Abstracts Volume, Atlanta, Georgia, USA, April 10–15, 2010*, pages 3211–3216. ACM, 2010.
23. Guihua Wen, Xingjiang Pan, Lijun Jiang, and Jun Wen. Modeling gestalt laws for classification. In Fuchun Sun, Yingxu Wang, Jianhua Lu, Bo Zhang, Witold Kinsner, and Lotfi A. Zadeh, editors, *Proceedings of the 9th IEEE International Conference on Cognitive Informatics, ICCI 2010, July 7–9, 2010, Beijing, China*, pages 914–918. IEEE Computer Society, 2010.

24. Nicholas F. Polys, Doug A. Bowman, and Chris North. The role of depth and gestalt cues in information-rich virtual environments. *Int. J. Hum.-Comput. Stud.*, 69(1-2):30–51, 2011.
25. Muhammad Hafiz Wan Rosli and Andres Cabrera. Gestalt principles in multimodal data representation. *IEEE Computer Graphics and Applications*, 35(2):80–87, 2015.
26. João Carlos Riccó Plácido da Silva, Luis Carlos Paschoarelli, and José Carlos Plácido da Silva. The importance of using gestalt and grid in building brands. In Constantine Stephanidis, editor, *HCI International 2015 - Posters' Extended Abstracts - International Conference, HCI International 2015, Los Angeles, CA, USA, August 2–7, 2015. Proceedings, Part II*, volume 529 of *Communications in Computer and Information Science*, pages 187–191. Springer, 2015.
27. Amalia I. Rusu, Andrew J. Fabian, Radu Jianu, and Adrian Rusu. Using the gestalt principle of closure to alleviate the edge crossing problem in graph drawings. In Ebad Banissi, Stefan Bertschi, Remo Aslak Burkhard, Urska Cvek, Martin J. Eppler, Camilla Forsell, Georges G. Grinstein, Jimmy Johansson, Sarah Kenderdine, Francis T. Marchese, Carsten Maple, Marjan Trutschl, Muhammad Sarfraz, Liz J. Stuart, Anna Ursyn, and Theodor G. Wyeld, editors, *15th International Conference on Information Visualisation, IV 2011, London, United Kingdom, July 13–15, 2011*, pages 488–493. IEEE Computer Society, 2011.
28. Jie Wu and Liqing Zhang. Gestalt saliency: Salient region detection based on gestalt principles. In *IEEE International Conference on Image Processing, ICIP 2013, Melbourne, Australia, September 15–18, 2013*, pages 181–185. IEEE, 2013.
29. Nadia Ali and David Peebles. The effect of gestalt laws of perceptual organization on the comprehension of three-variable bar and line graphs. *Human Factors*, 55(1):183–203, 2013.
30. Hyunrae Jo, Amitash Ojha, and Minho Lee. A study on region of interest of a selective attention based on gestalt principles. In Minho Lee, Akira Hirose, Zeng-Guang Hou, and Rhee Man Kil, editors, *Neural Information Processing - 20th International Conference, ICONIP 2013, Daegu, Korea, November 3–7, 2013. Proceedings, Part III*, volume 8228 of *Lecture Notes in Computer Science*, pages 41–48. Springer, 2013.
31. Mario Inostroza-Ponta, Regina Berretta, and Pablo Moscato. QAPgrid: A two level QAP-based approach for large-scale data analysis and visualization. *PloS one*, 6(1):e14468, 2011.
32. Tjalling C Koopmans and Martin Beckmann. Assignment problems and the location of economic activities. *Econometrica: journal of the Econometric Society*, pages 53–76, 1957.
33. Rainer E Burkard, Eranda Cela, Panos M Pardalos, and Leonidas S Pitsoulis. The quadratic assignment problem. In *Handbook of combinatorial optimization*, pages 1713–1809. Springer, 1998.
34. Mario Inostroza-Ponta, Alexandre Mendes, Regina Berretta, and Pablo Moscato. An integrated QAP-based approach to visualize patterns of gene expression similarity. In *Progress in Artificial Life*, pages 156–167. Springer, 2007.
35. Jaroslav Nešetřil, Eva Milková, and Helena Nešetřilová. Otakar Boruvka on minimum spanning tree problem translation of both the 1926 papers, comments, history. *Discrete Mathematics*, 233(1):3–36, 2001.
36. Evelyn Fix and Joseph L Hodges Jr. Discriminatory analysis-nonparametric discrimination: consistency properties. Technical report, DTIC Document, 1951.
37. Thomas M Cover and Peter E Hart. Nearest neighbor pattern classification. *Information Theory, IEEE Transactions on*, 13(1):21–27, 1967.
38. Pablo Moscato et al. On evolution, search, optimization, genetic algorithms and martial arts: Towards memetic algorithms. *Caltech concurrent computation program, C3P Report*, 826:1989, 1989.
39. Richard Dawkin. The selfish gene. *Oxford University Press*, 1:976, 1976.
40. Ferrante Neri, Carlos Cotta, and Pablo Moscato. *Handbook of memetic algorithms*, volume 379. Springer, 2012.
41. Ferrante Neri and Carlos Cotta. Memetic algorithms and memetic computing optimization: A literature review. *Swarm and Evolutionary Computation*, 2:1–14, 2012.
42. Peter Merz and Bernd Freisleben. Fitness landscape analysis and memetic algorithms for the quadratic assignment problem. *Evolutionary Computation, IEEE Transactions on*, 4(4):337–352, 2000.

43. Fred Glover and Manuel Laguna. *Tabu Search*. Springer, 2013.
44. Regina Berretta and Luiz Fernando Rodrigues. A memetic algorithm for a multistage capacitated lot-sizing problem. *International Journal of Production Economics*, 87(1):67–81, 2004.
45. Pablo Moscato, Alexandre Mendes, and Regina Berretta. Benchmarking a memetic algorithm for ordering microarray data. *Biosystems*, 88(1):56–75, 2007.
46. Luciana Buriol, Paulo M França, and Pablo Moscato. A new memetic algorithm for the asymmetric traveling salesman problem. *Journal of Heuristics*, 10(5):483–506, 2004.
47. Alexandre Mendes, Carlos Cotta, Vinícius Garcia, Paulo França, and Pablo Moscato. Gene ordering in microarray data using parallel memetic algorithms. In *Parallel Processing, 2005. ICPP 2005 Workshops. International Conference Workshops on*, pages 604–611. IEEE, 2005.
48. Les Daniels. *Marvel: Five fabulous decades of the world's greatest comics*, volume 1. Harry N Abrams Inc, 1991.
49. Ricardo Alberich, Joe Miro-Julia, and Francesc Rosselló. Marvel universe looks almost like a real social network. *arXiv preprint cond-mat/0202174*, 2002.
50. José Marıa González-Barrios and Adolfo J Quiroz. A clustering procedure based on the comparison between the k nearest neighbors graph and the minimal spanning tree. *Statistics & Probability Letters*, 62(1):23–34, 2003.
51. Mario Inostroza-Ponta, Regina Berretta, Alexandre Mendes, and Pablo Moscato. An automatic graph layout procedure to visualize correlated data. In *Artificial Intelligence in Theory and Practice*, pages 179–188. Springer, 2006.
52. Anne Capp, Mario Inostroza-Ponta, Dana Bill, Pablo Moscato, Chi Lai, David Christie, David Lamb, Sandra Turner, David Joseph, John Matthews, et al. Is there more than one proctitis syndrome? a revisitation using data from the TROG 96.01 trial. *Radiotherapy and oncology*, 90(3):400–407, 2009.
53. Stephen G Kobourov. Spring embedders and force directed graph drawing algorithms. *arXiv preprint arXiv:1201.3011*, 2012.
54. Paulo Cortez, António Cerdeira, Fernando Almeida, Telmo Matos, and José Reis. Modeling wine preferences by data mining from physicochemical properties. *Decision Support Systems*, 47(4):547–553, 2009.

Part V
Meta-Analytics

Chapter 17
An Overview of Meta-Analytics: The Promise of Unifying Metaheuristics and Analytics

Fred Glover and Carlos Cotta

Abstract Meta-analytics represents the unification of metaheuristics and analytics, two fields of the foremost interest and practical importance. While metaheuristics provide a modern framework and an arsenal of cutting-edge techniques to handle complex, real-world problems, Analytics embodies the use of prediction and optimization techniques in practical contexts. Thus, their marriage can be regarded as a natural step towards both the creation of effective tools for problems in the Analytics domain and the expansion of the scope of metaheuristic techniques. This introductory chapter describes the advantages obtained by the synergies of the techniques and the avenues for achieving such a unification of methodologies, and discusses some important themes in the field. We also introduce contributions contained in this section, in which these themes are explored in more detail.

Keywords Meta-analytics · Classification · Machine learning · Memetic algorithm · Multi-objective optimization

17.1 Introduction

The Meta-analytics theme of the chapters in this section has its origins in a series of seminal developments in optimization and machine learning and their practical applications. The term "Analytics" has gained unprecedented recognition as a referent for analyses that embody prediction and optimization in a broad sense, typically supported by interpretive aids for users. As a result, organizations from a wide range of disciplines have allied themselves with the Analytics area. This

F. Glover (✉)
Leeds School of Business and the College of Engineering & Applied Science, University of Colorado, Boulder, CO, USA
e-mail: glover@colorado.edu

C. Cotta
ETSI Informática, Universidad De Málaga, Malaga, Spain
e-mail: ccottap@lcc.uma.es

P. Moscato, N. J. de Vries (eds.), *Business and Consumer Analytics: New Ideas*,
https://doi.org/10.1007/978-3-030-06222-4_17

includes researchers and practitioners in classical optimization, notably in the fields of engineering, computer science, operations research and management science. As a compelling example, the prestigious *Institute of Management Science and Operations Research*[1] (INFORMS) has adopted the area of Analytics as a primary focus, and has created a new magazine called *Analytics*.[2]

Metaheuristics [8, 15, 17, 38], by contrast, emerged from the dawning recognition that many real-world problems in business, science and industry are too large or too complex for classical optimization methods to handle effectively. To remedy this problem, recourse was initially made to joining classical optimization with simple heuristic methods, but it soon became clear that more powerful approaches were needed, incorporating various ideas of heuristics but going beyond them. The methods of metaheuristics were conceived to meet this challenge, with innovative evolutionary and neighbourhood search approaches whose first forms appeared in the late 1960s and early 1970s, and which have since undergone substantial refinements and modifications. The name "metaheuristics" itself emerged in the mid-1980s, as the recognition of the essential focus of these new methods became universal.

Meta-analytics represents the, perhaps inescapable, unification of Metaheuristics and Analytics. While the former provides a modern framework and an arsenal of cutting-edge techniques to handle complex, real-world problems, the latter embodies the use of prediction and optimization techniques in practical contexts. Thus, their marriage can be only regarded as a natural step towards both the creation of effective tools for problems in the Analytics domain and the expansion of the scope of metaheuristic techniques. Indeed, modern versions of these latter techniques, such as evolutionary algorithms [13], tabu search [18], simulated annealing [12], swarm intelligence algorithms [26], memetic algorithms [11, 31] and a variety of others which have proved to be highly successful, producing an explosion of publications in international journals and presentations at international conferences. These developments have additionally resulted in the formation of new journals and new societies. The ability to deal with challenging practical problems more effectively, including those from domains that Analytics claims as its focus, lays a foundation for an alliance between Metaheuristics and Analytics. This is notably exemplified by the fact that the Metaheuristics field has made important contributions to predictive and prescriptive analysis, which are prominent concerns of Analytics.

The unification of Metaheuristics and Analytics within Meta-analytics brings about important advantages that were not fully realized in the past as these two fields evolved largely in isolation from each other. Chief among these are the promise of Metaheuristics to become a source of more effective tools for problems in the Analytics domain, and in turn the promise of Analytics to provide a perspective for expanding the scope of algorithmic methods within Metaheuristics. These

[1]https://www.informs.org/.

[2]http://analytics-magazine.org/.

potentials are accentuated by the fact that many researchers and practitioners in Analytics have not been exposed to the Metaheuristics field, and are unaware of its power for addressing practical applications, while many of those working within Metaheuristics have incompletely appreciated the value of incorporating elements that have become the purview of Analytics. The establishment of Meta-analytics creates an opportunity to reach an expanded community of decision makers in industry, science and government who can profit from the union of its component areas.

The scope of this union can be glimpsed by elucidating the primary themes of Meta-analytics and by looking at some previous approaches that have been paving the way. Subsequently, we shall describe the advantages obtained by the synergies of the techniques and the avenues for achieving such a unification of methodologies. We also introduce contributions contained in this section, in which these themes are explored in more detail.

17.2 Themes of Meta-analytics

Six themes broadly constitute the main thrusts of the Meta-analytics area:

1. Using Metaheuristics as a source for creating enhanced predictive and machine learning methods (as in clustering, discrimination, feature detection, pattern recognition and classification, etc.). While such concerns have long been a part of the metaheuristic domain, a more dedicated emphasis on them through Meta-analytics lays a foundation for significant new advances. A key source of contributions derives from highlighting such advances for their general relevance to Analytics, and hence to the Metaheuristics/analytics union.
2. Incorporating predictive and machine learning methods to enhance the performance of metaheuristics. In spite of a variety of proposals for exploiting learning within metaheuristics, e.g., see [7, 16, 25], very little has been done to pursue this theme. Predictive and machine learning procedures can be applied with metaheuristics in offline (pre-solution and post-solution) stages as well as during run time execution. An important step forward will be supplied by refining and implementing meritorious ideas which have been inadequately investigated, and by developing new proposals to capitalize on the opportunities opened up by joining analytics and metaheuristics.
3. Creating special mechanisms and interfaces for interpreting outcomes and relationships uncovered by metaheuristic solution processes. This focus has the goal of enabling users to interact with Meta-analytic procedures to achieve greater insight and yield better decisions. This interaction includes adaptive exploration of model assumptions as well as decision rules for guiding the methods studied [29].
4. Developing integrative methods that capitalize on one or more of the preceding themes to build highly effective algorithms that utilize domain knowledge

for solving problem from important classes. Tabu search [18] and Memetic algorithms [11, 31] are good examples of this theme.

5. Creating improved methods for analysing and explaining the operation of alternative solution approaches, including more effective and comprehensive forms of landscape analysis and quasi-decomposition analysis as embodied in vocabulary building strategies.
6. Establishing a repository of important applications in business, science and government where Meta-analytics provides advances of singular value, leading to improved insights, operations and policies.

17.3 Some Important Research Avenues

In this section we will explore some research lines that intersect with the area of meta-analytics and are playing (or can play in the near future) a major role in the field. We will highlight such connections as well as their relevance for meta-analytics.

17.3.1 Ensemble Learning

An ensemble of learning machines is a set of adaptive entities that deliver partial solutions to a given problem, and then integrate these solutions in some manner to construct a final or complete solution to the original problem. Recent advances in machine learning theory have highlighted the importance of understanding why the collective behaviour of such a collection of several learning agents can perform substantially better than individual ones. Whereas empirical studies on classification methods have shown that some classifiers perform best in some domains but not in all application domains—a phenomenon linked to the *"No Free Lunch" theorem* by Wolpert [42]—ensemble methods can purportedly overcome these limitations to some extent, by combining the output of many independent classifiers. It is perhaps a good analogue with hybrid metaheuristics [36], i.e., integrative or collaborative approaches aiming to combining the virtues of different algorithmic approaches to an optimization problem, so that some sort of "emergent phenomena" comes out of the synergy of the techniques employed by the individual components.

These new learning methods have been called "meta-learning schemes" or "meta-classifiers" or "ensembles". In the machine-learning paradigm, ensemble data mining methods strategically advance the power of committee methods, or combine models to achieve better prediction accuracy than any of the individual models could achieve [32]. The basic goal when designing an ensemble is to develop it in such a way that it provides independent models whose combination will produce better performance than the individual models in isolation. This involves several strategic decisions such as:

- Accounting for a diverse set of outputs: the diversity of the outputs of individual classifiers in an ensemble is a key issue to the generalization performance of the group as a whole. Consequently, a strategically combination of diverse classifiers can help to reduce the total error [34]. This can be attained (1) by combining different classifiers, (2) by having each classifier trained with different subsets of data (i.e., by performing horizontal partitions of the data), (3) by having each classifier trained using different features of the data (i.e., by performing vertical partitions of the data), or (4) any combination of these approaches among other possibilities.
- Designing an appropriate combination rule: The combination rule leads to a final classification from all participating single classifier's outcome and can be mainly designed in two major ways: (1) train the classifiers over the entire feature space and use a fusion method to integrate their outputs (e.g., [weighted] majority voting, summation, product, etc.) and (2) use domain expert classifiers (trained to become an expert in a specific part of the total feature space) and pick one depending on its competence, measured either statically (during training) or dynamically (during prediction).

Ensemble learning is relevant to meta-analytics in several regards. On the one hand, obtaining the best ensemble from a collection of classifiers (i.e., solving the design decisions sketched above) is an NP-hard problem [22], and hence researchers have commonly applied different types of metaheuristic strategies to this end—see, e.g., [14, 21, 28, 44]. On the other hand, they have been routinely applied with success in application domains related to analytics such as customer churn prediction [43], purchasing and marketing [19], predictive analytics [41] and business process management [45], just to cite a few.

17.3.2 Simulation-Based Optimization

The sustained increase of the computational power available for scientific purposes is continuously opening new possibilities for investigating systems that were out of reach not so long ago. Particularly, it allows obtaining increasingly accurate simulations of complex systems and processes from the analytics domain, which can in turn be used to optimize these. The term *simulation-based optimization* precisely refers to these kinds of approaches [1–6, 9, 33, 40].

Metaheuristics constitute the natural tool to tackle the optimization of a simulation system, since black-box optimization is one of the most distinctive realms in which these techniques excel. This does not mean simulation-optimization is a trivial quest though. Simulation models typically exhibit many features that can pose difficulties to the optimization process:

- Non-deterministic behaviour: the simulation may incorporate stochastic elements (if only by the presence of some noise in the output variables), resulting in a different outcome in each run. The optimization algorithm must take this into account in order to handle the resulting uncertainty, e.g. [30, 39]. Some authors have advocated for the term *simeheuristics* to denote approaches that integrate simulation (in any of its variants) into a metaheuristic-driven framework to solve complex stochastic combinatorial optimization problems, see [24].

- Granularity of the simulation: it is possible to attain different tradeoffs between the computational cost of the simulation and the accuracy/uncertainty of the output. Going beyond, it is possible to use machine-learning to create surrogate models to be optimized in lieu of the (more computationally expensive) simulation, e.g., [3, 4, 20, 35].
- Chaotic behaviour: although profoundly different in nature to the issue of noise, the presence of a chaotic regime in the simulated system—e.g., see [27]—may offer similar challenges to the optimization technique in terms of uncertainty.

Some connections between simulation optimization and machine learning methods can be also drawn. For example, in active learning approaches [37] the algorithms are allowed to query an oracle for additional data to infer better statistical models, much like simulation optimization methods may take the choice of sampling at each iteration of the simulation, cf. [1, 6].

17.3.3 Multi-objective Optimization and Analytics

Many problems in the area of analytics are naturally multi-objective: they exhibit multiple cost/benefit functions in partial conflict with each other. Hence, there is often no single optimal solution but a (potentially huge) collection of *non-dominated* or *efficient* solutions providing different optimal tradeoffs between the objectives. This is an issue that has been thoroughly analysed from the point of view of metaheuristics: nowadays, we have a huge arsenal of techniques and methods by which standard (i.e., mono-objective) optimization methods can be augmented in order to tackle many-objective problems, e.g., [10, 23].

However, no matter how successful these approaches can be in fulfilling their mission, it must be noted that the latter is often finding a large and broad sample of the Pareto front. This is useful only as long as there is a sensible decision-making policy whereby an appropriate solution can be extracted from this Pareto front. This can be a hard problem in itself, involving many of the themes mentioned before such as model building, preference learning, handling large collections of data, etc. Thus, this is an area in which much cross-fertilization between meta-analytics applications is possible.

17.4 Introducing the Contents of Chapters in This Section

The chapters in this section make several important contributions to Meta-analytics as instances of the foregoing themes.

Chapter 18, by Mohammad Nazmul Haque and Pablo Moscato, provides a broad overview of the area of ensemble learning, with particular emphasis on its application to analytics. The authors outline the guiding principles for the construction of an ensemble of learners and the issues that ought to be considered. Then, they proceed to overview the deployment of these approaches on four successful application areas: purchasing and marketing, predictive analytics, business process management and customer churn prediction (this latter area is tackled more in depth in a subsequent chapter by Haque et al.) Particular attention is paid to the interface of ensemble methods and metaheuristics, identifying common intersection points and popular methodological approaches. These issues will be further re-elaborated in a subsequent chapter by Thomschke et al.

Chapter 20, authored by Mohammad Nazmul Haque, Natalie Jane de Vries and Pablo Moscato, takes its starting point from the observation that researchers typically use single-objective optimization for the important area of ensemble learning in analytics. The authors go a step farther by introducing a metaheuristic multi-objective evolutionary algorithm, which they apply to the problem of customer churn prediction. The authors additionally investigate a complementary symbolic regression-based approach, noting that the multi-objective approach excels at prediction while the symbolic regression-based approach offers useful tools or business analysts. The results of their study demonstrate how combining their new multi-objective approach with symbolic regression analysis is effective for the paired goals of predicting those customers likely to churn and of providing insight into the types of resources companies can invest in to accurately predict churners and prevent them from churning.

Addressing ensemble learning from a different perspective, Chap. 19, by Ringolf Thomschke, Stefan Voss and Stefan Lessmann, fills a major gap resulting from the fact that previous studies have failed to consider more than a small number of strategies for ensemble member selection. The authors introduce a comprehensive set of metaheuristics to create alternative ensemble classifiers, and carry out an empirical study to compare the outcomes of applying these alternative classifiers. Based on this study, they identify a highly promising modelling approach and compare it to other ensemble regimes and prediction models. They find that their metaheuristic-based ensemble approach improves upon the state-of-the-art, and in the process also introduce a new method to approximate an optimality gap for predictive classification, leading to promising avenues for future research.

Finally, Chap. 21, by Buyang Cao, Cesar Rego and Fred Glover, provides a tailored meta-analytic procedure by developing a new clustering algorithm to address the challenge of analysing data sets of hotel ratings. The study concerns itself with the commonly occurring situation in clustering applications where coordinates are unknown and only distances between objects are available. A tabu search metaheuristic is employed to assure clusters that manifest a cohesiveness property, which the authors join with a new form of hierarchical clustering for additional refinement. Computational experiments demonstrate that the algorithm is both robust and extensible, and is suitable for running on a backend to perform the corresponding tasks with little human intervention. Future research is proposed to exploit the susceptibility of the approach to parallel processing.

Acknowledgements The authors wish to thank Mohammad Nazmul Haque and Pablo Moscato in connection to the subsection on ensemble learning. Carlos Cotta acknowledges support from Spanish Ministry of Economy and Competitiveness and European Regional Development Fund (FEDER) under project EphemeCH (TIN2014-56494-C4-1-P). This research was also supported in part by the Key Laboratory of International Education Cooperation of Guangdong University of Technology.

References

1. Amaran S, Sahinidis NV, Sharda B, Bury SJ (2016) Simulation optimization: a review of algorithms and applications. Annals of Operations Research 240(1):351–380
2. April J, Glover F, Kelly J, Laguna M (2004) The exploding domain of simulation optimization. Newsletter of the INFORMS Computing Society 24(2):1–14
3. April J, Better M, Glover F, Kelly J, Laguna M (2006) Enhancing business process management with simulation-optimization. In: Perrone LF, Wieland FP, Liu J, Lawson BG, Nicol DM, Fujimoto R (eds) 2006 Winter Simulations Conference, IEEE Press
4. Better M, Glover F, Laguna M (2007) Advances in analytics: Integrating dynamic data mining with simulation optimization. IBM Journal of Research and Development 51(3/4):477–487
5. Better M, Glover F, Kochenberger G, Wang H (2008) Simulation optimization: Applications in risk management. International Journal of Information Technology & Decision Making 7(4):571–587
6. Better M, Glover F, Kochenberger G (2015) Simulation optimization to improve decisions under uncertainty. In: Cox T (ed) Breakthroughs in Decision Science and Risk Analysis, Wiley Publishing, pp 59–62
7. Birattari M (2009) Tuning Metaheuristics. A Machine Learning Perspective, Studies in Computational Intelligence, vol 197. Springer, Berlin Heidelberg
8. Blum C, Roli A (2003) Metaheuristics in combinatorial optimization: Overview and conceptual comparison. ACM Computing Surveys 35(3):268–308
9. Chen CH, Lee LH (2010) Stochastic simulation optimization: An optimal computing budget allocation. System engineering and operations research, World Scientific Publishing Company, Singapore
10. Coello Coello C, Lamont G (2004) Applications of Multi-Objective Evolutionary Algorithms. World Scientific, New York
11. Cotta C, Gallardo J, Mathieson L, Moscato P (2016) A contemporary introduction to memetic algorithms. In: Wiley Encyclopedia of Electrical and Electronic Engineering, Wiley, pp 1–15
12. Dekkers A, Aarts E (1991) Global optimization and simulated annealing. Mathematical Programming 50:367–393
13. Eiben AE, Smith JE (2003) Introduction to Evolutionary Computation. Natural Computing Series, Springer-Verlag, Berlin Heidelberg
14. Gaber MM, Bader-El-Den M (2012) Optimisation of Ensemble Classifiers using Genetic Algorithm. In: Graña M, Toro C, Posada J, Howlett RJ, Jain LC (eds) Advances in Knowledge-Based and Intelligent Information and Engineering Systems, IOS Press
15. Glover F (1986) Future paths for integer programming and links to artificial intelligence. Computers and Operations Research 13(5):533–549
16. Glover F, Greenberg H (1989) New approaches for heuristic search: A bilateral linkage with artificial intelligence. European Journal of Operational Research 39(2):119–130

17. Glover F, Kochenberger G (eds) (2003) Handbook of Metaheuristics, International Series in Operations Research and Management Science, vol 57, 1st edn. Kluwer Academics Publishers, Boston MA
18. Glover F, Laguna M (1997) Tabu Search. Kluwer Academic Publishers, Norwell, MA
19. Govindarajan M (2015) Comparative study of ensemble classifiers for direct marketing. Intelligent Decision Technologies 9(2):141–152
20. Han ZH, Zhang KS (2012) Surrogate-based optimization. In: Roeva O (ed) Real-World Applications of Genetic Algorithms, InTech
21. Haque MN, Noman N, Berretta R, Moscato P (2016) Heterogeneous ensemble combination search using genetic algorithm for class imbalanced data classification. PLoS ONE 11(1):e0146,116, https://doi.org/10.1371/journal.pone.0146116
22. Hernández-Lobato D, Martínez-Muñoz G, Suárez A (2006) Pruning in ordered regression bagging ensembles. In: The 2006 IEEE International Joint Conference on Neural Network Proceedings, pp 1266–1273
23. Jaszkiewicz A, Ishibuchi H, Zhang Q (2012) Multiobjective memetic algorithms. In: Neri F, Cotta C, Moscato P (eds) Handbook of Memetic Algorithms, Springer Berlin Heidelberg, Berlin, Heidelberg, pp 201–217
24. Juan AA, Faulin J, Grasman SE, Rabe M, Figueira G (2015) A review of simheuristics: Extending metaheuristics to deal with stochastic combinatorial optimization problems. Operations Research Perspectives 2:62–72
25. Kelly J, Rangaswamy B, Xu J (1996) A scatter-search-based learning algorithm for neural network training. Journal of Heuristics 2(2):129–146
26. Kennedy J, Eberhart R (eds) (2001) Swarm Intelligence. Morgan Kaufmann Publishers, San Francisco, CA, USA
27. Lal DK, Swarup K (2011) Modeling and simulation of chaotic phenomena in electrical power systems. Applied Soft Computing 11(1):103–110
28. Lertampaiporn S, Thammarongtham C, Nukoolkit C, Kaewkamnerdpong B, Ruengjitchatchawalya M (2013) Heterogeneous ensemble approach with discriminative features and modified-SMOTEbagging for pre-miRNA classification. Nucleic acids research 41(1):e21
29. Meignan D, Knust S, Frayret JM, Pesant G, Gaud N (2015) A review and taxonomy of interactive optimization methods in operations research. ACM Trans Interact Intell Syst 5(3):17:1–17:43
30. Mininno E, Neri F (2010) A memetic differential evolution approach in noisy optimization. Memetic Computing 2(2):111–135
31. Neri F, Cotta C, Moscato P (eds) (2012) Handbook of Memetic Algorithms, Studies in Computational Intelligence, vol 379. Springer-Verlag, Berlin Heidelberg
32. Oza NC (2006) Ensemble data mining methods. In: Wang J (ed) Encyclopedia of Data Warehousing and Mining, Idea Group Reference, vol 1, pp 448–453
33. Pasupathy R, Ghosh S (2013) Simulation optimization: A concise overview and implementation guide. Tutorials in Operations Research 10:122–150
34. Polikar R (2006) Ensemble based systems in decision making. Circuits and Systems Magazine, IEEE 6(3):21–45
35. Queipo NV, Haftka RT, Shyy W, Goel T, Vaidyanathan R, Tucher PK (2005) Surrogate-based analysis and optimization. Progress in Aerospace Sciences 41:1–28
36. Raidl GR (2006) A unified view on hybrid metaheuristics. In: Almeida F, Aguilera MJB, Blum C, Moreno-Vega JM, Pérez MP, Roli A, Sampels M (eds) Hybrid Metaheuristics – HM 2006, Springer, Lecture Notes in Computer Science, vol 4030, pp 1–12
37. Settles B (2012) Active Learning. Synthesis Lectures on Artificial Intelligence and Machine Learning, Morgan & Claypool Publishers
38. Sörensen K, Sevaux M, Glover F (2017) A history of metaheuristics. In: Martí R, Pardalos P, Resende M (eds) Handbook of Heuristics, Springer, (available at arXiv:1704.00853 [cs.AI])

39. Tenne Y (2012) Memetic algorithms in the presence of uncertainties. In: Neri F, Cotta C, Moscato P (eds) Handbook of Memetic Algorithms, Springer Berlin Heidelberg, Berlin, Heidelberg, pp 219–237
40. Thengvall B, Glover F, Davino D (2016) Coupling optimization and statistical analysis with simulation models. In: Roeder TMK, Frazier PI, Szechtman R, Zhou E, Huschka T, Chick SE (eds) 2016 Winter Simulation Conference, IEEE Press, pp 545–553
41. Wang L, Wu C (2017) Business failure prediction based on two-stage selective ensemble with manifold learning algorithm and kernel-based fuzzy self-organizing map. Knowl-Based Syst 121:99–110
42. Wolpert DH (1996) The lack of a priori distinctions between learning algorithms. Neural computation 8(7):1341–1390
43. Xiao J, Jiang X, He C, Teng G (2016) Churn prediction in customer relationship management via GMDH-based multiple classifiers ensemble. IEEE Intelligent Systems 31(2):37–44
44. Zhang L, Wang X, Moon WM (2015) PolSAR images classification through GA-based selective ensemble learning. In: Geoscience and Remote Sensing Symposium (IGARSS), 2015 IEEE International, pp 3770–3773
45. Zhao W, Liu H, Dai W, Ma J (2016) An entropy-based clustering ensemble method to support resource allocation in business process management. Knowl Inf Syst 48(2):305–330

Chapter 18
From Ensemble Learning to Meta-Analytics: A Review on Trends in Business Applications

Mohammad Nazmul Haque and Pablo Moscato

Abstract Ensemble learning has been applied in different areas to improve the predictive performances using multiple learners. The two core building blocks, diversity and combination rule, which play a significant role in ensemble learners. The ensemble approach can be divided into two broad groups based on the variation of base classifiers: homogeneous and heterogeneous ensemble. We conducted a comprehensive review of the ensemble learning used for data analytics. The study has proceeded from the feature selection to classification. We found that the ensemble learning helps to overcome the problem associated with the dimensionality and class imbalance of data. For this reason, the ensemble approach found to be more suitable for the classification of high-dimensional data. Then we move towards the meta-analytics using ensemble learning. Our comprehensive review of the metaheuristics-based ensemble learning for homogeneous and heterogeneous ensemble found a substantial number of applications of ensemble learning from these categories. The in detail study of the ensemble learning in business applications able to identify four successful application areas: purchasing and marketing, predictive analytics, business process management, and customer churn prediction. From these application areas, we observe that majority of the approaches built homogeneous ensembles with dynamic selection for single objective optimization. Despite these success in various application domains, ensemble learning could face challenges in analytics in the future. We concluded the chapter with identifying those difficulties and some trends to overcome them for ensemble learning with meta-analytics.

Keywords Classifier ensembles · Ensemble learning · Meta-analytics · Failure prediction · Multi-objective optimization · Group-ensemble learner · Machine learning · Customer churn prediction

M. N. Haque (✉) · P. Moscato
School of Electrical Engineering and Computing, The University of Newcastle, Callaghan, NSW, Australia
e-mail: Mohammad.Haque@newcastle.edu.au; Pablo.Moscato@newcastle.edu.au

P. Moscato, N. J. de Vries (eds.), *Business and Consumer Analytics: New Ideas*,
https://doi.org/10.1007/978-3-030-06222-4_18

18.1 Introduction

Recent advances in machine learning theory have extended the research areas to increase the capabilities of basic learning methods. This has also been the consequence of a philosophical change. Quoting Leo Breitman [12]:

The best solution could be an algorithmic model, or maybe a data model, or maybe a combination. But the trick to being a scientist is to be open to using a wide variety of tools.

who proposed the successful *random forest* approach for classification [10], a "cultural change" has been slowly happening over the last two decades. This change is not only supported by performance, and there is an important scientific quest in understanding why the collective several learning agents can perform substantially better than individual ones. It is perhaps a good analogue with memetic algorithms, a metaheuristic method that aims at combining the virtues of different algorithmic approaches to an optimization problem, so that some sort of "emergent phenomena" comes out of the synergy of the techniques employed by the individual agents.

These new learning methods, the ones that are going to occupy our interest in these pages, have been called "meta-learning schemes" or "meta-classifiers" or "ensembles". In machine-learning paradigm, ensemble data mining methods strategically advance the power of committee methods, or combine models to achieve better prediction accuracy than any of the individual models could obtain [67]. The fundamental goal when designing an ensemble is to develop it in such a way that it will provide independent models. The final aim is that the combined model will produce better performance than the individual models.

Here the inducer is the classifier, applicable for a specific domain. This *"No Free Lunch"* phenomenon presents a dilemma for researchers and practitioners, namely how to select the best classifier for a specific domain. The ensemble learning

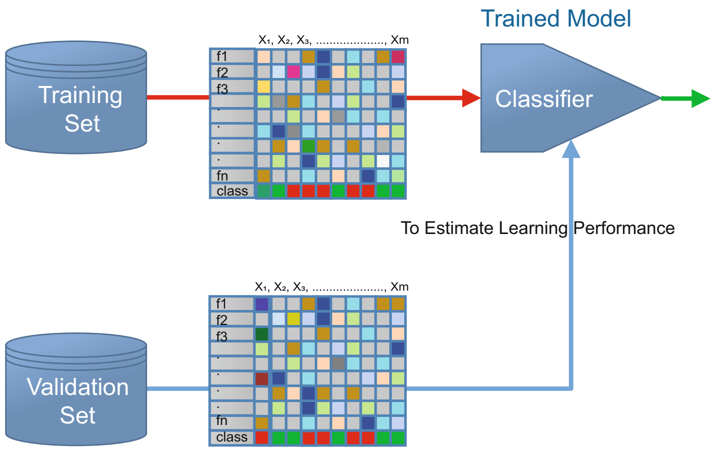

Trained Model
Training Set
Classifier
To Estimate Learning Performance
Validation Set
X₁, X₂, X₃, ..., Xm
f1
f2
f3
fn
class

Fig. 18.1 The learning process of a classifier. The classifier learns from the training data. The validation data is used to estimate the learning performance of the trained model. This trained classifier model is then used for classifying samples from the unknown test set

methods (defined in Definition 18.2) can overcome this dilemma by combining the output of many independent classifiers that perform well in certain domains, but are sub-optimal in others. We can then generally define this type of learning approach as follows:

Definition 18.1 (Learning Task) Given a set of pairs $\{(\mathbf{x_i}, \omega_i)\}$, where $\mathbf{x_i} \in S$ is a sample in a given set S (which could be finite or infinite) and ω_i is a target/class label associated with sample $\mathbf{x_i}$, the task is to find a mathematical model that can "predict" the class label for any other sample in S. The learning process is illustrated in Fig. 18.1.

Definition 18.2 (Ensemble Learning) An ensemble of learning machines is a set of adaptive entities that deliver partial solutions to a given problem, and then integrate these solutions in some manner to construct a final or complete solution to the original problem.

Ensemble methods have been used by researchers from various disciplines such as pattern recognition, statistics, and machine learning. Over the last decade, ensemble-based systems have drawn rising attention, gaining popularity, and spreading the application areas into a broad spectrum. Some promising areas to apply ensemble methods are: business, financial, biomedical, remote sensing, genomic, incremental learning, and other types of rapidly growing data analytics problems. A typical ensemble method for classification tasks contains the following building blocks [72]:

- **Training set:** It is a labelled dataset used for training the ensemble.
- **Validation set:** Another independent set of samples and their associated labels which used to estimate how well the classifiers perform after being trained. This set helps you to estimate properties of the models.
- **Test set:** The final dataset of samples and their labels which is used to estimate the performance of the ensemble.
- **Base Classifier:** The base classifier C is an induction algorithm to form a generalized relationship among sample $\mathbf{x}$ and its corresponding label ω. In many cases we have a discrete set of possible class labels (e.g. $\Omega = \{\omega_1, \cdots, \omega_c\}$).
- **Diversity Generator:** We aim at obtaining several independent models of the mapping between the samples and the class labels, so an algorithm is generally needed for generating the diverse models from individual learners after being trained with labelled training data even when the same technique is used for all of them.
- **Combiner:** Another algorithm (also called the *fusion method*) is responsible for integrating the outputs of a set of base classifiers such that the whole set would deliver stronger generalization ability than any individual classifier.

There are very little rules to "guide" how this process should be conducted. In practice, perhaps the simplest approach for ensemble learning is to "weight" the outputs of several individual models, and then to combine them in such a way that the obtained accuracy is maximized. This is a case where "meta-analytic" techniques can be used, guiding the process of combining the information provided by the independent classifiers. The next subsection clarifies the issues related to the design of such an ensemble.

18.1.1 Basic Design Techniques for Ensemble of Classifiers

Though, every ensemble method combines multiple classifiers outcome into a single decision, the building paradigms of the ensemble of classifiers usually differ from each other. They generally have different diversity generation mechanisms and in the strategy for combining them. The process involved in the ensemble learning is illustrated in Fig. 18.2. We will briefly discuss the basic algorithmic blocks.

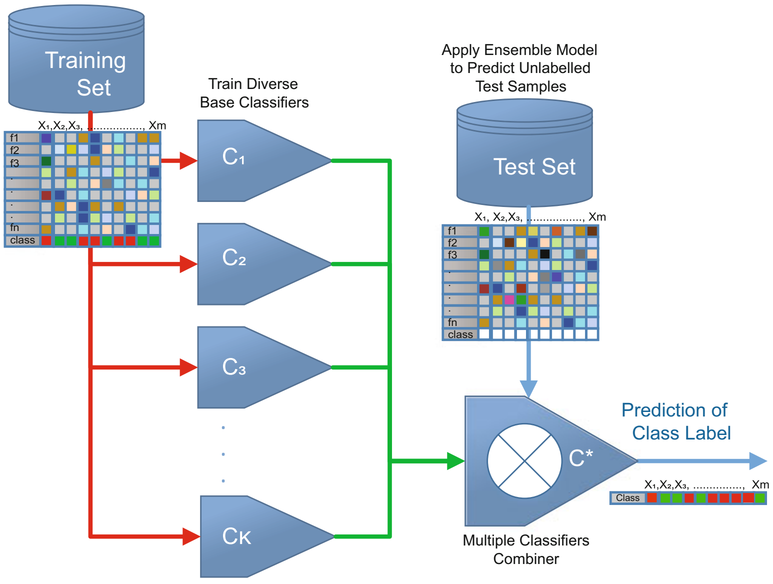

Training Set
X₁,X₂,X₃, ..., Xm
f1
f2
f3
fn
class
Train Diverse Base Classifiers
C₁
C₂
C₃
Cκ
Apply Ensemble Model to Predict Unlabelled Test Samples
Test Set
X₁, X₂,X₃, ..., Xm
f1
f2
f3
fn
class
C*
Multiple Classifiers Combiner
Prediction of Class Label
X₁,X₂,X₃, ..., Xm
Class

Fig. 18.2 A generic ensemble learning process. Diverse base classifiers are trained on training data. The combination rule (also known as combiner method, fusion approach) combines the learning of multiple base classifiers to create the ensemble. The ensemble is then used for predicting the class label of the samples in the test data

18.1.1.1 Diversity in Ensemble of Classifiers

In a way somewhat analogous to the need of low correlation of features for improving classification accuracy, it is also the case that this is not to say that we do not want them to be highly correlated with the correct output, but it is clearly the case that some variation of outputs is expected and that individual classifiers can predict the wrong output for some samples. The diversity can be achieved in many ways. Based on how the ensemble employs the training set, we can observe three sources of diversity.

- **Different Classifier:** To attain the diversity among base classifiers, this method is used to train all single classifiers using same training set. They use different learning algorithms as base classifiers, or their variations of parameters of the base classifiers. This is known as the *heterogeneous ensemble method.*
- **Different Training Set:** Another diversity generation mechanism can result from using different training sets (obtained from the original training set by re-sampling techniques, such as bagging and AdaBoost) to train base classifiers. These approaches use multiple instances of a base classifier trained on different training samples. This is called *homogeneous ensemble method.*
- **Ensemble Feature Selection:** Diversity can also be generated by training the individual classifiers with datasets that consist of different feature subsets, like the Random Subspace Method (RSM). In the ensemble feature selection, an ensemble method is used to draw feature subspace. Base classifiers are then trained using those sub-samples of features.

Ensemble methods can use any one of the alternatives mentioned above to generate diverse individual base classifiers. To determine the final decision, the process of combining the outcomes comes next in the process.

18.1.1.2 Designing the Combination Rule

The combination rule is the second key component of the ensemble method. It helps to reach into a final decision from all participating single classifier's outcome. The combination rule can be mainly designed in two ways:

- **Classifier Fusion:** In this case, each classifier is trained over the entire feature space. Then, results of individual classifiers are combined in an appropriate manner to reach a final decision. The fusion method can use majority voting, weighted majority voting, summation, product, maximum and minimum, fuzzy integral, the Dempster–Shafer based combiner, or the decision templates to decide the final outcome.
- **Classifier Selection:** This design approach for a combination method uses domain expert classifiers in the training phase. Here, each classifier is trained to become an expert in a specific part of the total feature space. In the decision prediction stage, only one base classifier is used, instead of combining the decision of all available classifiers. The selection of a final predictor classifier, among available expert single classifiers, can be done in two manners. They are the *Static Classifier Selection (SCS)* method and the *Dynamic Classifier Selection (DCS)* method.

- *Static Classifier Selection (SCS):* In the SCS method, the region of competence for a single classifier is defined during the training period. Therefore, one classifier is chosen for prediction task.
- *Dynamic Classifier Selection (DCS):* On the other hand, the regions of competence of participating single classifiers are defined during the prediction phase, based on the characteristics of the testing dataset.

18.2 Review on the Ensemble Learning for Data Analytics

In this section, we focus on the selection of important features (i.e. feature selection) and in the classification methods aimed at achieving better accuracy using ensemble methods. The canonical concept of ensemble approaches is to improve the prediction accuracy by using a set of individual classifiers. The underlying assumption is that distinct base classifiers in an ensemble "bring together" different patterns observed in the data. It is then the combination of these patterns that helps to better predict the class label. We review the literature that deals with the enhancement of classification accuracy by the adoption of an ensemble-based approach. We will also review the relevant literature where we have observed that metaheuristics have been employed to create or guide the ensemble. This review helps to explain the effectiveness of the ensemble method data analytics and in particular for feature selection and effective classification.

Researchers have published several reviews and empirical analysis on the ensemble methods since date. Most of them discussed and tried to explain why the ensemble of classifiers (EoC) outperforms the results of the best of its single classifier component. For instance, Jordan and Jacobs [37] proposed a statistical hierarchical tree-structured mixture model of classifiers utilizing ideas from mixture model estimation and generalized linear model theory. They presented an Expectation Maximization (EM) algorithm for adjusting the parameters of the architecture using an iterative approach to maximize the supervised learning performance. They have not emphasized important issues (such as convergence and consistency of the final result) in the paper. Dietterich [18] evaluated ensemble methods named Bayesian averaging, bagging, and boosting. The article had addressed three principal reasons that explain why the ensemble methods outperform any single classifier. They identified that when the amount of available training data is too small compared to testing data, then classifiers can lead to an "under-fitting" problem. On the other hand, classifiers that only use a local search based heuristic may, quite obviously of course, get stuck in local optima. They provide experimental evidence for choosing AdaBoost algorithm for ensemble creation. However, the study did not examine the interaction between AdaBoost and the properties of the underlying learning algorithm. Valentini and Masulli [82] presented a brief overview of ensemble methods. Moreover, they proposed a taxonomy based on the combination rule for the base classifiers in ensembles. After that, Polikar [69] examined the context in which the accuracy of an ensemble method is more useful than their individual classifier algorithms. They also analysed the various components of an ensemble system as well as various procedures by which individual predictions could be combined. Recently Krawczyk et al. [45] proposed an ensemble of fuzzy classifiers and used a GA for selecting samples for the undersampling approach. They applied a metaheuristic-based approach to select samples for balancing medical image dataset, and they have improved the classification performances.

Roli et al. [73] proposed two design methods based on the so-called *"overproduce" and "choose"* paradigm with metaheuristics. In this scheme, *overproduce* is the generation of a large set of candidate ensemble classifiers. Then *choose* is the extraction of the best performing sub-ensemble. They have used three heuristic search algorithm named forward search, backward search, and tabu search as their chosen methods. Even though these design techniques demonstrated some compelling features, they do not claim any best choice method among three and do not show how to design the optimal ensemble. Similarly, Kotsiantis et al. [43] described various classification algorithms and the recent effort for improving classification accuracy by using ensembles of classifiers. They have discussed some algorithms based on Artificial Intelligence (Logic-based techniques, Perceptron-based techniques) and Statistics (Bayesian Networks, Instance-based techniques). The book by Kuncheva [48] was entirely devoted to the field of the model combination. The book covers the topics of multiple binary classifier systems and their base classifier combination methods. Meanwhile Tulyakov et al. [81] proposed an ensemble of classifiers method to increase the performance of pattern recognition applications. They introduced a retraining operation which trained the optimal ensemble classifier with another set of training data and adjusts associated weight. They have also used local neighbourhood search like k-NN to identify similar subsamples of the test set to be used for the retraining of the classifier combinations. Such effects showed a significant effect on the performance of combinations. Oza and Tumer [68] reviewed ensemble methods for diverse classes of statistical classification algorithms and their various real-world application domains. They surveyed applications of ensemble methods to problems that have historically been most representative of the difficulties in classification. In particular, the survey covers the applications of ensemble methods to remote sensing, person recognition, one-vs-all recognition, and medicine. Recently, Galar et al. [26] developed an empirical analysis of different aggregations to combine outputs of binarization strategies for the dataset. They formed a dual study: firstly, they employed different base classifiers to monitor the suitability and capability of each combination within each classifier. Then, compared the performance of these ensemble techniques with the classifiers' themselves. The proposed research tested several well-known algorithms such as Support Vector Machines, Decision Trees, Instance Based Learning, or Rule-based Systems. Experimental evidence supported that the goodness of the binarization techniques about the base classifiers were most robust methods within this framework.

Collectively, these research papers showed that ensemble methods are a promising approach for use on incremental learning, data fusion, feature selection, learning with missing features, and multi-class classification domains. They also provided answers to the question: *"Why should we choose ensemble methods instead of* *improving individual classification performance?"* They have justified that ensemble of classifiers is an efficient method for improving classification accuracy.

18.2.1 Ensemble Learning in Feature Selection

Feature selection method is an essential step in classification. It plays a key role in classification performance enhancement. Literature suggests two main categories of feature selection procedure named *ranking* and *subset selection* method. Ranking feature selection methods applied each feature's worth to order them and select top *k* number of features. On the other hand, feature subset selection process uses the searching method to find a better subset of features that explains original feature most.

18.2.1.1 Population-Based Approaches

Many approaches are based on Genetic Algorithms (GAs), a population-based heuristics, to manage search space created from the ensemble of feature selection method. Kuncheva and Jain [46] proposed two basic ways to utilize the power of GAs to create a multiple-classifier system. Their method started with a GA version that selects disjoint feature subsets and the second variant selects overlapping feature subsets. They created the ensemble with three-classifier systems and basic types of individual classifiers. Oliveira et al. [65] applied GA to combine the predictions of different feature selection methods. They proposed an ensemble feature selection method based on a hierarchical multiobjective GA. In their first level, a set of robust classifiers applied for feature selection. Next level of GA combines the features to build a powerful ensemble. The proposed method is applied for handwritten digit recognition, using three different feature sets and neural networks (MLP) as classifiers. Experimental result showed the power of the proposed strategy. Later on, Minaei-Bidgoli et al. [59] applied GAs on feature selection to improve the prediction performance. First, they used feature selection method for decreasing computational cost and increasing classifier efficiency. The feature selection optimized the prediction accuracy of ensemble using GA. Their approach was more efficient than feature subset selection method, and also adaptable to analyse different attributes. Subsequently, Oliveira et al. [64] also dealt with feature selection by using GAs for a classification ensemble building in the similar manner of Oliveira et al. [65]. They experimented on handwritten digit recognition problem and compared with traditional Bagging and Boosting. The proposed method produced higher prediction accuracy. Recently, Ebrahimpour and Eftekhari [21] proposed a fuzzy-based CFS score with an ensemble of feature ranking algorithms. The experimental result revealed that their approach is more suitable than metaheuristics for extremely large search space.

18.2.1.2 Reduction of Dimensionality

On the other hand, many researchers used different types of feature selection methods to create an ensemble to reduce the dimensionality problem in the dataset. Cunningham and Carney [15] argue that feature subset-selection has emerged as a useful technique for creating diversity in classification ensembles. They proposed an entropy measure of the outputs of the ensemble members as a useful measure of the ensemble diversity, and they evaluated it on a medical prediction problem. Experimental results showed enhanced prediction performance on the ensemble and the entropy measure of diversity. Additionally, it expressed a relationship with the change in diversity and breadth of the ensemble. Similarly, Blanco et al. [6] proposed an ensemble of wrapper-based method for gene selection and classification in gene expression datasets. They employed the Naïve-Bayes classification algorithm in a wrapper form. Their design reduces the number of genes that have been selected with a similar accuracy than other conventional approaches. Nikulin et al. [62] created an ensemble that is capable of greater prediction accuracy than any of their individual members by means of utilizing selected features. They found a large number of relatively small and balanced subsets where representatives from the larger pattern are to be selected randomly. They have tested the ensemble strategy against datasets of the PAKDD-2007 data-mining competition. Their ensemble method produced higher accuracy than single classifiers. Recently Seijo-Pardo et al. [75] experimented on different ensemble configuration for feature selection. The ensemble combined the features selected from seven feature ranking algorithms to increase the stability of the selection process. Their experimental results suggested the usage of aggregated feature subset of the ensemble with Random Forest (RF) classifier for data classification.

18.2.1.3 Class Imbalance

Several other works in the literature have reported the issue of class imbalances in the dataset and have proposed alternative methods to deal with the problem. Duangsoithong and Windeatt [20] presented a bootstrap feature selection for ensemble classifiers. They selected optimal features from the full dataset before bootstrapping on selected data. Then they have applied ensemble classifier to evaluate the performance on UCI machine learning repository and a causal discovery datasets. The results showed that a bootstrap feature selection algorithm provides slightly better accuracy than traditional feature selection for ensemble classifiers. Akbaş et al. [2] has implemented different ensemble methods for feature selection for classification of a colon cancer dataset. They have evaluated the performance improvement of individuals with ensemble classification methods. Their results showed that the classification accuracy for the colon cancer can be increased by reducing the features using the ensemble methods. Recently, Yang et al. [93] proposed an ensemble-based wrapper method (feature selection) which is suitable for highly imbalanced class distribution. First, they have eliminated the imbalanced nature of the dataset by sampling and creating multiple stable subsets of actual data. Then, they have evaluated feature subsets using an ensemble of base classifiers each trained on a balanced dataset. Their test results indicate that ensemble-based feature selection by wrapper method outperformed original wrapper algorithms for imbalanced class data.

According to result from these papers, we can conclude that feature selection method can be a potential approach for resolving problems in the dataset. Imbalanced class data and dimensionality are two verdicts for classification algorithms, which can be eliminated using feature selection methods. Ensemble methods can lead to better selection of features from a large number of features and can circumvent problems related to imbalanced class distributions.

18.2.2 Ensemble Learning in Classification

A popular method for creating a perfect classifier from a set of training data is to build several classifiers, and then to combine their predictions for achieving better accuracy. Researchers have employed different types of ensembles of classifier methods. We begin this section of literature survey with embedded feature selection in classifiers then popular ensemble classifiers including tree-based structures.

Tsymbal et al. [80] created an ensemble consisting of multiple classifiers constructed by randomly selecting feature subsets. They conducted experiments on a set of 21 real-world and simulated datasets. In many cases, the ensembles have higher accuracy than the single classifier model. Namsrai et al. [61] introduce an ensemble classification method for classifying cardiac arrhythmia disease. They have applied feature subset selection process to obtain a number of feature subsets from the original dataset. Then they have built classification models using each feature subset and combined using a voting approach. They have considered both classification error rate and feature selection rate (which means the frequency of a specific feature in feature subsets) to calculate the score of the each classifier in the ensemble. The ensemble method improves the classification accuracy significantly. In contrast, Srimani and Koti [76] conducted a classification analysis on five medical datasets, and their results show that individual classifiers perform better than some cases than an ensemble. They have concluded with comments that only selected classifiers needed to be used for each dataset and to achieve optimal accuracy with the medical dataset; researchers need to choose the classifier accurately. They had selected default options for classifiers and did not use any optimization techniques to find the optimal solution for ensemble classifiers. This was missing in their conducted research.

There are some researches on creating the ensemble of classifiers only using tree based structures. In this regard, Zhang et al. [99] proposed an ensemble classification system based on diversified multiple trees which addressed the uncertainty of microarray data quality problem. The proposed diversified multiple decision tree algorithms (DMDT) can determine most informative features from abundant features. Then, they use this unique diversity value as an ensemble combination function. The test results showed that the DMDT is more accurate on average than other well-known ensemble of classifiers on microarray datasets. Likewise, Hu et al. [35] proposed maximally diversified multiple decision tree algorithm (MDMT) based on an ensemble method for robust microarray classification. They are concerned with the noise susceptibility of different decision tree algorithms and MDMT for microarray dataset classification. The experimental results showed that ensemble decision tree methods tolerate the noise values better than a single tree does. They recommended decision tree based ensemble classifiers to deal with noisy microarray datasets. Similarly, Osareh and Shadgar [66] proposed an ensemble method combining Rotation Forest and AdaBoost techniques which in turn preserve the accuracy and diversity in microarray datasets. They have applied five different feature selection algorithms to determine a concise subset of informative genes. Then they have applied decision tree based ensemble classifier using selected features as training set. The experiment showed that the proposed ensemble classifiers outperform not only the conventional machine learning classifiers but also the classifiers generated by two widely used ensemble learning methods, that is, bagging and boosting. Previously, Galar et al. [27] tackled the classifier learning problem with datasets that suffer from imbalanced class distributions and provide empirical comparisons based on ensemble taxonomy. They reviewed the state of the art on ensemble techniques in the framework of imbalanced binary datasets. Their results show empirically that ensemble-based algorithms are advantageous since they outperform the mere use of pre-processing techniques before learning the classifiers. Koutanaei et al. [44] proposed hybrid ensembles for feature selection and classification for credit scoring datasets. They evaluated three feature selection method using the performances of SVM classifiers. Afterwards they used ensemble of classifiers using neural networks for final classifier creation and revealed better performances with the feature subset selected by PCA approach.

These works of literature advocate that ensemble of classifiers are more effective methods than other combinations of classifiers like boosting and bagging. It is also possible to bring inherent power from embedded feature selection methods for imbalanced data classification, noise susceptibility, and multi-class classifier. In a word, the ensemble of classifiers methods is suitable enough to deal with high-dimensional biological datasets.

18.3 Ensemble Learning with Metaheuristics: Moving Towards Meta-Analytics

A current subject of intense research in data classification is the combination of several classifier systems. They can be built following either the same or different models and/or datasets building approaches. A large number of available single classifiers can create an enormous search space for ensemble combinations. To get the better combination for creating an ensemble from this NP-hard problem [34], researchers applied different types of metaheuristics strategies. In this extent, we found that the population-based metaheuristics have been applied in mainly three levels of the ensemble of classifiers (EOC) method. The application level of metaheuristics in the EOC can be categorized as: *decision combination level, feature selection level*, and *classifier formation* or *creation level*.

Firstly, we will discuss the literature where metaheuristic has been employed in the base classifier creation, next to the feature selection and finally as the combination methods.

Zhang and Bhattacharyya [97] demonstrated the potential of a genetic programming (GP) metaheuristic as a base classifier algorithm in building ensembles in the context of large-scale data classification. An ensemble built upon base classifiers trained with GP significantly outperformed its counterparts built upon individual base classifiers. Wang and Wang [84] proposed an ensemble of classifiers where each individual classifier is trained on a particular weighting over the training examples. They incorporated a genetic algorithm (GA) to search a substantial weighting space. They tested the algorithm on the UCI benchmark datasets and discovered the ensemble method as robust and consistent for face detection application. After a while, Ranawana and Palade [71] presented a current overview of Multi-Classifier Systems (MCSs) and provided an outline roadmap for MCS. They also presented a case-study of the MCS theoretical issues, and simple guidelines for the selection of different paradigms. Moreover, they introduced a novel optimization of the traditional majority voting combination method which uses a genetic algorithm. A couple of years later, Kim and Oh [40] proposed a hybrid genetic algorithm (a type of memetic algorithm) for classifier ensemble selection. They used two local search operations to improve offspring prior to replacement. They parameterized the local search operations to control the computation time and found it as an effective approach. Later on, Espejo et al. [31] survey existing literature about the importance of genetic programming for classification. They have shown the different ways in which evolutionary algorithm can help in the creation of accurate and reliable classifiers. Haque et al. [32] proposed to use GAs to find the best combination of ensembles from a large pool of base classifiers. The ensemble learning with the metaheuristic (population-based search) approach was able to find better ensembles than the state-of-the-art ensemble learner for image and biological data classification.

As explained in Sect. 18.1.1.1, we can classify the ensemble technique in three categories based on the diversity generation mechanism. Moreover, based on the variations of base classifiers used, the ensemble of classifiers can be categorized into *homogeneous* (as shown in Fig. 18.3) and *heterogeneous* (shown in Fig. 18.4) ensemble method. The feature selection ensemble can be embedded into both of them as the diversity generation methods. Now we will review works that fall into those two categories and that use population-based metaheuristics search algorithms.

18.3.1 *Homogeneous Ensemble Learning*

Kim et al. [39] proposed a meta-evolutionary (ME) based homogeneous ensemble of classifier method. It uses two-level evolutionary search through the feature selection and ensemble creation. The author applied an ME for feature selection using an ensemble method which works like Boosting algorithm. The fitness of the ensemble is updated with the cost based on its predictive accuracy, as determined by majority voting with equal weight among base classifiers. They used a variable number of ANNs (Artificial Neural Network) to form the ensemble and used majority voting as their combination function. Experimental results on 15 datasets suggested that the weighted ensemble is more effective than single classifiers and classic ensemble methods. They have tested their method on smaller datasets (maximum dimensions in the dataset were 3772 features, 27 samples), it needs to verify the suitability of their approach on large datasets.

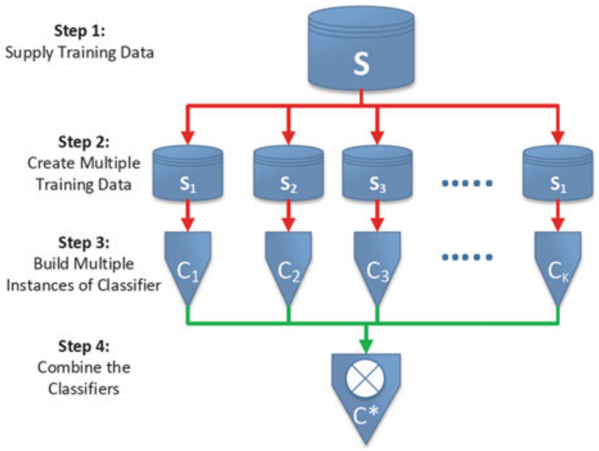


Fig. 18.3 The structure of homogeneous ensemble

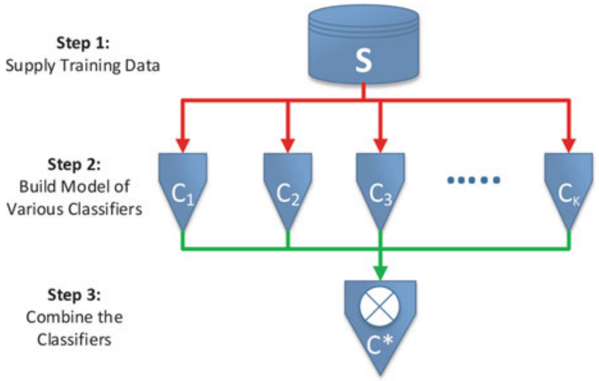


Fig. 18.4 The structure of heterogeneous ensemble

Cleofas et al. [14] proposed a GA-based ensemble to deal with the imbalance class problem in their datasets. They have applied the GA on feature selection level for balancing the class distribution by under-sampling the majority class samples. They used a m-dimensional chromosome, which represents all features in the training set of the m samples. Then they applied features selected by GA to the ensemble of classifiers. This method is impractical to apply on high-dimensional datasets due to encoding all features into the chromosome.

Gaber and Bader-El-Den [24] proposed a genetic algorithm based random forest (GARF) ensemble method. They used heuristic chromosome (HC) where an individual represents an Ensemble (Random Forest), and each gene represents a Random Tree. They stored pre-computed classification results for all random trees. Then, they evolve new ensemble using pre-generated trees (EV-Ensemble) to evaluate the fitness of new random forest using the HC. They have also applied an Indirect-GA to create random forest classifier. Their experiment showed that the performance of random forest could be boosted when using the genetic algorithm.

More recently, Oh and Gray [63] proposed a GA Robust Ensemble (GA-RE), which used the GA in the classifier formation level. It used variable sized individuals in the range of 5–20 decision stumps trees. But the stump tree they used, it contained only two terminal nodes which possibly loss informative features. Instead of using decision stumps, they could try using decision trees. Zhang et al. [96] proposed a homogeneous ensemble of classifier selected by a GA approach for image data classification. They select the homogeneous base classifiers created by varying the parameters using GA to experiment with SVM and NN based homogeneous ensemble of classifiers approach.

18.3.2 Heterogeneous Ensemble Learning

Heterogeneous ensemble method uses different types of classifiers to create an ensemble combination. Some researcher uses single instance from each of the different classifiers and other uses multiple instances of some classifier algorithms. Genetic algorithms have been applied on feature selection, classifier selection, and combination level of the heterogeneous ensemble.

In [25], the authors have proposed an interesting individual structure of their GA. They have represented each individual using a 3-dimensional incidence matrix. In the incidence matrix, one dimension represents features, another dimension represents classifiers, and the other dimension represents combination methods. They also have proposed associative genetic operators to work with the individual structure. Unfortunately, the whole potential of this idea is still to be tested as the authors have applied their method only on tiny dataset with six (6) features. The approach may not be entirely suitable for the high-dimensional dataset, because it encodes all feature in each individual.

Xu and He [92] proposed a genetic algorithm based ensemble classifier. The GA has been applied on both feature selection and decision combiner level simultaneously. They have applied it to a real-world multi-sensor dataset. They have randomly selected features for each base classifier and used very few numbers of features to build the classifier. There is scope for enhancing this system by adopting an appropriate feature selection method. However, the method has outperformed feature level and decision level fusion methods, it needs to be tested on larger datasets.

After that, Thammasiri and Meesad [79] proposed a GA-based classifier ensemble method to select the appropriate classifiers. They use majority vote in order to increase ensemble classification accuracy. Their evaluation showed that the proposed ensemble classification models selected by the GA yield the highest performance and are efficient in building the ensemble. The maximum feature count for the applied datasets was 30.

Later on, Lertampaiporn et al. [50] applied heterogeneous ensemble methods using GAs for feature selection. They have used Support Vector Machines (SVMs), k-Nearest Neighbour (k-NN), and Random Forests (RFs) to create the ensemble and they applied on large biological datasets. They have applied SMOTE-bagging methods to balance the dataset and used correlation-based feature (CFS) selection method with GA search method. Their experiment shows better performance than single classifiers. More recently, Haque et al. [33] proposed another population-based search metaheuristic, named Differential Evolution (DE) algorithm, for optimizing the weights of heterogeneous base classifiers for the weighted voting in ensemble learning. This approach has performed better in predicting heart disease in compared with other ensemble learning methods.

18.3.3 A Partial Summary Based on Three Streams

We can provide a partial summary of the studies in this subsection based on the application of the metaheuristics in different algorithmic components of the ensemble learning framework. We have observed that some methods applied the metaheuristics to find the combination rule that could enhance the generalization performance (e.g. [79]). There are some studies, which used it for feature selection level. They try to optimize the feature subset to gain better classification performances for base classifiers using various feature subsets (e.g. [14, 39, 50, 79]). The third category of ensemble learning uses metaheuristics for creating the classifier (e.g. [24, 39, 63]). The authors used several tree-based classifiers to create the ensemble. The GA has been applied here to form an optimal classifier using the trained tree-based models. The application of GA in the classifier formation level is common for homogeneous ensemble creation. There exists some study which applied the genetic algorithm to find best base classifiers for the classifier selection models (e.g. [42, 74]). They used only one base classifier selected by the associated metaheuristics. Searching the ensemble combination space for a large pool of base classifiers using metaheuristic was proposed in [32, 33], where the combinatorial space of the base classifiers is explored.

18.4 Ensemble Learning in Business Analytics

The successful invasion of ensemble learning in life science, engineering, management has inspired the business domain as well. We could broadly divide the applied areas of ensemble learning for business and consumer analytics into *marketing, finance, business process management, credit scoring, consumer behaviour, customer relation management*, and *churn prediction*. The key characteristics of ensemble learning used in business analytics for the last decade are presented in Table 18.1. We represent the information in chronological order so that it is easy to see the evolution of the techniques in a number of fronts. The following subsections organize the discussion by application area and summarize the types of business applications in a word cloud[1] of Fig. 18.5.

18.4.1 Purchasing and Marketing

The process of market analytics uses the purchase history information or other behavioural characteristics to find the most valuable customers from their responses to marketing campaign questions or data from market research. Meta-analytics along with metaheuristics has proven its success in this arena for quite a long time. Blaszczynski et al. [7] applied ensemble of weak learners to predict customer behaviour for internet users. Govindarajan [30] compared heterogeneous ensemble learning (created with via the Bagging method [11]) with Radial Basis Functions

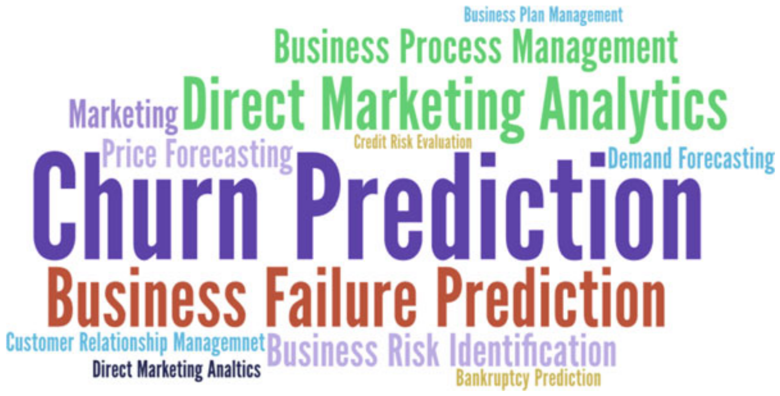

Business Plan Management
Business Process Management
Marketing
Direct Marketing Analytics
Price Forecasting
Credit Risk Evaluation
Demand Forecasting
Churn Prediction
Business Failure Prediction
Customer Relationship Managemnet
Business Risk Identification
Direct Marketing Analtics
Bankruptcy Prediction

Fig. 18.5 Word cloud view of the types of business analytics used ensemble learning in the last 10 years (from 2007 to 2017). The most frequent application has the largest size, and the least frequent application has the smallest size in the word cloud

[1]We created the Word Clouds using the Pro Word Cloud add-ins for Microsoft Word 2016 found at: https://store.office.com/en-us/app.aspx?assetid=WA104038830.

Table 18.1 A survey of ensemble learning methodologies and applications in business for the last 10 years (from 2007 to 2017 sorted by year)

Application area and paper	Ensemble technique	Key characteristics
Customer Relationship Management [49]	Homogeneous ensemble learning	Used SVM ensembles to effectively manage customer relationship from a risk avoidance perspective by identifying high risk customers
Business risk identification [58]	Neural network and rule-based ensembles	Applied the ensemble classifiers to predict the potential defaults for a set of personal loan accounts
Business risk identification [94]	Metaheuristics-based ensemble learner	Applied classifiers to generate knowledge and aggregated into an ensemble output using an evolutionary programming (EP) technique
Price forecasting [95]	Homogeneous ensemble	Proposed an empirical mode decomposition (EMD)-based neural network ensemble learning model for modelling and forecasting the spot price of world crude oil
Direct marketing analytics [38]	Homogeneous ensemble learning	Compared two fundamental ways of developing ensemble models—sampling and feature selection using homogeneous types of ensemble
Direct marketing analytics [98]	Group-ensemble learner	Proposed a 3-level ranking model (Group-Ensemble) to solve crucial data mining problems like data imbalance, missing value, and time-variant distribution for marketing data analytics
Churn prediction [8]	Ensemble learning	Proposed an ensemble classification models using probability estimation trees (PETs) with weighted voting fusion based on lift measure
Churn prediction [9]	Rotation-based ensemble classifiers	Rotation Forests, feature extraction is applied to feature subsets in order to rotate the input data for training base classifiers, while RotBoost combines Rotation Forest with AdaBoost
Demand forecasting [56]	Homogeneous ensemble of regression	Proposed a balanced-sampling-based ensemble of support vector regression (SVR) forecasting method to improve the predictive accuracy in demand forecasting for supply chain management
Business failure prediction [53]	Multiple models combination ensemble	Formulated an ensemble learner by incorporating the feature selection methods in the representation level, a hybrid of principal component analysis (PCA) at the modelling level, and weighted majority voting at the fusion level
Business failure prediction [54]	Multiple models combination ensemble	Formulated an ensemble learner using case-based reasoning (CBR) predictor models with feature-bagging as diversity generation approach. The CBR ensemble for multiple case representations used majority voting for producing final prediction
Churn prediction [87]	Cost-sensitive ensemble learning	Proposed an ensemble learning with dynamic classifier selection of cost-sensitive learning from imbalanced data

(continued)

Table 18.1 (continued)

Application area and paper	Ensemble technique	Key characteristics
Marketing [87]	Homogeneous ensemble learning	Comparison of performances in customer choice modelling is done with three ensemble learning approach: “Bagging”, “Boosting”, and “MultiBoosting” evaluated with bias–variance decomposition
Business process management [23]	Ensemble-based clustering method	Proposed a clustering method for discovering performance-oriented process models where separate prediction models were built on different subsamples of the data
Business plan management [19]	Metaheuristics-based ensemble learner	The genetic programming-based ensemble of classifiers is compared against an intelligent model of business plans’ appraisal system. Subject matter experts were satisfied with the Reliability and the accuracy of the selected plan
Business failure prediction [55]	Two-stage ensemble	Four feature spaces are created to build different models using multivariate discriminant analysis (MDA) and logit. Afterwards, two levels of ensembles are implemented: one with each of the same type of models and another with two ensembles. Final decision made using majority voting fusion approach
Churn prediction [88]	Ensemble learning with dynamic classifier selection	Train classifiers on different sub-sample of the data using resampling technique to balance the class distribution. Then, dynamically select a proper classifier ensemble for each test sample
Churn prediction [1]	Homogeneous ensemble learning	Compared the performance of four popular ensemble methods: “Bagging”, “Boosting”, “Stacking”, and “Voting”. Four known base classifiers are used: C4.5 Decision Tree (DT), Artificial Neural Network (ANN), Support Vector Machine (SVM), and Reduced Incremental Pruning to Produce Error Reduction (RIPPER)
Direct marketing analytics [3]	Ensemble of classifiers	Used an ensemble classification method which removes imbalance in the data, using a combination of clustering and under-sampling
Direct marketing analytics [30]	Bagging classifier	Designed both homogeneous and heterogeneous ensembles using RBF and SVM as base classifiers. Bagging of RBF exhibited the best performance on direct marketing data
Churn prediction [89]	Dynamic classifier selection ensembles	Proposed a feature-selection-based dynamic transfer ensemble (FSDTE) model to introduce transfer learning theory for utilizing the customer data. Iterative feature generation was used for training the base classifier. Finally, base classifier was selected dynamically

(continued)

Table 18.1 (continued)

Application area and paper	Ensemble technique	Key characteristics
Churn prediction [5]	Ensemble selection	Proposed a decision-centric framework to create churn models using heterogeneous base classifiers. Then, pairwise combinations of base classifiers performances were assessed to select the best classifier for decision-making
Business process manage-ment [100]	Clustering ensemble	Proposed an entropy-based clustering ensemble method for mining different types of preference patterns with a priority-based schedule algorithm for dynamic resource allocation in multi-instance process contexts
Churn prediction [90]	Multiple classifiers ensemble	First base classifier models were generated by using different subset of the data. Then, final decisions were combined based on weighted voting and supplied threshold
Churn prediction [4]	Heterogeneous ensemble selection	Proposed a heterogeneous ensemble method where the selection of best base classifier subset was done using soft set based method to combine the decisions for final result
Business process management [16]	Multi-tier ensemble learning	Proposed a multi-view ensemble learning approach with a clustering-based trace abstraction method and a context- and probability-aware stacking method as decision fusion
Business failure prediction [83]	Heuristic-based ensemble selection	Proposed a two-stage selective ensemble model with manifold learning algorithm, feature selection and ensemble selection were used to predict business failure
Bankruptcy prediction [22]	Metaheuristic-based ensemble learning	Proposed a two-stage ensemble learning (with SVMs) for feature selection and classification using several metaheuristics (including artificial bee colony (ABC), genetic algorithm (GA), and sequential forward selection (SFS)) to predict bankruptcy
Credit risk evaluation [17]	Hybrid Homogeneous ensemble learning	Proposed a feature selection-based hybrid-bagging algorithm (FS-HB) for improved credit risk evaluation

(RBF) and Support Vector Machine (SVM) classifiers. They found that the RBF bagging approach performed better on analysis of direct marketing data. Kim [38] compared two fundamental ways of developing ensemble models (using sampling and feature selection) for homogeneous types of ensemble. Different types of homogeneous ensemble learners are created with Artificial Neural Network (ANN), Naive Bayes (NB), and SVM as base classifiers and applied on vehicle insurance policy direct marketing data. Their experimental outcome suggested the ensembles with Naive Bayes (NB) and SVM classifiers marginally improved performance. Similar approach has been used in [85] for profiling household and customer targeting from existing customer database. Zhang et al. [98] proposed a 3-level ranking model for ensemble learner with Bagging, RankBoost, and Expending Regression Tree to solve crucial data mining problems (data imbalance, missing value, and time-variant distribution) for design marketing strategies. Amini et al. [3] used ensemble learner to increase the prediction accuracy and improve the response rate for the bank's direct marketing.

18.4.2 Predictive Analytics

Forecasting plays a major role in making the business decision. The ability to predict the demand or forecast the price, identify the risks, and predict the failure at an early stage helps businesses to be less prone to failure. A homogeneous ensemble with empirical mode decomposition (EMD)-based neural network was proposed by Yu et al. [95] to forecast the spot price of world crude oil. Liu et al. [56] proposed a homogeneous ensemble learning method using SVMs as base regression method to forecast the customer demand in supply chain management. These forecasting will help the business to act in advance to gain more. Risk identification in business contributes towards a successful business plan and to reduce future loss. Matthews and Scheurmann [58] proposed an artificial neural network method and rule-based ensembles to predict the potential defaults for a set of personal loan accounts. Yu et al. [94] applied ensemble classifiers to generate knowledge and aggregated into an ensemble output using an evolutionary programming (EP) technique to identify corporate financial risk. The identification of risk is a crucial step which influences the process of business plan management. In this arena, Li et al. [55] proposed two-stage ensemble using four feature selection and two classifiers (multivariate discriminant analysis (MDA) and logit). The final decision was made using majority voting fusion approach for the business failure prediction. Wang and Wu [83] also used similar two-stage ensembles for classifier model and feature selection, but with heuristics, to predict the business failure.

18.4.3 Business Process Management

Business process management (BPM) is a systematic approach to redesign the workflow of an organization to make it more efficient, effective, and adaptive. The BPM requires analysis of the existing business to identify candidate workflow for improvement and is a good area for analytics. Ensemble learning techniques are showing a very strong footprint in this regard. Folino et al. [23] proposed an ensemble learning-based clustering method for discovering performance-oriented process models to help the BPM. Zhao et al. [100] also proposed a clustering ensemble, but using entropy measure for mining different types of preference patterns and dynamic resource allocation in multi-instance process contexts. Subsequently, Cuzzocrea et al. [16] proposed multi-tier ensemble learning approach with a clustering-based trace abstraction method and a context- and probability-aware stacking method as decision fusion for identifying the deviations between business processes.

18.4.4 Customer Churn Prediction

Customer churn prediction plays a significant role in deciding the company's Customer Relationship Management (CRM) strategy. Among all the business applications using ensemble learning, churn prediction standouts for many applications in compared with others. In [8], they proposed an ensemble classification models using probability estimation trees (PETs) with a weighted voting fusion based on lift measure for customer churn prediction. Bock and den Poel [9] proposed rotation-based ensemble classifiers named RotBoost which combines Rotation Forest with AdaBoost in customer churn prediction. Xiao et al. [87] introduced a cost-sensitive ensemble learning with dynamic classifier selection from imbalanced data. They applied the method in popular Germany credit scoring and a branch of Sichuan Telecommunication data. The experimental result analysis suggested the usage of both the accuracy and diversity of the ensemble simultaneously in the process of selection to improve the classification performances. Xiao et al. [88] proposed ensemble learning with dynamic classifier selection and resampling technique to balance the class distribution in churn data. Abbasimehr et al. [1] used four different homogeneous ensemble learning formulated with C4.5 Decision Tree (DT), Artificial Neural Network (ANN), Support Vector Machine (SVM), and Reduced Incremental Pruning to Produce Error Reduction (RIPPER) as a base classifier in each setting. In their experiments, boosting ensemble learning approach formulated with any of the four base classifiers performed better than other approaches in churn prediction. Xiao et al. [89] proposed dynamic classifier selection Ensembles and iterative feature generation. Baumann et al. [5] also proposed ensemble selection, with a different approach, by considering pairwise combinations of base classifiers performances for churn prediction. Xiao et al. [90] proposed Multiple Classifiers Ensemble based on weighted voting and supplied threshold. Awang et al. [4] proposed heterogeneous ensemble selection where the selection of best base classifier subset was performed using soft set based method.

From those applications of ensemble learning in business, we identify some important characteristics of ensembles. Some used multi-tier ensemble for simultaneous selection of feature selection method and the base classifiers. A substantial number of ensembles were built on dynamic selection approach. The majority of the ensembles proposed in these areas were in the category of homogeneous ensemble learning. While this volume presents a multi-objective approach, most ensemble learning techniques so far are based on a single objective. It is clear that we are moving towards a meta-analytic frontier with the combination of more powerful heuristics, metaheuristics, and the increasing availability of software codes that provide access to a large number of base classifiers. The phenomenon is also seen

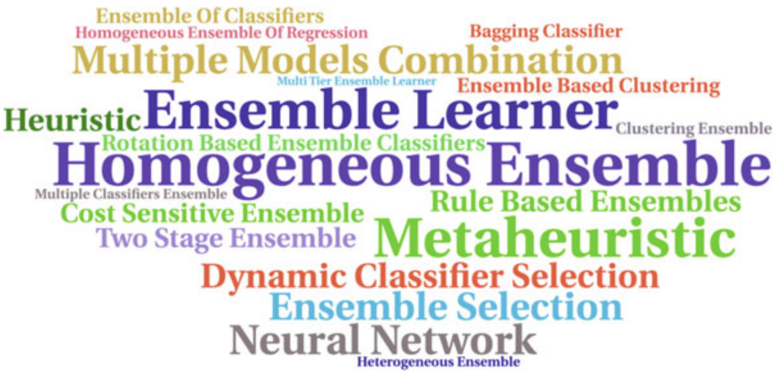


Fig. 18.6 Word cloud view of the types of approaches used in business analytics for the last decade. The size of the word depends on the frequency of the approach in the literature, i.e. most frequent to least frequent method size varies from the largest to smallest. Note the conspicuous appearance of the word "Metaheuristics" in this set

from the word cloud of the types of methods used in business analytics in Fig. 18.6. Here, the metaheuristic appeared nearly equal to the Ensemble learner.

18.5 Future Challenges and Conclusion

Despite the success of some existing ensemble learning, take these as the pioneers of the more advanced meta-analytics methods to come in the future, they show not only the opportunities were given but some of the challenges ahead. Some of them stem from the inherent characteristics of the datasets. The recent advent of big data and stream datasets brings particular challenges to ensemble learning techniques. For instance, they add an extra layer of problem-domain difficulties, e.g. redundancy, noise, heterogeneity of data sources, lack of annotation, and imbalanced data [101]. For eliminating the redundancy in data, traditional pairwise approaches such as the use of Euclidean distance-based algorithms are not fast enough for big data and for data streams. Data gathered from various sources for Big Data analytics (especially in business analytics tasks) sometimes poses the problem of syntactic (distribution of values) and semantic (understanding) heterogeneity between data gathered from different sources [51].

Knowledge discovery from imbalanced class dataset is a challenging topic in data mining. This problem takes place when the number of cases that represent one class is remarkably lower than the number on other classes. This is not uncommon when samples of one class occur with remarkably lower probability than in the other one. The ensemble of classifiers process needs to find a way to handle these situations. Li [52] proposed a bagging of classification to creating the ensemble learner. They used the minority class data maximally without creating synthetic data or making changes to the existing classification systems. The experimental results using real world imbalanced data are advocated for the efficacy of the proposal. In [77], a review of existing methods for classification of data with imbalanced class distribution was provided. This work gives an overview of the classification of imbalanced data regarding the application domains and the nature of the problem. They also addressed how to obtain objective and measure accuracy, the learning difficulties with conventional classifier learning algorithms, and finally the class imbalance problem in the presence of multiple classes. For this reason, diversity is particularly crucial for ensembles. Mirza et al. [60] proposed an ensemble of online sequential extreme learning machine (ESOS-ELM) approach that employs neural networks and two different learning schemes to handle the imbalance ratio in data streams. However, the algorithm presented some limitations due to the assumption that no drift takes place on the minority class.

This diversity in real-world applications has served along with a growth of research from researchers. Hence, Kuncheva [47] have studied ten statistics to determine diversity among binary classifier outputs. They have examined the relationship between the ensemble accuracy and different measures of diversity, and among the measures themselves. Their results disagreed with some proven relationship between accuracy and diversity measures in building classifier ensembles in real-life pattern recognition problems. Brown et al. [13] first reviewed the various attempts to provide a formal interpretation of error diversity, including several heuristics and qualitative explanations in the literature. They surveyed the various techniques used for creating various ensembles, and categorized them, forming an exploratory taxonomy of diversity creation methods. They introduced the concept of implicit and explicit diversity generation methods and proposed some new directions that may prove useful in understanding classification error diversity.

Another challenge for creating an ensemble of classifiers is the combination method. It plays a significant role in the process of ensemble classifiers final decision and classification accuracy. In this regard, Jain et al. [36] explored the principle of several classifiers combination method. They also mentioned different types of combination process like feature sets, training sets, classification methods, or different training sessions joined together. The result is fused by a number of classifiers with the hope of improving overall classification accuracy. The nearest mean method for classifier combination gave the best overall result. This was also the best result of the entire experiment. Kleinberg [41] bridged the gap between the theoretical prediction expressed by stochastic discernment and practical explanation of algorithmic implementation. He also tested and found out that stochastic discrimination outperformed both boosting and bagging in the majority of benchmark problems.

In machine learning, many researchers used the ensemble of classifiers to improve the accuracy of individual classifiers by mixing many of them. Neither of these learning methods alone solves the class imbalance problem. The ensemble algorithms need to be specially designed to deal with these problems. Other relevant problems with the data characteristics need to be handled in the pre-processing step of the ensemble learning. The recent advances of redundancy elimination techniques from big data (faster version of minimum-Redundancy- Maximum-Relevance (Fast-mRMR) [70]) have opened new opportunities for ensemble learning by reducing the redundancy in big data. To reduce the noise, García-Gil et al. [29] proposed a homogeneous ensemble and a heterogeneous ensemble filter approach for big data. Research on ensemble learning for Big Data is advancing through solving the different problems with peculiar characteristics [28] and proposing new methods and approaches to tackle the big data using ensemble learning [57, 78, 86, 91].

We expect the future research in the big data analytics will continue their investigation to address these issues. From this comprehensive review, it is evident that the ensemble learning has the power to deal with the analytics of big data. Additionally, Meta-heuristics will increase the capability of ensemble learning into a new dimension. Hence, the ensemble learning is posed as an effective way of utilizing the power of meta-heuristics to solve real-world big data analytics problems.

Acknowledgements Pablo Moscato acknowledges previous support from the Australian Research Council Future Fellowship FT120100060 and Australian Research Council Discovery Projects DP120102576 and DP140104183.

References

1. Abbasimehr H, Setak M, Tarokh MJ (2014) A comparative assessment of the performance of ensemble learning in customer churn prediction. Int Arab J Inf Technol 11(6):599–606
2. Akbaş A, Turhal U, Babur S, Avci C (2013) Performance improvement with combining multiple approaches to diagnosis of thyroid cancer. Engineering 5(10):264
3. Amini M, Rezaeenour J, Hadavandi E (2015) A cluster-based data balancing ensemble classifier for response modeling in bank direct marketing. International Journal of Computational Intelligence and Applications 14(4)
4. Awang MK, Makhtar M, Rahman MNA, Deris MM (2016) A new customer churn prediction approach based on soft set ensemble pruning. In: SCDM, Springer, Advances in Intelligent Systems and Computing, vol 549, pp 427–436
5. Baumann A, Lessmann S, Coussement K, Bock KWD (2015) Maximize what matters: Predicting customer churn with decision-centric ensemble selection. In: ECIS
6. Blanco R, Larrañaga P, Inza I, Sierra B (2004) Gene selection for cancer classification using wrapper approaches. International Journal of Pattern Recognition and Artificial Intelligence 18(08):1373–1390
7. Blaszczynski J, Dembczynski K, Kotlowski W, Pawlowski M (2006) Mining direct marketing data by ensembles of weak learners and rough set methods. In: DaWaK, Springer, Lecture Notes in Computer Science, vol 4081, pp 218–227
8. Bock KWD, den Poel DV (2010) Ensembles of probability estimation trees for customer churn prediction. In: IEA/AIE (2), Springer, Lecture Notes in Computer Science, vol 6097, pp 57–66
9. Bock KWD, den Poel DV (2011) An empirical evaluation of rotation-based ensemble classifiers for customer churn prediction. Expert Syst Appl 38(10):12,293–12,301
10. Breiman L (2001) Random forests. Machine Learning 45(1):5–32
11. Breiman L, Breiman L (1996) Bagging predictors. In: Machine Learning, pp 123–140
12. Breiman L, et al (2001) Statistical modeling: The two cultures (with comments and a rejoinder by the author). Statistical science 16(3):199–231

13. Brown G, Wyatt J, Harris R, Yao X (2005) Diversity creation methods: a survey and categorisation. Information Fusion 6(1):5–20
14. Cleofas L, Valdovinos RM, García V, Alejo R, Universitario C, Valle U (2009) Use of Ensemble Based on GA for Imbalance Problem. In: 6th International Symposium on Neural Networks, ISNN 2009 Wuhan, China, May 26-29, 2009 Proceedings, Part II, Springer Berlin Heidelberg, pp 547–554
15. Cunningham P, Carney J (2000) Diversity versus quality in classification ensembles based on feature selection. In: European Conference on Machine Learning, Springer, pp 109–116
16. Cuzzocrea A, Folino F, Guarascio M, Pontieri L (2016) A multi-view multi-dimensional ensemble learning approach to mining business process deviances. In: IJCNN, IEEE, pp 3809–3816
17. Dahiya S, Handa S, Singh N (2017) A feature selection enabled hybrid-bagging algorithm for credit risk evaluation. Expert Systems
18. Dietterich T (2000) Ensemble methods in machine learning. In: Multiple Classifier Systems, Lecture Notes in Computer Science, vol 1857, Springer Berlin Heidelberg, pp 1–15
19. Dounias G, Tsakonas A, Charalampakis D, Vasilakis E (2013) Effective business plan evaluation using an evolutionary ensemble. In: DATA, SciTePress, pp 97–103
20. Duangsoithong R, Windeatt T (2010) Bootstrap feature selection for ensemble classifiers. In: Proceedings of the 10th industrial conference on Advances in data mining: applications and theoretical aspects, Springer-Verlag, Berlin, Heidelberg, ICDM'10, pp 28–41
21. Ebrahimpour MK, Eftekhari M (2017) Ensemble of feature selection methods: A hesitant fuzzy sets approach. Applied Soft Computing 50:300–312
22. Fallahpour S, Lakvan EN, Zadeh MH (2017) Using an ensemble classifier based on sequential floating forward selection for financial distress prediction problem. Journal of Retailing and Consumer Services 34:159–167
23. Folino F, Guarascio M, Pontieri L (2012) Context-aware predictions on business processes: An ensemble-based solution. In: NFMCP, Springer, Lecture Notes in Computer Science, vol 7765, pp 215–229
24. Gaber MM, Bader-El-Den M (2012) Optimisation of Ensemble Classifiers using Genetic Algorithm. In: Graña M, Toro C, Posada J, Howlett RJ, Jain LC (eds) Advances in Knowledge-Based and Intelligent Information and Engineering Systems, IOS Press
25. Gabrys B, Ruta D (2006) Genetic algorithms in classifier fusion. Applied Soft Computing 6(4):337–347
26. Galar M, Fernández A, Barrenechea E, Bustince H, Herrera F (2011) An overview of ensemble methods for binary classifiers in multi-class problems: Experimental study on one-vs-one and one-vs-all schemes. Pattern Recognition 44(8):1761–1776
27. Galar M, Fernández A, Barrenechea E, Bustince H, Herrera F (2012) A Review on Ensembles for the Class Imbalance Problem: Bagging-, Boosting-, and Hybrid-Based Approaches. IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews 42(4):463–484
28. García S, Ramírez-Gallego S, Luengo J, Benítez JM, Herrera F (2016) Big data preprocessing: methods and prospects. Big Data Analytics 1(1):9
29. García-Gil D, Luengo J, García S, Herrera F (2017) Enabling Smart Data: Noise filtering in Big Data classification. ArXiv e-prints 1704.01770
30. Govindarajan M (2015) Comparative study of ensemble classifiers for direct marketing. Intelligent Decision Technologies 9(2):141–152
31. Espejo P, Ventura S, Herrera F (2010) A survey on the application of genetic programming to classification. IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews 40(2):121–144
32. Haque MN, Noman N, Berretta R, Moscato P (2016) Heterogeneous ensemble combination search using genetic algorithm for class imbalanced data classification. PLoS ONE 11(1):e0146,116.

33. Haque MN, Noman N, Berretta R, Moscato P (2016b) Optimising weights for heterogeneous ensemble of classifiers with differential evolution. In: 2016 IEEE Congress on Evolutionary Computation (CEC), pp 233–240
34. Hernández-Lobato D, Martínez-Muñoz G, Suárez A (2006) Pruning in ordered regression bagging ensembles. In: The 2006 IEEE International Joint Conference on Neural Network Proceedings, pp 1266–1273
35. Hu H, Li J, Wang H, Daggard G (2008) Robustness analysis of diversified ensemble decision tree algorithms for Microarray data classification. 2008 International Conference on Machine Learning and Cybernetics pp 115–120
36. Jain AK, Duin RPW, Mao J, Member S (2000) Statistical Pattern Recognition : A Review. IEEE Transactions on Pattern Analysis and Machine Intelligence 22(1):4–37
37. Jordan MI, Jacobs RA (1994) Hierarchical Mixtures of Experts and the EM Algorithm. Neural Computation 6(2):181–214
38. Kim Y (2009) Boosting and measuring the performance of ensembles for a successful database marketing. Expert Syst Appl 36(2):2161–2176
39. Kim Y, Street WN, Menczer F (2006) Optimal ensemble construction via meta-evolutionary ensembles. Expert Systems with Applications 30(4):705–714
40. Kim YW, Oh IS (2008) Classifier ensemble selection using hybrid genetic algorithms. Pattern Recognition Letters 29(6):796–802
41. Kleinberg E (2000) On the algorithmic implementation of stochastic discrimination. IEEE Transactions on Pattern Analysis and Machine Intelligence 22(5):473–490
42. Ko AH, Sabourin R, Britto AS, Jr (2008) From dynamic classifier selection to dynamic ensemble selection. Pattern Recognition 41(5):1718–1731
43. Kotsiantis S, Zaharakis I, Pintelas P (2006) Machine learning: a review of classification and combining techniques. Artificial Intelligence Review 26(3):159–190
44. Koutanaei FN, Sajedi H, Khanbabaei M (2015) A hybrid data mining model of feature selection algorithms and ensemble learning classifiers for credit scoring. Journal of Retailing and Consumer Services 27:11–23
45. Krawczyk B, Galar M, Jeleń Ł, Herrera F (2016) Evolutionary undersampling boosting for imbalanced classification of breast cancer malignancy. Applied Soft Computing 38:714–726
46. Kuncheva L, Jain L (2000) Designing classifier fusion systems by genetic algorithms. IEEE Transactions on Evolutionary Computation 4(4):327–336
47. Kuncheva L, Whitaker C (2003) Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy. Machine Learning 51(2):181–207
48. Kuncheva LI (2014) Combining Pattern Classifiers: Methods and Algorithms, 2nd edn. John Wiley & Sons, Inc.
49. Lai KK, Yu L, Wang S, Huang W (2007) An intelligent CRM system for identifying high-risk customers: An ensemble data mining approach. In: International Conference on Computational Science (2), Springer, Lecture Notes in Computer Science, vol 4488, pp 486–489
50. Lertampaiporn S, Thammarongtham C, Nukoolkit C, Kaewkamnerdpong B, Ruengjitchatchawalya M (2013) Heterogeneous ensemble approach with discriminative features and modified-SMOTEBagging for pre-miRNA classification. Nucleic acids research 41(1):e21
51. L'Heureux A, Grolinger K, ElYamany HF, Capretz M (2017) Machine learning with big data: Challenges and approaches. IEEE Access PP(99)
52. Li C (2007) Classifying imbalanced data using a bagging ensemble variation (BEV). In: Proceedings of the 45th annual southeast regional conference, ACM, New York, NY, USA, ACM-SE 45, pp 203–208
53. Li H, Sun J (2011) Principal component case-based reasoning ensemble for business failure prediction. Information & Management 48(6):220–227
54. Li H, Sun J (2012) Case-based reasoning ensemble and business application: A computational approach from multiple case representations driven by randomness. Expert Syst Appl 39(3):3298–3310

55. Li H, Sun J, Li J, Yan X (2013) Forecasting business failure using two-stage ensemble of multivariate discriminant analysis and logistic regression. Expert Systems 30(5):385–397
56. Liu Y, Wei W, Wang K, Liao Z, Gao J (2011) Balanced-sampling-based heterogeneous SVR ensemble for business demand forecasting. In: ICIC (1), Springer, Lecture Notes in Computer Science, vol 6838, pp 91–99
57. Ma C, Zhang HH, Wang X (2014) Machine learning for big data analytics in plants. Trends in plant science 19(12):798–808
58. Matthews C, Scheurmann E (2008) Ensembles of classifiers in arrears management. In: Soft Computing Applications in Business, Studies in Fuzziness and Soft Computing, vol 230, Springer, pp 1–18
59. Minaei-Bidgoli B, Kortemeyer G, Punch W (2004) Optimizing classification ensembles via a genetic algorithm for a web-based educational system. In: Fred A, Caelli T, Duin R, Campilho A, de Ridder D (eds) Structural, Syntactic, and Statistical Pattern Recognition, Lecture Notes in Computer Science, vol 3138, Springer Berlin Heidelberg, pp 397–406
60. Mirza B, Lin Z, Liu N (2015) Ensemble of subset online sequential extreme learning machine for class imbalance and concept drift. Neurocomputing 149:316–329
61. Namsrai E, Munkhdalai T, Li M, Shin JH, Namsrai OE, Ryu KH (2013) A Feature Selection-based Ensemble Method for Arrhythmia Classification. Journal of Information Processing Systems 9(1):31–40
62. Nikulin V, Mclachlan GJ, Ng SK (2009) Ensemble Approach for the Classification of Imbalanced Data. In: AI 2009: Advances in Artificial Intelligence, Springer, pp 291–300
63. Oh DY, Gray JB (2013) GA-Ensemble: a genetic algorithm for robust ensembles. Computational Statistics 28(5):2333–2347
64. Oliveira L, Morita M, Sabourin R, Bortolozzi F (2005) Multi-objective genetic algorithms to create ensemble of classifiers. In: Coello Coello CA, Hernández Aguirre A, Zitzler E (eds) Evolutionary Multi-Criterion Optimization, Lecture Notes in Computer Science, vol 3410, Springer Berlin Heidelberg, pp 592–606
65. Oliveira LS, Sabourin R, Bortolozzi F, Suen CY (2003) Feature selection for ensembles: A hierarchical multi-objective genetic algorithm approach. In: Proceedings of the Seventh International Conference on Document Analysis and Recognition - Volume 2, IEEE Computer Society, Washington, DC, USA, ICDAR ’03, pp 676–680
66. Osareh A, Shadgar B (2013) An Efficient Ensemble Learning Method for Gene Microarray Classification. BioMed Research International 2013:1–10
67. Oza NC (2006) Ensemble data mining methods. In: Wang J (ed) Encyclopedia of Data Warehousing and Mining, Idea Group Reference, vol 1, pp 448–453
68. Oza NC, Tumer K (2008) Classifier ensembles: Select real-world applications. Information Fusion 9(1):4–20
69. Polikar R (2006) Ensemble based systems in decision making. Circuits and Systems Magazine, IEEE 6(3):21–45
70. Ramírez-Gallego S, Lastra I, Martínez-Rego D, Bolón-Canedo V, Benítez JM, Herrera F, Alonso-Betanzos A (2017) Fast-mRMR: Fast minimum redundancy maximum relevance algorithm for high-dimensional big data. International Journal of Intelligent Systems 32(2):134–152
71. Ranawana R, Palade V (2006) Multi-Classifier Systems: Review and a roadmap for developers. International Journal of Hybrid Intelligent Systems 3(1):35–61
72. Rokach L (2009) Ensemble-based classifiers. Artificial Intelligence Review 33(1-2):1–39
73. Roli F, Giacinto G, Vernazza G (2001) Methods for designing multiple classifier systems. In: Kittler J, Roli F (eds) Multiple Classifier Systems, Lecture Notes in Computer Science, vol 2096, Springer Berlin Heidelberg, pp 78–87
74. Santana A, Soares R, Canuto A, Souto MCPd (2006) A dynamic classifier selection method to build ensembles using accuracy and diversity. In: Neural Networks, 2006. SBRN ’06. Ninth Brazilian Symposium on, pp 36–41
75. Seijo-Pardo B, Bolón-Canedo V, Alonso-Betanzos A (2017) Testing different ensemble configurations for feature selection. Neural Processing Letters pp 1–24

76. Srimani PK, Koti MS (2013) Medical Diagnosis Using Ensemble Classifiers - A Novel Machine-Learning Approach. Journal of Advanced Computing pp 9–27
77. Sun Y, Wong AKC, Kamel MS (2009) Classification of Imbalanced Data: A Review. International Journal of Pattern Recognition and Artificial Intelligence 23(04):687–719
78. Tang Y, Wang Y, Cooper KM, Li L (2014) Towards big data Bayesian network learning-an ensemble learning based approach. In: Big Data (BigData Congress), 2014 IEEE International Congress on, IEEE, pp 355–357
79. Thammasiri D, Meesad P (2012) Ensemble Data Classification based on Diversity of Classifiers Optimized by Genetic Algorithm. Advanced Materials Research 433-440:6572–6578
80. Tsymbal A, Puuronen S, Patterson DW (2003) Ensemble feature selection with the simple Bayesian classification. Information Fusion 4(2):87–100
81. Tulyakov S, Jaeger S, Govindaraju V, Doermann D (2008) Review of classifier combination methods. In: Marinai S, Fujisawa H (eds) Machine Learning in Document Analysis and Recognition, Studies in Computational Intelligence, vol 90, Springer Berlin Heidelberg, pp 361–386
82. Valentini G, Masulli F (2002) Ensembles of learning machines. In: Marinaro M, Tagliaferri R (eds) Neural Nets, Lecture Notes in Computer Science, vol 2486, Springer Berlin Heidelberg, pp 3–20
83. Wang L, Wu C (2017) Business failure prediction based on two-stage selective ensemble with manifold learning algorithm and kernel-based fuzzy self-organizing map. Knowl-Based Syst 121:99–110
84. Wang X, Wang H (2006) Classification by evolutionary ensembles. Pattern Recognition 39(4):595–607
85. Wang Y, Xiao H (2011) Ensemble learning for customers targeting. In: KSEM, Springer, Lecture Notes in Computer Science, vol 7091, pp 24–31
86. Wang Y, Yu C (2016) Research on the database marketing in the big data environment based on ensemble learning. Economics 12(6):21–32
87. Xiao J, Xie L, He C, Jiang X (2012) Dynamic classifier ensemble model for customer classification with imbalanced class distribution. Expert Syst Appl 39(3):3668–3675
88. Xiao J, Wang Y, Wang S (2013) A dynamic transfer ensemble model for customer churn prediction. In: BIFE, IEEE Computer Society, pp 115–119
89. Xiao J, Xiao Y, Huang A, Liu D, Wang S (2015) Feature-selection-based dynamic transfer ensemble model for customer churn prediction. Knowl Inf Syst 43(1):29–51
90. Xiao J, Jiang X, He C, Teng G (2016) Churn prediction in customer relationship management via GMDH-based multiple classifiers ensemble. IEEE Intelligent Systems 31(2):37–44
91. Xie L, Draizen EJ, Bourne PE (2017) Harnessing big data for systems pharmacology. Annual Review of Pharmacology and Toxicology 57:245–262
92. Xu R, He L (2008) GACEM: Genetic Algorithm Based Classifier Ensemble in a Multi-sensor System. Sensors 8(10):6203–6224
93. Yang P, Liu W, Zhou BB, Chawla S, Zomaya AY (2013) Ensemble-based wrapper methods for feature selection and class imbalance learning. In: Pacific-Asia Conference on Knowledge Discovery and Data Mining, Springer, pp 544–555
94. Yu L, Lai KK, Wang S (2008a) An evolutionary programming based knowledge ensemble model for business risk identification. In: Soft Computing Applications in Business, Studies in Fuzziness and Soft Computing, vol 230, Springer, pp 57–72
95. Yu L, Wang S, Lai KK (2008b) An EMD-based neural network ensemble learning model for world crude oil spot price forecasting. In: Soft Computing Applications in Business, Studies in Fuzziness and Soft Computing, vol 230, Springer, pp 261–271
96. Zhang L, Wang X, Moon WM (2015) PolSAR images classification through GA-based selective ensemble learning. In: Geoscience and Remote Sensing Symposium (IGARSS), 2015 IEEE International, pp 3770–3773
97. Zhang Y, Bhattacharyya S (2004) Genetic programming in classifying large-scale data: an ensemble method. Information Sciences 163(1–3):85–101

98. Zhang Z, Chen Q, Ke S, Wu Y, Qi F (2010a) Ranking potential customers based on group-ensemble. In: Strategic Advancements in Utilizing Data Mining and Warehousing Technologies, IGI Global, pp 355–365
99. Zhang Z, Li J, Hu H, Zhou H (2010b) A robust ensemble classification method analysis. In: Arabnia HR (ed) Advances in Computational Biology, Advances in Experimental Medicine and Biology, vol 680, Springer New York, pp 149–155
100. Zhao W, Liu H, Dai W, Ma J (2016) An entropy-based clustering ensemble method to support resource allocation in business process management. Knowl Inf Syst 48(2):305–330
101. Zhou L, Pan S, Wang J, Vasilakos AV (2017) Machine learning on big data: Opportunities and challenges. Neurocomputing 237:350–361

Chapter 19
Metaheuristics and Classifier Ensembles

Ringolf Thomschke, Stefan Voß, and Stefan Lessmann

Abstract A classifier ensemble combines several base models into a composite model to increase predictive accuracy. Given a set of candidate base models, the question which of these to incorporate into an ensemble and whether to weight base models differently has received much interest in the machine learning literature. Using heuristic search for ensemble member selection has proven a viable approach. However, research has till now considered only a small set of (meta-)heuristics for this type of problem. More generally, whether the choice of a metaheuristic is important has not been addressed at all. This paper aims at filling this gap. To that end, a comprehensive set of metaheuristics is employed to create alternative ensemble classifiers and these are compared in the scope of an empirical study. The results observed in several experiments provide original insights concerning the relative effectiveness of different metaheuristics and fitness functions for ensemble modelling. Having identified a particularly promising modelling approach, the paper proceeds with comparisons to other ensemble regimes and more generally prediction models to assess the degree to which a metaheuristic-based ensemble improves upon the state-of-the-art. As part of this analysis, the paper also proposes an approach to approximate an optimality gap for predictive classification models.

Keywords Classifier ensembles · Machine learning · Metaheuristics · Predictive analytics

R. Thomschke · S. Lessmann
Humboldt-Universität zu Berlin, School of Business and Economics, Berlin, Germany
e-mail: stefan.lessmann@hu-berlin.de

S. Voß (✉)
University of Hamburg, Institute of Information Systems, Hamburg, Germany
e-mail: stefan.voss@uni-hamburg.de

P. Moscato, N. J. de Vries (eds.), *Business and Consumer Analytics: New Ideas*,
https://doi.org/10.1007/978-3-030-06222-4_19

19.1 Introduction

Meta-Analytics is an emerging discipline at the interface of heuristic search and analytics. Heuristic search, although a relatively young academic discipline in its own right, is a domain with a strong track record in developing powerful algorithms capable of solving complex optimization problems in industrial engineering, production planning, and logistics; to name a few of the application areas where such complex problems abound. A variety of heuristics and metaheuristics have been developed and demonstrated their success in various real-world applications. Obtaining a clear view of *analytics* is maybe more challenging, with a need to cut through the promotional hype that often obscures its fundamental concepts. In this paper, we focus on one clearly distinguishable part of the analytics wave, namely predictive analytics, which is a discipline concerned with developing empirical models that provide operationally accurate forecasts [52]. Studying the interface of metaheuristics and predictive analytics through the lens of ensemble modelling, our goal is to illustrate a potential use case of Meta-Analytics and examine the effectiveness of the specific design artefact that emerges from integrating concepts from metaheuristics and ensemble classification.

In machine learning, ensembles are composite models that consist of a variety of base classifiers. Much theoretical and empirical evidence is available which discloses that ensembles of base classifiers provide more accurate predictions than an individual classification model in isolation. One strategy to create an ensemble is to weight each model according to its contribution to a certain measure of fitness, such as a measure of predictive accuracy. Determining such weights can be formulated as a high-dimensional optimization problem for which a large number of solvers exist. These also include metaheuristics, a class of procedures that are problem-independent in the sense of applying to problems from many domains, and perform a robust search in complex solution spaces. While metaheuristics do not guarantee finding an optimal solution, they have been shown to be remarkably effective in providing good solutions, especially if the optimization task is highly complex.

In recent years, a marriage of ensemble methods and metaheuristics has received some interest in the literature, particularly by using metaheuristics for specific tasks in the ensemble modelling process. One such task is the selection of a subset of base classifiers from an overall pool. For example, [19] used a genetic algorithm (GA) with bit-encoding to find a (near-)optimal subset of classifiers and compared different combination rules for two data sets. Zhou et al. [68] presented an algorithm that trains different neural networks, uses a GA to perform a subset selection and subsequently averaging to combine the outputs. Ant colony optimization was employed for subset selection by Chen and Wong [9] who benchmarked their approach on 18 data sets and compared it to five other algorithms. Also, simulated annealing was suggested to be combined with forward selection by Taghavi and Sajedi [57] who tested their proposal on ten different data sets with pools of 100 classifiers each.

A second domain of application undertakes to optimize classifier weights for weight combination rules. Here, Nabavi-Kerizi et al. [40] employed particle swarm optimization (PSO) to find (near-)optimal weight matrices to linearly combine neural networks for solving multi-class problems of three data sets. Quoos et al. [45] compared PSO and GA in finding weights for weighted averaging of classifiers built on three data sets. Ekbal and Saha [17] presented a multi-objective simulated annealing approach which determines weights for weighted majority voting for a combined fitness measure of accuracy and diversity.

Another popular use case for metaheuristics in classification is feature selection. Tahir and Smith [58] aimed at creating diverse nearest neighbour ensembles using a combination of tabu search and neighbourhood search to simultaneously select features in 14 data sets. Palanisamy and Kanmani [42] presented a classifier ensemble design that integrated an artificial bee colony based feature selection. Lastly, metaheuristics were utilized to optimize metaparameters of existing ensemble algorithms. For example, Coletta et al. [13] compared five different metaheuristics in optimizing two control parameters of the *Combination of Classifier and Cluster Ensembles* algorithm.

The above examples illustrate that previous research has considered the use of metaheuristics in ensemble modelling, but has done so in an ad hoc manner. For example, we may see the use of a GA or PSO in a paper but typically lack a well-reflected discussion whether GA or PSO is the best choice for the task at hand. At best, a study might provide some anecdotal evidence to motivate the specific choice of metaheuristic that was made, referring, for example, to promising results observed with some metaheuristic X in some prior work. The lack of a broad and systematic analysis of the effectiveness and efficiency of alternative metaheuristics for the task of ensemble development is the starting point of this paper. Performing a comparison that tests the relative merits of a large set of metaheuristics from different families, and that provides solid empirical evidence for or against the importance of choosing the right metaheuristic for ensemble modelling, are the goals we pursue in this work. More specifically, we compare ensemble classifier resulting from the use of different metaheuristics during ensemble construction while keeping other factors in the modelling process constant, in terms of accuracy, diversity, computation time, and the sparsity of the base model weight vector. It is straightforward to see that predictive accuracy and computational efficiency are important measures of the quality of a predictive decision support model. Diversity and sparsity are measures that originate from theories of statistical learning. Sparse models are less complex and more likely to classify novel observations correctly [29]. Diversity captures the degree to which different base models in an ensemble complement each other and governs the success of ensemble modelling [34].

The contributions of the paper to the academic literature are as follows: First, we conduct a comprehensive comparison of a diverse set of metaheuristics using a large collection of data sets with different characteristics, the important task of ensemble modelling in predictive analysis. This provides novel insights concerning the relevance of the base model selection step and the extent to which more sophisticated search techniques improve predictive accuracy. For example, many of the metaheuristics we consider have, to the best of our knowledge, not been considered in predictive modelling at all—even in the simple case where no attempt is made to optimize the parameter settings. Second, we propose an approach to approximate what is known as an optimality gap in operations research in a predictive modelling context. We argue that such an analysis augments standard evaluation practices in predictive and ensemble modelling in a valuable way, in that it provides an estimate of the best possible accuracy on a given hold-out test set. The standard practice in machine learning is to employ lower-bound models using, for example, naive predictions and alternative prediction models as benchmarks. We illustrate how an additional upper-bound of performance can provide auxiliary information. Last, a third contribution of the paper consists in identifying a potential application area for Meta-Analytics and may, in that respect, help to further advance this field. When introducing some data sets below, we briefly mention possible business applications and their uses.

The paper is organized as follows: In the next section, we examine the theoretical background of ensemble modelling and discuss alternative measures of model performance. Section 19.3 explains the theoretical concept of a metaheuristic and briefly describes the basic functioning of each of the 16 metaheuristic procedures which we consider in our study. Section 19.4 presents the experimental study with the data, algorithmic framework, and results. Section 19.5 concludes the paper.

19.2 Foundations of Ensemble Modelling

Ensemble learners combine several individual models and form a composite model. The individual models originate from applying different learning algorithms to the same data set or applying a single learning algorithm to resampled versions of the data set. The ensemble learning literature refers to the individual models as ensemble members, weak learners, or base models. The goal of combining base models and developing an ensemble model, respectively, is to increase predictive accuracy. This is possible because model combination facilitates reducing both bias and variance, which are the main sources of errors in predictive models [67]. A reduction in variance follows directly from the combination of multiple base models, each of which has a different view on the data and makes different errors [14]. In addition, combining multiple relatively simple models can create a more powerful model with higher representational complexity. For example, a complex non-linear function can be approximated through a piece-wise linear function (i.e., splines), which can be considered a base model.

The prevailing approach to develop an ensemble forecast is to average over the prediction of the base models. Average-based combination works for regression models (which predict a continuous target variable) and classification models alike. In the latter case, the base model predictions can have the form of class probability estimates or discrete class prediction. Note that calculating an average over discrete class predictions is equivalent to majority voting.

Simple averaging is a special form of averaging that unites base classifiers h_i, $i = 1, \ldots, N$ to an ensemble H such that, for an instance x, the combined output is given by

$$H(x) = \frac{1}{N} \sum_{i=1}^{N} h_i(x). \tag{19.1}$$

It is also possible to form a weighted average ensemble to give some base models a large influence on the composite forecast. This corresponds to an ensemble of the form

$$H(x) = \sum_{i=1}^{N} s_i h_i(x), \tag{19.2}$$

whereby the weights s_i, $i = 1, \ldots, N$ are constrained by

$$s_i \geq 0 \ \text{ and } \ \sum_{i=1}^{N} s_i = 1. \tag{19.3}$$

There exists a closed-form solution to the optimal weighting. The weights s_i which minimize the error of the ensemble are given by

$$s_i = \frac{\sum\limits_{j=1}^{N} C_{ij}^{-1}}{\sum\limits_{k=1}^{N} \sum\limits_{j=1}^{N} C_{kj}^{-1}}, \tag{19.4}$$

where C is the $N \times N$ correlation matrix of the predictions of the base classifiers h_i. This solution requires C to be invertible. However, in general, C is ill-conditioned or singular because the errors of the base classifiers are strongly correlated due to the fact that the base classifiers are trained on the same data. Then, a heuristic approach is to weight base models according to their predictive accuracy on a hold-out set of validation data.

Simple averaging is a special case of weighted averaging with all weights being equal. Weighted averaging does not necessarily improve performance. In particular, determining weights can be a complex undertaking, due to, for example, the noisiness of data that often occurs in practice. Learning weights may then lead to over-fitting or unstable results, whereas simple averaging is more robust towards over-fitting. However, more recently, the integration of ensemble modelling and heuristic search has led to an increasing interest in weighted ensembles and much empirical evidence has been provided to demonstrate the potential of such an integration [37, 43, 61]. In general, procedures proposed in this scope first develop a potentially large set of base models, called a model library, and subsequently employ a heuristic search to select and weight a subset of the base models for the final ensemble [60]. In such a framework, the decision of how to organize the search for suitable base models from the library (e.g., base models that complement each other) and which specific objective to pursue become important. Prior work has considered the second question to some extent. Whether it is better to maximize predictive accuracy of the final ensemble (on validation data) or the diversity of the base models in the ensemble is a question that has generated some interest [59]. A systematic analysis of the relative merits of alternative search strategies (e.g., metaheuristics), however, is lacking, which is exactly the research gap we strive to close with this paper.

19.3 Metaheuristics

Metaheuristics (MHs) are global optimization techniques which are not in themselves problem-dependent, but which can exploit domain knowledge and are applicable to a wide range of optimization problems. They connect strategies of local improvement (intensification) and global search (diversification) that allow for escaping from local optima and perform an effective exploration of the search space. Since they make only a few assumptions about the problem to be optimized, MHs are not guaranteed to find a globally optimal solution. However, they remain applicable even in the case of discontinuous, non-convex, or not differentiable objective functions and have been shown to effectively provide near-optimal solutions for highly complex problems where heuristic approaches fail to do so [8, 25]. An informative history of metaheuristic algorithms and a discussion of their disparate concepts are given in [55].

The algorithms considered in this paper are built as solvers to continuous optimization problems which in case of minimization can be modelled as follows:

Definition 19.1 The model $\mathcal{P} = \{\mathcal{S}, \Omega, f\}$ describes a continuous optimization problem with

- $\mathcal{S}$ being the search space over a finite set of decision variables X_j, $j = 1, \ldots, D$,
- Ω being the set of constraints, and
- $f : \mathcal{S} \rightarrow \mathbb{R}_+$ being the objective function to be optimized.

A solution $s \in \mathcal{S}$ consisting of assigned X_j is called a feasible solution if it satisfies all constraints in Ω. The optimization aims at finding a globally optimal solution $s^* \in \mathcal{S}$ for which $f(s^*) \leq f(s)\ \forall s \in \mathcal{S}$ must be fulfilled [5].

Blum and Roli [6] describe different points of view on how to categorize metaheuristic algorithms. For example, they can be divided in methods that operate starting from a single solution called trajectory methods and those starting from a set of solutions called population-based methods. Other ways to classify them are by their use of memory or whether they use the objective function statically or dynamically. The MHs described below are separated by their field of inspiration to obtain a balanced scheme of important, diverse, and easily accessible algorithms. Each metaheuristic is presented with its parameters and their settings used in the simulation study. A final comment refers to the usefulness of having these MHs despite the fact that some of them have not been shown to be novel despite their somewhat "sexy" naming and their appearance in dozens or even hundreds of papers. We are aware of the critics and conceptual overlaps (see, e.g., [26, 62]).

19.3.1 Evolutionary Algorithms

Evolutionary algorithms (EAs) are stochastic, metaheuristic optimization techniques which are inspired by principles of natural evolution. Yu and Gen [66] identify three main properties of EAs: They are population-based as they optimize the objective by maintaining a set of possible solutions called individuals. They are fitness-oriented as they select individuals with better fitness values which are determined in every iteration, also called generation. And they are variation-driven in the sense that they apply operations which change existing solutions in order to explore the search space. Analogously to natural evolution, the varying operators are generally referred to as mutation and crossover. Subsequently, five of these algorithms are further specified [66].

19.3.1.1 Genetic Algorithms

Genetic algorithms (GAs) are among the most widely used methods in stochastic optimization. They mimic the principles of biological evolution such as selection, crossover, and mutation to evolve a set of solutions towards a (near-)optimum value. The set of solutions is regarded as a population consisting of individuals and the objective function yields the fitness of an individual. The original idea was to encode the individuals into a bit-string representation, the chromosome, where each bit represents a gene [39]. However, it has been proven by Janikow and Michalewicz [33] that the bit-string representation tends to be less precise and slower than a floating point representation when optimizing in high dimensional continuous search spaces. Additional limitations of these representations and ways to overcome them are identified in [23]. For this reason, the algorithm described and applied in this paper is a real-valued GA as found in Matlab's Genetic Algorithm Toolbox [10]. Here, the individuals and their chromosome representation coincide, with the single variables being regarded as the genes. The algorithm begins with a random initialization of a predefined number of N individuals which are sorted by their fitness value quality in decreasing order. After this, the new generation of individuals is created. Determined by the elite count parameter e, a number of best

Table 19.1 GA parameters

Parameters	Symbol	Settings
Number of individuals	N	25
Gene representation		Real-valued
Elite count	e	2
Crossover parameter	p_c	0.8
Crossover type		"Intermediate crossover"
Mutation type		"Adaptive feasible"

individuals are selected to be passed onto the next generation without any changes. The crossover parameter p_c defines the fraction N_c of the next generation that will be produced by crossover such that

$$N_c = round(p_c \cdot (N - e)). \tag{19.5}$$

In order to perform crossover, two individuals s_1 and s_2 of the population are selected as parents based on probabilities that are scaled proportionally to their fitness. As recommended by the default settings for constraint continuous optimization, parents produce a child individual s_{new} by intermediate crossover, i.e.,

$$s_{new,k} = s_{1,k} + x(s_{2,k} - s_{1,k}),\ \ k = 1, \ldots, D, \tag{19.6}$$

with $x \sim U(0, 1)$ and D being the dimension of the search space. Another fraction of children in the next generation is created by mutation. For this purpose, a randomly selected individual from the population is changed by adding a random direction with random step length such that the child solution remains feasible. The number of mutation children (N_m) corresponds to the remaining $N - e - N_c$ spots in the next generation. While elitism and crossover intensify the search, mutation allows to explore undiscovered regions of the search space. These operators are repeatedly applied to the population until a termination criterion is reached. Table 19.1 presents the used GA parameters [10].

19.3.1.2 Covariance Matrix Adaptation Evolution Strategy

The covariance matrix adaptation evolution strategy (CMAES) is a population-based optimization procedure that finds new solutions by sampling from a multivariate normal distribution. In each iteration of the search, the parameters of this distribution, mean and covariance matrix, are estimated based on the population such that previously successful solutions are most likely. Furthermore, it makes use of the correlation of consecutive search steps by taking record of the distribution mean in the so-called evolution paths which guide the search and prevent it from converging early. The algorithm starts with randomly initializing the mean m inside the boundaries of the D-dimensional search space and setting the covariance matrix $C = I_D$ and the step-size parameter $\sigma > 0$. Then a predefined number of N solutions are generated as a random sample, i.e.,

$$s_i \sim N(m, \sigma^2 C),\ i = 1, \ldots, N. \tag{19.7}$$

The solutions are evaluated with the objective function and ordered by their quality. The algorithm proceeds with updating the mean as

$$m_{new} = \sum_{i=1}^{\mu} w_i s_{i:N}, \tag{19.8}$$

where $s_{i:N}$ is the i-th best solution, $\mu \leq N/2$ the number of parents of the new mean and w_i are the recombination weights with $\sum_{i=1}^{\mu} w_i = 1,\ w_i \geq 0$. Subsequently, the step-size σ_{new} is adjusted by first calculating the evolution path p_σ with

$$p_{\sigma,new} = (1 - c_\sigma)p_\sigma + \sqrt{1 - (1 - c_\sigma)^2}\sqrt{\mu_{eff}}\,/C^{\ 1/2} \cdot \frac{(m_{new} - m)}{\sigma}, \tag{19.9}$$

in which the initial $p_\sigma = 0$, c_σ is a discount parameter and $\mu_{eff} = \left(\sum_{i=1}^{\mu} w_i^2\right)^{-1}$ is called the variance effective selection mass. σ_{new} is now refreshed by

$$\sigma_{new} = \sigma\ \exp\left(\frac{c_\sigma}{d_\sigma}\left(\frac{\|p_{\sigma,new}\|}{E\|\mathcal{N}(0, I)\|} - 1\right)\right) \tag{19.10}$$

with d_σ being a damping parameter and $E\|\mathcal{N}(0, I)\|$ is the Euclidian norm of a $\mathcal{N}(0, I)$ distributed random vector. The step-size is only increased if $\|p_{\sigma,new}\|$ is larger than the denominator in Eq. (19.10); otherwise, it is decreased. Finally, the covariance matrix is adjusted after a respective evolution path was constructed with

$$p_{c,new} = (1 - c_c)\, p_c + \sqrt{c_c(2 - c_c)}\, \sqrt{\mu_w}\, \frac{m_{new} - m}{\sigma}, \tag{19.11}$$

where $c_c \leq 1$ is the respective discount parameter. The covariance matrix of the next generation is then obtained by

$$C_{new} = (1-c_1-c_\mu)\, C + c_1\, p_{c,new}\, p_{c,new}^T + c_\mu \sum_{i=1}^{\mu} w_i \left(\frac{s_{i:N} - m}{\sigma}\right)\left(\frac{s_{i:N} - m}{\sigma}\right)^T, \tag{19.12}$$

where c_1 and c_μ are learning rates. The algorithm restarts the sampling process with the updated values for m, C, and σ until a termination criterion is reached [28]. The parameters of CMAES are summarized in Table 19.2.

Table 19.2 Parameters of CMAES

Parameters	Symbol	Settings
Sample size	N	$(4 + round(3\ln(D))) \cdot 10$
Number of parents	μ	$N/2$
Initial step-size	σ	0.3
Recombination weights	w_i	$w_i = (ln(\mu + 1/2) - ln(i)),\ w_i = w_i / \sum_{j=1}^{\mu} w_j$
Variance effective selection mass	μ_{eff}	$1/\sum_{i=1}^{\mu} w_i^2$
Discount parameter	c_σ	$(\mu_{eff} + 2)/(D + \mu_{eff} + 5)$
Damping parameter	d_σ	$1 + c_\sigma + 2max\{\sqrt{(\mu_{eff} - 1)/(D + 1)} - 1, 0\}$
Euclidian norm	$E\lVert\mathcal{N}(0, I)\rVert$	$\sqrt{D}(1 - \frac{1}{4D} + \frac{1}{21D^2})$
Discount parameter	c_c	$(4 + \frac{\mu_{eff}}{D})/(4 + D + 2\frac{\mu_{eff}}{D})$
Learning rates	c_1 c_μ	$2/((D + 1.3)^2 + \mu_{eff})$ $min\{1 - c_1, 2(\mu_{eff} - 2 + \frac{1}{\mu_{eff}})/((D + 2)^2 + 2\frac{\mu_{eff}}{2})\}$

19.3.1.3 Differential Evolution

Differential evolution (DE) is a search method that uses a population of candidate solutions $s_i, i = 1, \dots, N$ which is cyclically exposed to the operators mutation, crossover, and selection [56]. Other than the GA, DE's mutation variant is not based on adding a random perturbation to an existing solution but adding a perturbation formed from other solutions in the population. For each candidate solution s_i with D components, a mutant vector v_i is produced with

$$v_{i,k} = s_{r_1,k} + F_k \cdot (s_{r_2,k} - s_{r_3,k}),\ k = 1, \dots, D, \tag{19.13}$$

where $r_1, r_2, r_3 \in \{1, \dots, N\}$ are random indices different from i and $F_k \sim U(F_{min}, F_{max})$ is an amplification factor. The algorithm proceeds in creating a new solution via crossover of the candidate solution s_i and the mutant v_i. This new solution u_i is composed as follows:

$$u_{i,k} = \begin{cases} v_{i,k} & if\ x_k \leq p_{Cr}\ or\ k = l \\ s_{i,k} & else. \end{cases} \tag{19.14}$$

Here, $x_k \in [0, 1]$ is a uniform random number and p_{Cr} is the crossover probability. The index l is randomly sampled from $\{1, \dots, N\}$ and ensures that at least one component of the mutant vector is inherited to u_i. Finally, the new solution u_i is evaluated by the fitness function and selected to replace s_i in the population if it has a better fitness value. The algorithm repeats the cycle until a termination criterion is reached. DE's parameters are listed in Table 19.3.

Table 19.3 Parameters of DE

Parameters	Symbol	Settings
Size of population space	N	25
Amplification factor lower bound	F_{min}	0.2
Amplification factor upper bound	F_{max}	0.8
Crossover probability	p_{Cr}	0.2

Table 19.4 Parameter of ES1P1

Parameter	Symbol	Setting
Mutation strength	σ_k	$1(\forall k)$

19.3.1.4 (1+1)-Evolutionary Strategy

The (1+1)-evolutionary strategy (ES1P1) is one of the oldest and simplest optimization techniques among the evolutionary algorithms. It goes back to [47, 50] who investigated ES1P1 with binomial mutation and further implemented it with Gaussian mutation. The simplicity of ES1P1 comes from operating on just one single solution s from which just one offspring s_{new} is created. In case of Gaussian mutation, the new solution is given by

$$s_{new,k} = s_k + \sigma_k x, \ \ k = 1, \ldots, D \tag{19.15}$$

with σ_k being the mutation strength, $x \sim \mathcal{N}(0, 1)$ a random perturbation, and D the number of solution components. If the quality of s_{new} which is given by the objective function exceeds the quality of s, s_{new} is selected to replace s. Otherwise s is retained. This procedure of mutation followed by selection is iteratively repeated until a termination criterion is met [4]. For the parameter of ES1P1's see Table 19.4.

19.3.1.5 Shuffled Complex Evolution

Suggested by Duan et al. [15], the shuffled complex evolution (SCE) synthesizes four concepts in global optimization: an integration of probabilistic and deterministic search steps, a clustering of solutions into complexes, systematic evolution in direction of optima, and competitive evolution. The algorithm begins with defining a number P of complexes that each will include M solutions. This is followed by a random initialization of a population of solutions $s_i, i = 1, \ldots, N$ with $N = P \cdot M$ which are ordered by their fitness function value from best to worst quality. The population is then partitioned into complexes $A_l, l = 1, \ldots, P$ which all undergo the competitive complex evolution procedure (CCE). In the CCE, the solutions $s_j, j = 1, \ldots, M$ in a complex A_l are assigned probabilities p_j,

$$p_j = 2\,\frac{M + 1 - j}{M(M + 1)}, \ \ j = 1, \ldots, M. \tag{19.16}$$

Table 19.5 Parameters of SCE

Parameters	Symbol	Settings
Population size	N	25
Number of complexes	P	5
Complex size	M	5
CCE iterations	β	5
Number of parents	q	3
Number of offspring	α	3

Based on these probabilities, q parent solutions $u_1, \ldots, u_q$ are selected to form a sub-complex B in which, again, the solutions are ordered based on the fitness function value. From the $q-1$ best solutions in B, the centroid of the sub-complex is computed as

$$g = \frac{1}{q-1} \sum_{t=1}^{q-1} u_t. \tag{19.17}$$

A new solution r is generated as the reflection of u_q in g by $r = 2g - u_q$. If r is not a feasible solution in the search space, a new solution r is randomly generated within the smallest hypercube H that contains complex A_l. Subsequently, if the fitness of r is better than the one of u_q, r will replace u_q. However, if r has a worse fitness, a new solution c, called contraction, is generated by $c = 0.5(g + u_q)$. Anew, c will be substituted by a random solution inside of H if it exceeds the boundaries of the search space and c will then replace u_q if it has higher fitness. How often offspring in the form of r or c is generated from B is controlled by the parameter α. The parameter β decides the number of times a sub-complex B is selected from A_l. After all complexes A_l have undergone the CCE, they are merged and the order of the solutions s_i is updated. The algorithm restarts by dividing the solutions into complexes of equal size and proceeds in the described manner until a termination criterion is met [15]. Table 19.5 lists the parameters of SCE and their adjusted settings.

We note that other forms of evolutionary search are embodied in the scatter search and path relinking methods. Reviews of these methods and their successes in applications can be found in [24, 48].

19.3.2 Swarm Intelligence Algorithms

Metaheuristics based on swarm intelligence take inspiration from the social behaviour of animals such as ants, bees, schooling fish, or flocking birds. The sets of solutions which these algorithms operate on are considered as swarms. By certain self-organizing mechanisms, information is shared among the individuals of the swarm in order to take up optimal states. Four algorithms of this class are described below.

Table 19.6 Parameters of PSO

Parameters	Symbol	Settings
Number of particles	N	25
Inertia weight	ω	$\in [0.1, 1.1]$
Acceleration coefficients	ϕ_1	1.49
	ϕ_2	1.49

19.3.2.1 Particle Swarm Optimization

Since its introduction by Eberhart and Kennedy [16], particle swarm optimization (PSO) has been a very active field of study with a wide variety of applications [44]. The metaheuristic inspired by the social behaviour of bird flocks and fish schools begins with the random placement of initial D-dimensional solutions $s_i, i = 1, \ldots, N$, the so-called particles, in the search space of the objective function. The particles are evaluated by the objective function and their positions are stored in D-dimensional vectors $p_i, i = 1, \ldots, N$ of previous best solutions. In each iteration, these will be updated if the respective function value improves. Furthermore, the particles are defined by their D-dimensional velocities $v_i, i = 1, \ldots, N$ which determine the movement of the particles in exploration of the search space. In each cycle of the algorithm and for each particle s_i, a new velocity is calculated by

$$v_i = \omega v_i + \phi_1 u_1 \circ (p_i - s_i) + \phi_2 u_2 \circ (p_g - s_i), \; u_1, u_2 \sim U(0,1)^D. \qquad (19.18)$$

Here, ω is termed the inertia weight and ϕ_1 and ϕ_2 are the acceleration coefficients. Vector p_g is the best position among a randomly selected subset of one or more previous best positions p_i and $\circ$ stands for the element-wise multiplication. In the second term of Eq. (19.18), the search direction of particle s_i is adjusted towards its historically best position, whereas the third term guides the particle in direction of other good solutions. After finding the velocity v_i, the new position of a particle is given by

$$s_i = s_i + v_i. \qquad (19.19)$$

The algorithm restarts the cycle of finding new solutions until a certain termination criterion is met [16]. The parameters of PSO are summarized in Table 19.6.

An alternative version of this approach called Cyber Swarm Optimization [65] has been shown to possess advantages over PSO, but introduces a memory that requires additional parameters.

19.3.2.2 Ant Colony Optimization

Ant colony optimization (ACO) was originally designed for the solution of combinatorial optimization problems in the early 1990s but [54] adapted the basic ideas of this metaheuristic to extend its applicability to the continuous domain. ACO is inspired by the foraging behaviour of ants. Firstly, ants explore the territory around their nest in a random manner. In case they find food, they examine it and bring it back to the nest. On their paths, the ants put pheromone traces which, according to their intensity, give information about quality and quantity of the food source. Other ants' search for food is guided by previously deposited pheromone and they, again, put their own pheromone trace on their way back to the nest. As the pheromone evaporates, its concentration will decrease on less frequented paths, whereas on shorter paths to better food sources, the concentration will rise. Since the ants are more likely to choose paths with higher pheromone concentration, this mechanism enables them to effectively exploit food sources. Analogously to the ants' random search for food, the ACO procedure starts with a random initialization of N solutions $s_1, \ldots, s_N$ which each represent a path to one food source. The fitness function value assigned to a solution represents the path length. The solutions are stored in the solution archive where they are ranked from best to worst. This phase is followed by a loop in which new solutions are calculated. In order to guide the ants during their search process, probability density functions (PDFs) with Gaussian kernel are dynamically generated based on the previous solutions. For each of the D variables of the optimization problem, the PDF is given by

$$G_k(x) = \sum_{i=1}^{N} \omega_i g_{i,k}(x) = \sum_{i=1}^{N} \omega_i \frac{1}{\sigma_{i,k}\sqrt{2\pi}} \exp\left(-\frac{1}{2}\left(\frac{x-\mu_{i,k}}{\sigma_{i,k}}\right)^2\right), \quad k = 1, \ldots, D \tag{19.20}$$

with ω being the vector of weights, μ_k and σ_k being the vector of means and the vector of standard deviations, respectively, of the Gaussian functions g_k. The values of the parameters μ_k, σ_k, and ω are calculated from the solutions in the solution archive. The vector of means is simply set as

$$\mu_k = \{\mu_{1,k}, \ldots, \mu_{N,k}\} = \{s_{1,k}, \ldots, s_{N,k}\}. \tag{19.21}$$

The weights ω_i are defined as values of the Gaussian function with argument i, mean 1, and standard deviation qN, i.e.,

$$\omega_i = \frac{1}{qN\sqrt{2\pi}} \exp\left(-\frac{1}{2}\left(\frac{i-1}{qN}\right)^2\right), \quad i = 1, \ldots, N. \tag{19.22}$$

Here, q serves as a parameter of diversification. A small value for q leads to preferring the best-ranked solutions, whereas q being large yields more uniform weights. The standard deviations $\sigma_{i,k}$ are based on the average distance of solution s_i to all other solutions:

$$\sigma_{i,k} = \xi \sum_{j=1}^{N} \frac{|s_{j,k} - s_{i,k}|}{N-1}, \quad \xi > 0. \tag{19.23}$$

Table 19.7 Parameters of ACO

Parameters	Symbol	Settings
Archive size	N	25
New solutions per iteration	T	40
Diversification rate	q	0.5
Evaporation rate	ξ	1

The search is less biased towards points which were already explored if the evaporation rate ξ is higher. Thereby, the speed of convergence decreases. In practice, the sampling process for the predefined number T of new solutions to be generated simplifies to two steps. Firstly, upon the probabilities p_i,

$$p_i = \frac{\omega_i}{\sum_{j=1}^{N} \omega_j}, \; i = 1, \ldots, N, \tag{19.24}$$

the Gaussian functions g_i are randomly selected. Secondly, for each of the D variables, a new value is generated from the function $g_{i,k}$, $k = 1, \ldots, D$. A cycle ends with the pheromone update. The T new solutions are evaluated and added to the solution archive. Now the solution archive contains $N + T$ solutions which are re-ranked and the worst T solutions are discarded [54]. Table 19.7 gives a summary of the ACO parameters.

19.3.2.3 Artificial Bee Colony Algorithm

Proposed by Karaboga and Basturk [35], the artificial bee colony algorithm (ABC) is based on a replication of the foraging behaviour of honey bees. For the optimization, three sorts of bees are used: employed bees, onlooker bees, and scout bees. The algorithm begins with a random initialization of N positions in the D-dimensional search space which can be seen as initial food sources. The corresponding solution quality (fitness function value) indicates the nectar amount that this food source yields. After the initialization phase, the main loop of the procedure starts. The employed bees which are each associated with one food source position create new solutions by modifying their current positions. A new position found will replace the former one if its nectar amount is higher. In nature, the information gathered by the employed bees is passed on to the onlookers by performing a waggle dance. Depending on the intensity of the waggle dance, the more onlookers will be recruited to foraging at the promoted source. Algorithmically, this is realized by the computation of probabilities which are proportional to the nectar amounts of the food sources. Based on these probabilities an onlooker will select a food source and produce a modified position which, again, will replace the old position if its nectar amount is higher. New source positions v for employed and onlooker bees are created from old positions s by the formula

$$v_i = s_i + \phi_i(s_i - s_j), \tag{19.25}$$

Table 19.8 Parameters of ABC

Parameters	Symbol	Settings
Number of employed bees	N	25
Number of onlookers	N_o	N
Acceleration parameter	a	1
Abandonment limit parameter	L	$0.6 \cdot N \cdot D$

where $i = 1, \ldots, N$, $j \in \{1, \ldots, N\}$, $j \neq i$, and $\phi \sim U(-a, a)^D$, with a being the predefined acceleration coefficient and D the search space dimension. In case a new source position exceeds the solution space boundaries in a certain dimension, the solution will be adjusted by setting it to the boundary in this dimension. A decreasing difference between the positions s_i and s_j ensures the convergence of the algorithm as the perturbation around s_i is also reduced. A food source will be abandoned if the search for new positions around this source by employed bees and onlooker bees does not lead to a better solution quality over a given number of cycles L. If the Abandonment limit parameter L is exceeded for a source i, a scout bee will be send out, i.e., source i will be replaced by a new source that has been randomly generated inside the boundaries of the search space. While the search of employed bees and onlookers can be seen as the algorithmic mean of search space exploitation, the scout search enables the algorithm to perform an exploration of undiscovered areas of the search space [35]. For ABC parameters and their settings see Table 19.8.

19.3.2.4 Firefly Algorithm

The flashing behaviour of fireflies which functions as a mean to attract mating partners inspired the firefly algorithm (FA) [63]. In FA, a candidate solution is considered as a unisexual firefly and its objective function value represents the brightness at which the firefly emits light. The attractiveness of a firefly is proportional to its brightness such that, for two fireflies, the less bright one will move in direction of the brighter one. With the increase of mutual distance, the perceptible brightness decreases and thereby the attraction. For a given firefly, if there are no brighter ones, the firefly changes its position randomly. At the beginning of the algorithm, the randomly initialized set of candidate solution s_i, $i = 1, \ldots, N$ is sorted by their solution quality. After this, the cyclic creation of new solutions starts. Each solution is compared to all other solutions in the set. If a solution s_i has a lower quality (brightness) than another solution s_j, $j \neq i$, it will move towards s_j which is realized as

$$s_i = s_{i-1} + \beta_0 e^{-\gamma r_{ij}^2}(s_j - s_{i-1}) + \alpha x, \;\; x \sim U(-\delta, \delta)^D, \tag{19.26}$$

with D being the dimension of the search space. In Eq. (19.26), the second term refers to the distance-dependent attractiveness which is defined by the base

Table 19.9 Parameters of FA

Parameters	Symbol	Settings
Number of fireflies	N	25
Base attractiveness	β	2
Light absorption coefficient	γ	1
Randomization parameter	α	0.2
Range parameter	δ	0.05

attractiveness β_0, the light absorption coefficient γ, and the Euclidean distance r_{ij} between s_i and s_j. The third term in Eq. (19.26) simulates random motion as a mean of intensification with the randomization parameters α and the range parameter δ. A cycle finishes with the re-evaluation of the solutions such that newly constructed better solutions replace old ones [63]. All parameters of FA with their settings are listed in Table 19.9.

19.3.3 Metaheuristics Inspired by Human Behaviour

Another class of MHs is characterized by taking inspiration from human behaviour. This includes analogies to individual conduct, group dynamics, and interaction among social systems. Four of these algorithms are described below.

19.3.3.1 Cultural Algorithm

Cultural algorithms (CAs) were introduced by Reynolds [49] and are conceptually based on the social evolution of human beings. These metaheuristics contain a population space and a belief space which share information among each other in order to guide the search process. At the beginning of the algorithm, N individuals of the population space are randomly initialized inside the boundaries of the D-dimensional search area. After the evaluation of these individuals by the fitness function and their ordering, a preset share of best individuals is used to update the information of the belief space. The algorithm applied in the simulation study in Sect. 19.4 uses a situational and a normative component in the belief space. The situational component stores the current best solution and its cost. The normative component is represented by D intervals. These intervals match the range which is momentarily believed to be the most appropriate for each of the D variables, i.e., maximum and minimum value in each direction of the population space [1]. For cyclically creating new solutions of the cost function, the individuals in the population space are each transformed by using information of the belief space. For each variable $k, k = 1, \ldots, D$ of an individual $i, i = 1, \ldots, N$, a value ds is calculated by

$$ds_{ik} = \alpha \cdot r_k \cdot |x|, \;\; x \sim N(0, 1), \tag{19.27}$$

Table 19.10 Parameters of CA

Parameters	Symbol	Settings
Size of population space	N	25
Size of belief space		$\lceil 0.35N \rceil$
Scaling parameter	α	0.3

Table 19.11 Parameters of HS

Parameters	Symbol	Settings
Size of harmony memory	N	25
Number of new solutions	N_{new}	20
Harmony memory considering rate	$HMCR$	0.90
Pitch adjustment rate	PAR	0.10
Bandwidth	σ	0.02

where α is a scaling parameter and r_k the currently believed best range of variable k stored in the normative component. The value ds_{ik} is added to solution variable s_{ik} if s_{ik} is smaller than the according variable from the situational component. However, if s_{ik} is larger, ds_{ik} will be subtracted from s_{ik}. At the end of each cycle, the new individuals in the population space are utilized to update the components of the believe space. The procedure continues until a termination criterion is met [49]. For a list of CA parameters see Table 19.10.

19.3.3.2 Harmony Search

The metaheuristic algorithm of harmony search (HS) was developed by Geem et al. [21] and models the process of finding musical harmony among different musical instruments. The idea behind the algorithm is to consider a solution $s_i, i = 1, \ldots, N$ in the search space as a set of sounds jointly played from D different instruments where the variable values represent the pitches. The aesthetic quality of such a set of sounds is evaluated by the cost function and each iteration can be seen as a practice session. The algorithm starts by initializing a predefined number of N solution, the so-called harmony memory. After ordering these solutions by their quality, the search for novel N_{new} solutions begins. Determining a new element of a solution s_{ik} is steered by two parameters: the Harmony Memory Considering Rate ($HMCR$) and the Pitch Adjustment Rate (PAR). If a random number $x, x \sim U(0, 1)$ exceeds the HMCR, then a random value inside the boundaries of the search space is assigned to s_{ik}. However, if x does not exceed the HMCR, s_{ik} will randomly take a value from a solution in the harmony memory at index k. In order to escape local optima, the value of s_{ik} can be shifted by a bandwidth $\sigma \cdot z, z \sim N(0, 1)$, where σ is another parameter of the algorithm. Whether this mutation operation is performed is based on the probability given by PAR. Subsequently, the novel solutions will replace old solutions in the harmony memory if their respective quality is higher. The practising process is re-started until a termination criterion is reached [21]. See Table 19.11 for a summary of HS parameters.

19.3.3.3 Imperialist Competitive Algorithm

The imperialist competitive algorithm (ICA) is a metaheuristic which is based on mechanisms of imperialistic competition [3]. It starts from a randomly initialized population of size N in which each individual is considered as a country. A predefined number of countries N_{imp} that have lowest cost are then chosen to be imperialistic states, whereas the remaining $N - N_{imp}$ countries are the so-called colonies. In order to create the initial empires, the colonies will be assigned randomly to an imperialist. The higher the power of an imperialist, i.e., the higher its solution quality, the more colonies will be assigned to it. After this initialization phase, the assimilation phase begins. Here, analogously to imperialists economically improving their colonies, the colonies are moved in direction of the imperialist by a distance $x, x \sim U(0, \beta d)$, where $\beta > 1$ and d is the distance between colony and imperialist. To enable the procedure to explore more points around the imperialist, an additional deviation from the direction is introduced which is defined by an angle $\theta, \theta \sim U(-\gamma, \gamma)$. The assimilation phase is followed by the revolution phase which is similar to the mutation operator in GA [31]. The position of a colony is changed to avoid the early convergence towards the imperialist. This step is governed by the revolution probability p_r which determines whether a colony undergoes revolution, and the revolution rate μ which determines how many variables of a colony will be changed randomly. After the revolution phase, the colonies are re-evaluated and if a colony has higher solution quality than its imperialist, their positions will be exchanged. The total power of an empire is calculated by the power of the imperialist plus a fraction of the mean colony power

$$TP_j = Cost(imperialist_j) + \xi\, mean\{Cost(colonies\ of\ empire_j)\}, \tag{19.28}$$

where $0 < \xi < 1$ and $j = 1, \ldots, N_{imp}$. The subsequent imperialistic competition phase uses the total power to calculate the proportional possession probability p_j for each empire. The empire with the highest value $q_j = p_j - r_j$, where $r_n \sim U(0, 1)$ obtains the power over the weakest of all colonies. Eventually, the imperialistic competition causes powerless empires, i.e., empires that lost all their colonies, which will be eliminated from the population of solutions. The algorithm has converged and is terminated if all empires except the most powerful have collapsed and all colonies have the same position and cost as the imperialist. Table 19.12 presents the parameters of ICA with their settings.

19.3.3.4 Teaching–Learning-Based Optimization

Rao et al. [46] proposed another metaheuristic technique named teaching–learning-based optimization (TLBO). This population-based algorithm functions without the use of any adjustable parameters but is only driven by the properties of the set of solutions. The set of solutions is assumed to be a class of students and the current best solution is considered as the teacher. The objective function yields the state

Table 19.12 Parameters of ICA

Parameters	Symbol	Settings
Number of countries	N	25
Number of imperialists	N_{imp}	5
Assimilation coefficient	β	1.50
Range parameter	γ	$\pi/4$
Colonies mean coefficient	ξ	0.20
Revolution probability	p_R	0.05
Revolution rate	μ	0.20

Table 19.13 Parameter of TLBO

Parameter	Symbol	Setting
Class size	N	25

of knowledge of a solution. The algorithmic design is divided into two phases: the teacher phase and the learner phase. In the teacher phase, the students are moved into the direction of the teacher simulating the teacher sharing his or her knowledge with the class to bring it to his or her own level of knowledge. Computationally, this is achieved by determining the difference between the teacher solution T and the mean of all solutions

$$\Delta_k = r_k(T_k - T_F \cdot \mu_k), \; k = 1, \dots, D, \tag{19.29}$$

where $r_k \sim U(0, 1)$, T_F is the teaching random factor with $T_F \in \{1, 2\}$ and μ_k is the mean of the solution variables in dimension k. Each existing solution $s_i, i = 1, \dots, N$ is modified in each dimension by

$$s_{ik,new} = s_{ik} + \Delta_k, \tag{19.30}$$

and the new solution $s_{i,new}$ will replace s_i if the quality of the new solution is higher. In the subsequent learner phase, the students exchange knowledge among each other. Therefore, for a solution s_i, a different solution s_j from the class is randomly selected and the worse of the two is subtracted from the better to a difference δ. Again, a new solution is generated based on s_i such that

$$s_{ik,new} = s_{ik} + \delta_k. \tag{19.31}$$

As in the teacher phase, the new solution will replace s_i if its solution quality is higher. The algorithm repeats teacher and learner phase until a termination criterion is reached [46]. For a list of TLBO's parameters refer to Table 19.13.

19.3.4 Other Metaheuristics

Beyond the evolutionary and swarm intelligence approaches described above, we also apply three additional algorithms including simulated annealing.

Table 19.14 Parameters of SA

Parameters	Symbol	Settings
Initial temperature	T_0	100
Cooling parameter	c	0.95
Reannealing interval		100

19.3.4.1 Simulated Annealing

First introduced by Kirkpatrick et al. [36], the simulated annealing algorithm (SA) has since become a widely applied optimization technique. It is inspired by the annealing process for metal or glass which aims at attaining states of lower energy by following a cooling schedule. The key principle of this single-solution method is to allow for a new solution that has worse objective function value than the previous solution. Such moves in the iterative search are only executed with a certain probability that tends to decrease by the iteration number. After randomly initializing a solution s, at each step a new solution s_{new} is sampled from the neighbourhood of s. The new solution s_{new} will replace s if its quality is higher. If the quality of s_{new} is lower, it can still be accepted with a probability proportional to $exp(-\varDelta/T)$. Here, $\varDelta$ is the difference of the function values of s_{new} and s and T is the current temperature. The temperature is decreased during the progress of the algorithm. In the simulation study in Sect. 19.4, an exponential cooling schedule of

$$T = T_0 \cdot c^l \tag{19.32}$$

is assumed with T_0 being the initial temperature, c the cooling parameter, and l the annealing parameter. The annealing parameter l is equal to the iteration number until, after a predefined number of iterations, reannealing is initiated. Reannealing allows for the annealing parameter to take lower values than the number of iterations which leads to an increase in temperature. Explicitly, this is realized by

$$l_k = \log\left(\frac{T_0}{T_k}\frac{\max_j\{x_j\}}{x_k}\right), \ j = 1, \ldots, D, \tag{19.33}$$

where, for component $k, k = 1, \ldots, D$, l_k is the annealing parameter, T_k is the current temperature, and x_k is the estimated gradient of the objective [32]. Since the temperature is decreased up to reannealing, worse solutions are more likely to be accepted at the beginning of the search which leads to an exploration of the search space. However, at a later stage, the algorithm converges to an iterative improvement of s and, thereby, is able to find a local optimum [6]. A summary of SA's parameters is shown in Table 19.14.

19.3.4.2 Biogeography-Based Optimization

Proposed by Simon [53], biogeography-based optimization (BBO) is another population-based metaheuristic which is inspired by the migration behaviour of species between habitats. Each solution of the algorithm can be seen as a habitat and its solution quality given by the fitness function is considered as the habitat suitability index (HSI). The elements of a solution defining the HSI are called suitability index variables (SIVs) in this context. They refer to natural factors such as rainfall, temperature, diversity of vegetation, and area of the habitat. Habitats with high HSI are prone to have a high emigration rate due to the fact that they accommodate a high number of species. A high HSI also indicates a low immigration rate since the habitat is almost saturated with species. Habitats with low HSI have high immigration rates since they are only sparsely populated. On the other hand, their emigration rates are low according to the low number of species in them. Algorithmically, this mechanism is seized by the migration operator that allows solutions to exchange their information and, thereby, functions as a tool for exploration and exploitation of the search space. The migration operator comprises the immigration rates

$$\lambda_i = I\left(1 - \frac{i}{N}\right),\ i = 1, \ldots, N \tag{19.34}$$

and the emigration rates

$$\mu_i = E\left(\frac{i}{N}\right),\ i = 1, \ldots, N \tag{19.35}$$

with I and E being the maximum immigration and emigration rate, respectively, and N the predefined number of habitats. Before the migration operator is applied, the solutions are sorted from best to worst HSI whereby they are affiliated with their adequate migration rates. For each element of a solution $s_{ik}, i = 1, \ldots, N, k = 1, \ldots, D$, it is randomly decided on the basis of the immigration rate λ_i whether it will be changed. The element s_{ik} is replaced by an element of another solution $s_{jk},\ j = 1, \ldots, N,\ j \neq i$ which is randomly selected on the basis of the emigration rates mu_i. This can be formally expressed as

$$s_{ik} = s_{ik} + \alpha(s_{jk} - s_{ik}), \tag{19.36}$$

where α is a parameter of the algorithm.

Natural habitats are likely to undergo cataclysmic events such as floods, disease, or unusual immense immigration of species which might change the HSI drastically. The BBO algorithm adapts this fact by the mutation operator. An element of a solution s_{ij} is mutated with mutation probability m and by a certain scale $\sigma, \sigma \in (0, 1)$, i.e.,

$$s_{ik} = s_{ik} + x,\ x \sim N(0, \sigma). \tag{19.37}$$

Table 19.15 Parameters of BBO

Parameters	Symbol	Settings
Number of habitats	N	25
Maximum immigration rate	I	1
Maximum emigration rate	E	1
Migration parameter	α	0.90
Mutation probability	m	0.10
Mutation scale	σ	0.02
Elitism parameter	e	$\lceil 0.2N \rceil$

This allows for a diversification of the set of solutions. BBO also incorporates elitism whereby a predefined number e of best solutions remain unchanged in each iteration [53]. Table 19.15 presents a listing of BBO's parameters.

Another metaheuristic using extended ideas of migration is known as migrating birds optimization approach; see [51] and the references therein.

19.3.4.3 Backtracking Search Optimization

Suggested in [12], the backtracking search optimization algorithm (BSA) is another example of an evolutionary algorithm. The algorithm incorporates the three fundamental genetic operators selection, mutation, and crossover, yet the mechanism for generating new individuals is unique. Firstly, a population P of given size N and number of variables D is randomly initialized inside the predefined boundaries. After that, the selection-I stage begins by generating a historical population $oldP$ for the calculation of the search direction. The initial $oldP$ is again randomly generated but, at the start of each iteration, $oldP$ will be optionally redefined as $oldP := P$ if the condition $a < b$, $a, b \sim U(0, 1)$ is fulfilled. The historical population is stored until it is overwritten which characterizes BSA as a memory-based algorithm. The selection-I stage ends with a random permutation of $oldP$. In the subsequent mutation phase, new individuals ($Mutants$) are generated by

$$Mutant = P + F(oldP - P), \tag{19.38}$$

where the parameter F adjusts the amplitude of the search direction matrix ($oldP - P$). By using the historical population for computing the search direction matrix, BSA tries to seize its experience advantageously. Crossover is performed in the next stage in which a final trial population T is produced. Initially, T is set to be equal to $Mutant$ whereupon the individuals of T are evolved. A binary matrix map of size $N \times D$ is randomly generated where each cell entry represents a position in T. If $map_{ik} = 1, i = 1, \ldots, N, k = 1, \ldots, D$, then T_{ik} is replaced by the corresponding P_{ik}. Row-wise, map contains either only zeros or a random amount of ones which is determined by the function

$$\lceil mixrate \cdot c \cdot D \rceil , \tag{19.39}$$

Table 19.16 Parameters of BSA

Parameters	Symbol	Settings
Population size	N	25
Scale factor	F	$3 \cdot d,\ where\ d \sim N(0,1)$
Mixrate		1.00

where $c \sim U(0,1)$, $mixrate$ is a predefined parameter, and $\lceil\rceil$ is the ceiling function. The stage ends with adjusting positions that exceed the predefined boundaries which might have been produced during the mutation phase. In the final selection-II stage, an individual P_i is replaced by individual T_i if the corresponding fitness value of T_i is better than the one of P_i. At the end of an iteration the so far best fitness value, and the associated global optimizer are determined. The algorithm repeats the stages selection-I, mutation, crossover, and selection-II iteratively until a certain termination condition is fulfilled [12]. See Table 19.16 for a listing of the parameters of BSA.

19.4 Experimental Study

In this experimental study, the MHs presented in Sect. 19.3 are compared with regard to their performance in finding weights of weighted average ensembles. Determining the weights for a weighted average ensemble optimal to a certain metric is a continuous optimization problem as stated in Definition 19.1. Therefore, the application of MHs is appropriate and advantageous in a sense that they allow for different objective functions, i.e., metrics, to optimize the weights to.

The following sections describe the experimental environment in terms of data and parametric settings. Conclusively, the results are presented.

19.4.1 Data Sets and Base Models

From the UCI Machine Learning Repository [2], 15 multivariate data sets with binary classification task were selected. This selection was assorted in order to represent a broad range in the number of instances, the number of attributes, and the percentage of the majority class. Table 19.17 lists the data sets and their aforementioned properties. Every data set represents a binary classification problem. For example, in the Credit Approval data set, the binary target variable indicates whether a credit applicant who was granted a loan eventually paid back the credit in full or defaulted (i.e., failed to do so). Applicants of the former and latter type are considered good and bad risks, respectively. The attributes capture characteristics of the borrowers such as data from the application form. The task of the classification model is to infer a functional relationship between the attributes and the binary target

Table 19.17 Fifteen data sets with binary classification tasks for which machine learning models are estimated

Data sets	# Instances	# Attributes	% Majority class	# Base models
Fertility	100	10	88.0	198
Cleveland heart	303	14	54.1	198
Ionosphere	351	34	64.1	195
Credit approval	690	15	55.5	183
Pima Indians diabetes	768	8	65.1	198
QSAR biodegradation	1054	41	66.3	195
Seismic bumps	2584	16	93.4	198
Chess (KRKPA7)	3196	36	52.2	195
Spambase	4601	57	60.6	195
Wilt	4839	6	94.6	198
EEG eye state	14,980	15	55.1	185
MAGIC gamma telescope	19,020	11	64.8	194
Bank marketing	45,211	17	88.3	173
Adult	48,842	14	76.1	176
Statlog shuttle	58,000	9	78.6	195

that facilitates the lender to discriminate between good and bad risks (i.e., predict group memberships) using application form data (i.e., attributes). In a similar way, the Adult data set represents a task where a classification model aims to predict the income situation among US citizens. More specifically, the binary target variable captures whether the income of a household exceeds 50,000. The attributes refer to census variables such as residential status or the number of household members. The Pima Indians Diabetes data set is associated with a classification task to predict whether a person, with Pima Indian heritage, has diabetes whereby the attributes capture diagnostic measures. In the same way, each data set represents a specific classification problem, the common denominator being that the prediction target is always binary and that the attributes always embody information that is believed to determine the class membership.

For each data set, the binary classification task was modelled with eight different machine learning algorithms with R's caret package. The applied algorithms were support vector machine with polynomial kernel (svmPoly), support vector machine with radial kernel (svmRadial), neural network (nnet), regularized random forest (RRF), multi-layer perceptron (mlp), k-nearest neighbour (knn) (using the Euclidean metric), gradient boosting machine (gbm), and C5.0 decision tree (C5.0). Adjusted parameters of these algorithms are presented in Table 19.18. Except for standardizing of numerical variables in each data set, no further pre-processing was carried out. The data sets were split into training (70%) and test sets (30%) in a way that the class ratio remained equal to the initial class ratio. Using fivefold cross-validation, models were estimated on the training sets to maximize the percentage of correctly classified instances and to produce class probabilities. The modelling process yielded libraries of base models with a maximum number of 198 elements.

Table 19.18 Eight machine learning algorithms with changing modelling parameters applied to build pools of a maximum of 198 classifiers per data set

Algorithm	Model parameters	# Models
svmPoly	$cost \in \{0.25, 0.5, 1\}$	27
	$scale \in \{0.001, 0.01, 0.1\}$	
	$degree \in \{1, 2, 3\}$	
svmRadial	$cost \in \{10^{-6}, 10^{-5}, \ldots, 10^{3}\}$	30
	$sigma \in \{0.01, 0.033, 0.05\}$	
nnet	$size \in \{0, 1, 2, \ldots, 5\}$	18
	$decay \in \{0, 0.001, 0.1\}$	
RRF	$mtry \in \{0.5, 1, 2\} * \lfloor\sqrt{no.of\ vars}\rfloor$	27
	$coefReg \in \{0.01, 0.505, 1\}$	
	$coefImp \in \{0, 0.5, 1\}$	
mlp	$size \in \{1, 3, 5, \ldots, 19\}$	10
knn	$k \in \{3, 4, 5, \ldots, 30\}$	28
gbm	$n.trees \in \{100, 200, \ldots, 500\}$	30
	$interaction.depth \in \{1, 2, 4, 6, 8, 10\}$	
	$shrinkage = 0.1$	
	$n.minobsinnode = 10$	
C5.0	$model \in \{rules, tree\}$	28
	$winnow \in \{FALSE, TRUE\}$	
	$trials \in \{10, 20, \ldots, 70\}$	

However, some parametric setting excited errors during the modelling process which is why the number of base models is reduced for the respective data set (see Table 19.17).[1]

19.4.2 Metaheuristics and Fitness Function Settings

In order to build ensembles from the model libraries, we use the MHs described in Sect. 19.3 to select models and find model weights. GA, PSO, and SA are taken from the Matlab Optimization Toolbox [38], ES1P1 from [41], and BSA from [11]. All other MHs are available at the online archive [64].

Given that the question whether to maximize ensemble performance (i.e., predictive accuracy) or base model diversity (e.g., avoid error correlation) has been studied in prior work [43, 59, 61], we consider the ensemble selection objective an important experimental factor and include it in our study. In particular, we consider four objectives: the Brier score, the H-measure, the interrater agreement,

[1]The code to develop the base models and produce predictions for the above data sets is available at https://github.com/ringothom/MHE/tree/master/MHEModelClassifiers.

Table 19.19 Confusion matrix in binary classification

	Actual 1	Actual 0
Predicted 1	a	b
Predicted 0	c	d

and difficulty. The former two are measures of predictive performance, whereas the latter two capture diversity. For each data set and, respectively, base model library, we let each metaheuristic produce a solution, that is, an ensemble, for each of the four objectives. The ultimate goal of an ensemble and more generally predictive modelling is to develop an accurate forecast. Therefore, we benchmark the resulting ensembles in terms of their predictive performance using the area under the receiver-operating-characteristics-curve (AUC) as performance measure (see below). The AUC is a well-established indicator of classifier performance and widely used in the literature [18, 37]. The following sub-sections detail the criteria employed in the study.

19.4.2.1 Receiver-Operating-Characteristics Curve, AUC, and H-Measure

The ROC curve is a two-dimensional plot of the true positive rate $TPrate$ on the y-axis against the false positive rate $FPrate$ on the x-axis. These rates are calculated from the confusion table, which depicts cross-classifications of actual and predicted group memberships (see Table 19.19).

$$TPrate = \frac{a}{a+c} \,, \tag{19.40}$$

$$FPrate = \frac{b}{b+d} \,. \tag{19.41}$$

The points (1, 1) and (0, 0) in ROC space refer to a model that classifies all instances as positive and negative, respectively. In the point (0, 1), all instances are correctly classified. For a classifier h, the ROC curve can be generated based on confusion tables created at different thresholds for the probability outputs of h. Necessarily, this curve begins in (0, 0) and ends in (1, 1) and, of two classifiers, the one whose curve lies closer to (0, 1) is better. Figure 19.1 shows an exemplary ROC curve for three classifiers, C_1, C_2, C_3. The diagonal line represents a (naive) benchmark, which guesses classes at random. All three classifiers outperform the benchmark in that their ROC curves are consistently closer to the point (0,1). Consequently, the classifiers perform better than random and have succeeded in distilling some systematic patterns from the underlying data. Furthermore, the figure suggests that C_1 is the best classifier in the comparison. The corresponding ROC curve is always above that of C_2 and C_3. Given that the ROC curves of the latter intersect, the figure does not allow to conclude that one is consistently better than the other. The area under the ROC curve summarizes the whole curve in a scalar measure of classifier performance. The AUC ranges between 0.5 and 1 where higher values indicate better performance [27].

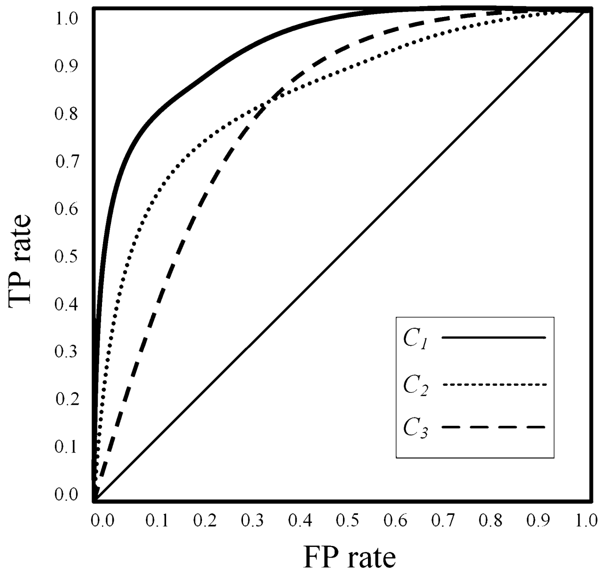


Fig. 19.1 ROC curve example

The H-measure (*hm*) is a measure of misclassification cost and closely related to the AUC but incorporates prior class probabilities to standardize inter-classifier comparisons. The H-measure takes values in [0, 1] where higher values indicate better predictive accuracy. A detailed derivation of the H-measure can be found in [27].

19.4.2.2 Brier Score

The Brier score (*brier*) is a measure for the calibration of probability forecasts. It is equivalent to the mean squared error and calculated as

$$BS = \frac{1}{M} \sum_{k=1}^{N} \left(p(x_k) - y_k \right)^2, \tag{19.42}$$

where p gives the predicted class probability of belonging to class 1, $y \in \{0, 1\}$ the actual outcome, and M the number of instances x. The Brier score takes values in the range $\{0, 1\}$, with 0 and 1 signalling maximal and worst accuracy, respectively [30].

19.4.2.3 Interrater Agreement

The interrater agreement (*ia*) is a diversity measure, which quantifies the extent to which base models in an ensemble agree in their (discrete) class predictions. It is calculated as

$$\kappa = 1 - \frac{\frac{1}{N}\sum_{k=1}^{M} \rho(x_k)(N - \rho(x_k))}{M(N-1)\bar{p}(1-\bar{p})}, \tag{19.43}$$

where $\bar{p}$ is the average accuracy of the individual classifiers given by

$$\bar{p} = \frac{1}{NM}\sum_{i=1}^{N}\sum_{k=1}^{M} \mathbb{I}\{h_i(x_k) = y_k\} \tag{19.44}$$

and $\rho(x_k)$ is the number of correct classifications among the N individual classifiers. $\kappa = 1$ is the upper bound of this measure and indicates that the classifiers agree on all instances x_k. If $\kappa \leq 0$, there is even fewer agreement among the classifiers than expected by chance. Hence, in order to create a diverse ensemble, one should choose a set of classifiers that minimizes κ [67].

19.4.2.4 Difficulty

The difficulty measure (*diff*) is another measure for ensemble diversity. It is defined as the variance of a random variable X that can take values in $\{0, \frac{1}{N}, \frac{2}{N}, \ldots, 1\}$ and indicates the share of N classifiers that correctly classify a random instance x. By computing the class prediction of the classifiers for a given data set, one can estimate the probability mass function. In case where the classification of the same instance is difficult, i.e., most classifiers do not predict correctly, whereas for all other instances the classification is easy, the distribution shows two separate peaks. The distribution has only one peak if the instances that are difficult for some classifiers are correctly predicted by other classifiers. If all classifiers predict equally, there is no definite peak in the distribution. Therefore, the variance of X provides information about the shape of the distribution and

$$\theta = Var(X) \tag{19.45}$$

is the difficulty measure. Ensemble diversity is increased if $\theta \in [0, 1]$ is decreased [67, p. 108].

19.4.3 Results

In the subsequent analysis of the ensembles created with the MHs, the scores, i.e., accuracy and diversity values, are compared regarding predictive accuracy, computational time, and weight vector sparsity. This is followed by the comparison of the MHs and a final contrasting juxtaposition of the best MHs and other ensemble techniques. All statistical comparisons are conducted by means of the Friedman rank sum test and the Friedman rank sum test with Finner adjustment for post hoc tests as it is recommended in [20].

19.4.3.1 Comparison of Scores

In order to compare the scores, the sets of weights produced by the MHs are used to compute test set predictions. For mutual comparison of predictive accuracy, the test set predictions are evaluated with the AUC measure and then averaged over the MHs. The test yields the following results:

$$\text{Friedman's chi-squared} = 8.04,\ \text{df} = 3,\ p\text{-value} = 0.045.$$

From Table 19.20, the Brier score ranks highest among the four measures indicating that using it as an objective, on average, it yields the most accurate ensembles throughout all data sets and MHs. The difference among the four measures is significant up to a 5%-significance level as the p-value of 0.045 implies. The post hoc analysis reveals that only the difference between Brier score and interrater agreement is significant; see Table 19.21.

Since computation time was recorded for each algorithmic setup, comparisons between scores are conductible in this aspect. Computation time was averaged over the metaheuristic for each score and data set in order to carry out the Friedman test. This shows the following results:

$$\text{Friedman's chi-squared} = 7.4,\ \text{df} = 3,\ p\text{-value} = 0.06.$$

Table 19.20 Average score rank in accuracy comparison over data sets from high (1) to low (4) AUC of ensembles

Brier	Diff	hm	ia
2.00	2.47	2.27	3.27

Table 19.21 Finner-corrected p-values of Friedman post hoc test regarding differences among scores in AUC of ensembles

	Brier	Diff	hm	ia
Brier		0.44	0.64	0.04
Diff	0.44		0.67	0.17
hm	0.64	0.67		0.10
ia	0.04	0.17	0.10	

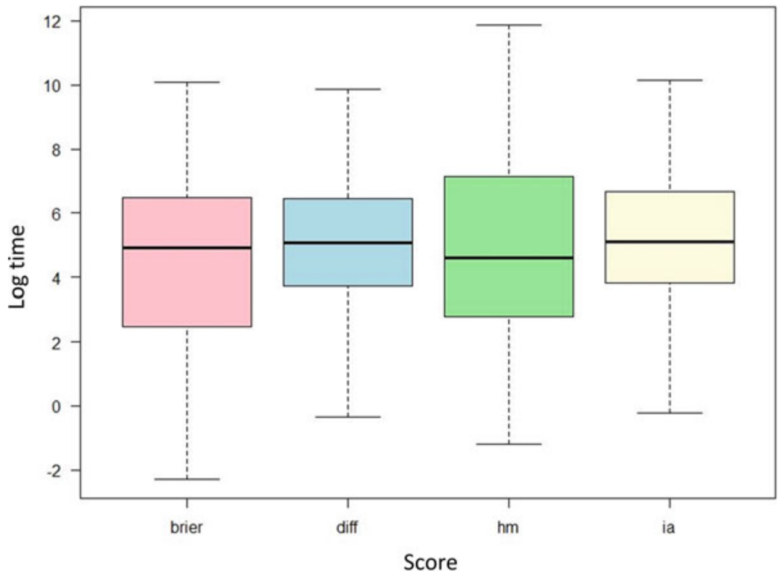


Fig. 19.2 Boxplots of computation times (log sec) for each score function

Table 19.22 Average score rank in time-wise comparison over data sets from slow (1) to fast (4)

Brier	Diff	hm	ia
1.93	3.07	2.80	2.20

Although providing a low *p*-value of 0.06, there is no significant difference between the scores regarding computation time. The difficulty score shows the lowest rank (Table 19.22) which indicates that optimizing to this measure demands the least time with respect to the given data sets. In Fig. 19.2, the boxplot for the Brier score shows the lowest values, whereas the boxplot for the H-measure ranges up to very high values making it less preferable over other measures. Another valuable property of the weight vector determined by a metaheuristic is its sparsity. Sparse weight vectors suggest that only a few base models are necessary to give accurate ensemble predictions. The score that provides the sparsest vectors is determined by averaging the percentage of zero weights for each score and data set over the 16 MHs. The Friedman test results in:

$$\text{Friedman's chi-squared} = 38.52,\ \text{df} = 3,\ p\text{-value} = 2.19\text{e}{-08}.$$

The difference between the scores in providing weight vector sparsity is highly significant given that the *p*-value equals 2.19e−08. From Table 19.23, the Brier

Table 19.23 Average score rank in sparsity comparison over data sets from high (1) to low (4) percentage of zero weights

Brier	Diff	hm	ia
1.20	2.80	4.00	2.00

Table 19.24 Finner-corrected p-values of Friedman post hoc test regarding differences among scores in sparsity

	Brier	Diff	hm	ia
Brier		0.00	0.00	0.11
Diff	0.00		0.02	0.11
hm	0.00	0.02		0.00
ia	0.11	0.11	0.00	

score ranks highest among the four fitness measures meaning that it yields the sparsest weight vectors. The post hoc analysis given in Table 19.24 reveals that only the H-measure differs from all the other measures. Brier score and interrater agreement and difficulty and interrater agreement, respectively, can be considered as pairwise equal.

In Fig. 19.3, the boxplot for the Brier score shows the highest median and, also, the widest interquartile range. The boxplot for the H-measure supports the result in Table 19.23: Optimizing to the H-measure yields larger ensembles in comparison to the other three measures. From the comparison of the four fitness measures, the Brier score can be favoured. It ranks highest in accuracy and weight vector sparsity and, regarding computation time, it shows the lowest results and its maximum values lie much lower than for the H-measure.

19.4.3.2 Comparison of Metaheuristics

In order to compare the performance of the 16 MHs, their differences among the four fitness measures are investigated. We will report the top three algorithms from 16 used in the experiment as "best" algorithms. Subsequently, their computation time and their produced weight vector sparsity are compared on the basis of the Brier score results. Comparing the MHs based on the Brier score results of the test sets yields the following results:

$$\text{Friedman's chi-squared} = 14.218,\ \text{df} = 15,\ p\text{-value} = 0.51.$$

The differences between the MHs in terms of the Brier score are not significant; the p-value is above the 5%-significance level. SCE, CMAES, and BSA rank best

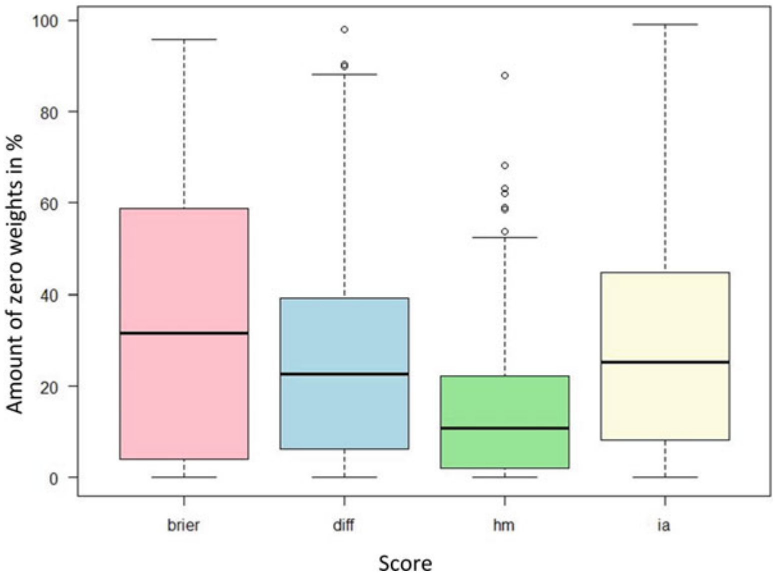


Fig. 19.3 Boxplots of percentage of zero weights for each score function

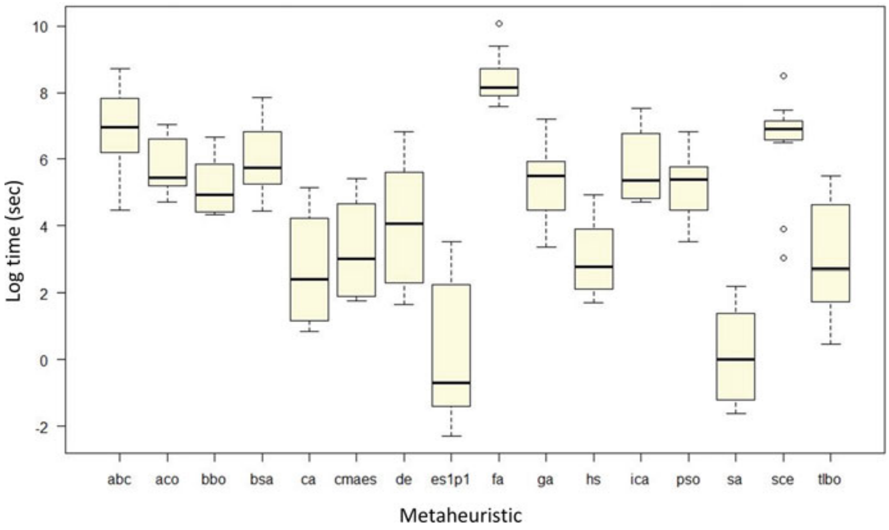


Fig. 19.4 Boxplots of computation times (log sec) for each MH

out of the 16 algorithms which can be seen in Table 19.25. The Friedman test for differences in the H-measure rankings of the test set predictions results in:

Friedman's chi-squared = 28.36, df = 15, p-value = 0.02

Table 19.25 Average rank of the MHs algorithms over data sets from high (1) to low (16) Brier score

ABC	ACO	BBO	BSA	CA	CMAES	DE	ES1P1	FA	GA	HS	ICA	PSO	SA	SCE	TLBO
7.53	6.20	8.20	9.40	9.33	9.60	7.00	7.93	7.93	8.93	9.33	9.33	8.87	9.07	10.60	6.73

Table 19.26 Average rank of the metaheuristics from high (1) to low (16) H-measure

ABC	ACO	BBO	BSA	CA	CMAES	DE	ES1P1	FA	GA	HS	ICA	PSO	SA	SCE	TLBO
7.97	9.97	7.77	6.80	6.03	10.23	8.90	5.87	11.73	9.60	8.40	7.50	9.73	7.47	7.43	10.60

with a p-value of 0.02, the Friedman test suggests that there are significant differences between the algorithms. ES1P1, CA, and BSA rank best out of all algorithms (Table 19.26). From Table 19.27, the post hoc analysis reveals no further insight about the differences between the MHs and, thereby, contradicts the result of the Friedman test.

The difficulty measure of test set predictions is also used to compare the MHs. The Friedman test shows the following results:

$$\text{Friedman's chi-squared} = 7.82, \text{ df} = 15, \ p\text{-value} = 0.93.$$

The null hypothesis that apart from the effect of the data set, there is no difference between the MHs cannot be rejected. The p-value exceeds the 5%-significance level by far. ES1P1, TLBO, and ACO have the best rankings of all MHs with respect to the difficulty measure (see Table 19.28). The test for differences in the interrater agreement of test set predictions results in:

$$\text{Friedman's chi-squared} = 6.59, \text{ df} = 15, \ p\text{-value} = 0.97.$$

Similar to the previous test, there are no differences between the MHs regarding the interrater agreement as the p-value equals 0.97. Here, the best ranked algorithms are SCE, BBO, and CA (see Table 19.29). When comparing the MHs for computation time, only the results for the Brier score are used since this measure proved to be most suitable in the previous comparison of the scores. The test for differences in computation time results in:

$$\text{Friedman chi-squared} = 194.91, \text{ df} = 15, \ p\text{-value} < 2.2\text{e}{-16}.$$

There clearly are differences in computation time as the p-value is below 2.2e−16. ES1P1, SA, and CA have the best average ranks among the 16 algorithms (see Table 19.30). The post hoc test in Table 19.32 shows where pairwise differences between single MHs can be found. Figure 19.4 reveals how different the algorithms are regarding the ranges as well as minimum and maximum computational time. Also in terms of weight vector sparsity there are differences between the MHs as the comparison of the percentage of zero-weights attests. Again, the p-value is very small:

Table 19.27 Finner-corrected p-values in Friedman post hoc test for differences of MH regarding H-measure

	ABC	ACO	BBO	BSA	CA	CMAES	DE	ES1P1	FA	GA	HS	ICA	PSO	SA	SCE	TLBO
ABC		0.47	0.92	0.64	0.48	0.45	0.70	0.46	0.21	0.55	0.84	0.83	0.52	0.82	0.81	0.38
ACO	0.47		0.45	0.32	0.21	0.90	0.67	0.21	0.52	0.86	0.55	0.41	0.91	0.41	0.41	0.79
BBO	0.92	0.45		0.70	0.52	0.41	0.65	0.49	0.21	0.50	0.79	0.90	0.47	0.88	0.87	0.36
BSA	0.64	0.32	0.70		0.75	0.26	0.46	0.70	0.17	0.36	0.55	0.77	0.35	0.78	0.79	0.21
CA	0.48	0.21	0.52	0.75		0.21	0.36	0.93	0.08	0.23	0.43	0.57	0.21	0.57	0.58	0.19
CMAES	0.45	0.90	0.41	0.26	0.21		0.59	0.21	0.56	0.79	0.50	0.36	0.82	0.36	0.36	0.86
DE	0.70	0.67	0.65	0.46	0.36	0.59		0.32	0.36	0.77	0.82	0.58	0.72	0.57	0.57	0.53
ES1P1	0.46	0.21	0.49	0.70	0.93	0.21	0.32		0.08	0.21	0.41	0.55	0.21	0.55	0.55	0.18
FA	0.21	0.52	0.21	0.17	0.08	0.56	0.36	0.08		0.46	0.28	0.21	0.47	0.21	0.21	0.65
GA	0.55	0.86	0.50	0.36	0.23	0.79	0.77	0.21	0.46		0.64	0.46	0.94	0.46	0.45	0.69
HS	0.84	0.55	0.79	0.55	0.43	0.50	0.82	0.41	0.28	0.64		0.71	0.59	0.70	0.70	0.45
ICA	0.83	0.41	0.90	0.77	0.57	0.36	0.58	0.55	0.21	0.46	0.71		0.45	0.99	0.97	0.32
PSO	0.52	0.91	0.47	0.35	0.21	0.82	0.72	0.21	0.47	0.94	0.59	0.45		0.45	0.44	0.72
SA	0.82	0.41	0.88	0.78	0.57	0.36	0.57	0.55	0.21	0.46	0.70	0.99	0.45		0.99	0.32
SCE	0.81	0.41	0.87	0.79	0.58	0.36	0.57	0.55	0.21	0.45	0.70	0.97	0.44	0.99		0.32
TLBO	0.38	0.79	0.36	0.21	0.19	0.86	0.53	0.18	0.65	0.69	0.45	0.32	0.72	0.32	0.32	

Table 19.28 Average rank of the MHs from high (1) to low (16) difficulty measure

ABC	ACO	BBO	BSA	CA	CMAES	DE	ES1P1	FA	GA	HS	ICA	PSO	SA	SCE	TLBO
6.93	9.20	8.40	8.60	7.33	8.93	9.00	10.33	8.33	8.07	7.67	8.07	9.13	8.40	7.80	9.80

Table 19.29 Average rank of the MHs from high (1) to low (16) interrater agreement

ABC	ACO	BBO	BSA	CA	CMAES	DE	ES1P1	FA	GA	HS	ICA	PSO	SA	SCE	TLBO
8.13	7.27	9.60	7.20	8.80	8.07	8.93	8.13	8.67	9.33	9.47	8.67	8.07	7.67	9.93	8.07

Table 19.30 Average rank of MHs in time-wise comparison over data sets from slow (1) to fast (16)

ABC	ACO	BBO	BSA	CA	CMAES	DE	ES1P1	FA	GA	HS	ICA	PSO	SA	SCE	TLBO
3.20	5.53	7.80	5.13	12.73	11.33	9.47	15.53	1.07	7.27	12.53	5.33	7.13	15.47	4.00	12.47

Table 19.31 Average rank of MHs in sparsity comparison from high (1) to low (16) percentage of zero weights

ABC	ACO	BBO	BSA	CA	CMAES	DE	ES1P1	FA	GA	HS	ICA	PSO	SA	SCE	TLBO
9.80	4.97	8.67	12.57	15.07	8.03	3.47	7.80	1.23	13.00	13.83	5.70	3.47	12.87	13.13	2.40

$$\text{Friedman chi-squared} = 206.97,\ \text{df} = 15,\ p\text{-value} < 2.2\text{e}{-16}.$$

FA, TLBO, DE, and PSO are the highest ranked algorithms in this context (see Tables 19.31 and 19.32). In Table 19.33, one can inspect where significant differences between the algorithms are located. Lastly, from Fig. 19.5 it becomes obvious that BSA, CA, GA, HS, SA, and SCE create rather big ensembles. On the other hand, DE, FA, PSO, and TLBO tend to build ensembles with fewest models.

19.4.3.3 Comparison Between Metaheuristic Ensembles and Other Techniques

From the preceding comparison, four MHs were selected that ranked highest in terms of accuracy, diversity, computation time, and weight vector sparsity. Also, the algorithm complexity, i.e., the number of adjustable parameters, was taken into account such that less complex algorithms are preferred. The selected algorithms are SA, ES1P1, BSA, and CA which are compared to the best base model, the simple average, the ensemble selection with bagging (20 times) and without bagging (greedy optimization) from [7], and a LASSO regression model. The Brier score of the respective test set predictions was scaled to the best solution which was chosen as the Brier score of a LASSO regression model fitted to the test set. The comparative overview is presented in Table 19.34. For seven out of the fifteen data sets, the metaheuristic ensembles achieve better accuracy than the other algorithms. CA, SA, and BSA show the highest average rank among

Table 19.32 Finner-corrected p-values in Friedman post hoc test for differences of MHs regarding computation time

	ABC	ACO	BBO	BSA	CA	CMAES	DE	ES1P1	FA	GA	HS	ICA	PSO	SA	SCE	TLBO
ABC		0.23	0.02	0.32	0.00	0.00	0.00	0.00	0.27	0.03	0.00	0.27	0.04	0.00	0.68	0.00
ACO	0.23		0.25	0.84	0.00	0.00	0.04	0.00	0.02	0.37	0.00	0.92	0.40	0.00	0.42	0.00
BBO	0.02	0.25		0.17	0.01	0.07	0.38	0.00	0.00	0.78	0.01	0.21	0.73	0.00	0.05	0.01
BSA	0.32	0.84	0.17		0.00	0.00	0.02	0.00	0.03	0.27	0.00	0.92	0.30	0.00	0.55	0.00
CA	0.00	0.00	0.01	0.00		0.46	0.09	0.15	0.00	0.00	0.92	0.00	0.00	0.16	0.00	0.89
CMAES	0.00	0.00	0.07	0.00	0.46		0.33	0.03	0.00	0.03	0.53	0.00	0.03	0.03	0.00	0.55
DE	0.00	0.04	0.38	0.02	0.09	0.33		0.00	0.00	0.26	0.12	0.03	0.23	0.00	0.00	0.12
ES1P1	0.00	0.00	0.00	0.00	0.15	0.03	0.00		0.00	0.00	0.12	0.00	0.00	0.97	0.00	0.12
FA	0.27	0.02	0.00	0.03	0.00	0.00	0.00	0.00		0.00	0.00	0.03	0.00	0.00	0.13	0.00
GA	0.03	0.37	0.78	0.27	0.00	0.03	0.26	0.00	0.00		0.01	0.32	0.94	0.00	0.09	0.01
HS	0.00	0.00	0.01	0.00	0.92	0.53	0.12	0.12	0.00	0.01		0.00	0.00	0.13	0.00	0.97
ICA	0.27	0.92	0.21	0.92	0.00	0.00	0.03	0.00	0.03	0.32	0.00		0.35	0.00	0.48	0.00
PSO	0.04	0.40	0.73	0.30	0.00	0.03	0.23	0.00	0.00	0.94	0.00	0.35		0.00	0.11	0.00
SA	0.00	0.00	0.00	0.00	0.16	0.03	0.00	0.97	0.00	0.00	0.13	0.00	0.00		0.00	0.12
SCE	0.68	0.42	0.05	0.55	0.00	0.00	0.00	0.00	0.13	0.09	0.00	0.48	0.11	0.00		0.00
TLBO	0.00	0.00	0.01	0.00	0.89	0.55	0.12	0.12	0.00	0.01	0.97	0.00	0.00	0.12	0.00	

Table 19.33 Finner-corrected p-values in Friedman post hoc test for differences of MHs regarding sparsity

	ABC	ACO	BBO	BSA	CA	CMAES	DE	ES1P1	FA	GA	HS	ICA	PSO	SA	SCE	TLBO
ABC		0.01	0.57	0.15	0.01	0.36	0.00	0.30	0.00	0.10	0.03	0.03	0.00	0.11	0.08	0.00
ACO	0.01		0.05	0.00	0.00	0.11	0.45	0.14	0.05	0.00	0.00	0.70	0.45	0.00	0.00	0.19
BBO	0.57	0.05		0.04	0.00	0.74	0.01	0.66	0.00	0.02	0.01	0.12	0.01	0.03	0.02	0.00
BSA	0.15	0.00	0.04		0.20	0.02	0.00	0.01	0.00	0.82	0.53	0.00	0.00	0.87	0.77	0.00
CA	0.01	0.00	0.00	0.20		0.00	0.00	0.00	0.00	0.29	0.53	0.00	0.00	0.26	0.32	0.00
CMAES	0.36	0.11	0.74	0.02	0.00		0.02	0.90	0.00	0.01	0.00	0.24	0.02	0.01	0.01	0.00
DE	0.00	0.45	0.01	0.00	0.00	0.02		0.02	0.26	0.00	0.00	0.26	1.00	0.00	0.00	0.59
ES1P1	0.30	0.14	0.66	0.01	0.00	0.90	0.02		0.00	0.01	0.00	0.28	0.02	0.01	0.00	0.00
FA	0.00	0.05	0.00	0.00	0.00	0.00	0.26	0.00		0.00	0.00	0.02	0.26	0.00	0.00	0.56
GA	0.10	0.00	0.02	0.82	0.29	0.01	0.00	0.01	0.00		0.67	0.00	0.00	0.94	0.94	0.00
HS	0.03	0.00	0.01	0.53	0.53	0.00	0.00	0.00	0.00	0.67		0.00	0.00	0.62	0.72	0.00
ICA	0.03	0.70	0.12	0.00	0.00	0.24	0.26	0.28	0.02	0.00	0.00		0.26	0.00	0.00	0.09
PSO	0.00	0.45	0.01	0.00	0.00	0.02	1.00	0.02	0.26	0.00	0.00	0.26		0.00	0.00	0.59
SA	0.11	0.00	0.03	0.87	0.26	0.01	0.00	0.01	0.00	0.94	0.62	0.00	0.00		0.89	0.00
SCE	0.08	0.00	0.02	0.77	0.32	0.01	0.00	0.00	0.00	0.94	0.72	0.00	0.00	0.89		0.00
TLBO	0.00	0.19	0.00	0.00	0.00	0.00	0.59	0.00	0.56	0.00	0.00	0.09	0.59	0.00	0.00	

Table 19.34 Performance comparison of four best base models to other techniques

Data set	SA	ES1P1	BSA	CA	Best base model	Simple average	Greedy optimization	Ensemble selection	LASSO regression
Adult	0.9375	0.9391	**0.9396**	0.9373	0.9028	0.9336	0.9359	0.9358	0.8965
Bank	0.8159	0.8106	0.7603	0.7990	0.5921	**0.8260**	0.6907	0.6991	0.4279
Chess	0.9874	0.9890	0.9940	0.9905	0.9973	0.9857	**0.9979**	**0.9979**	0.9958
Credit	0.8162	0.8187	0.8429	0.8295	**0.8804**	0.8081	0.8533	0.8544	0.5479
Eye	0.8904	0.8942	0.8388	0.8846	0.7362	**0.9154**	0.8802	0.8750	0.8518
Fertility	**0.9067**	0.9034	0.9036	0.9048	0.8160	0.9058	0.8642	0.8740	0.5587
Heart	0.6192	0.5849	0.6302	**0.6303**	0.4069	0.6096	0.5222	0.5115	0.5134
Magic	0.7731	0.7748	0.7855	0.7764	0.6472	0.7782	0.7977	**0.8062**	0.6255
Ionosphere	0.7930	0.7887	0.7943	0.7912	0.7551	0.7890	0.7831	0.7862	**0.8286**
Pima	**0.5653**	0.5613	0.5428	0.5597	0.3566	0.5650	0.4836	0.4987	0.4968
Qsar	0.7707	0.7710	0.7662	**0.7730**	0.7628	0.7695	0.7507	0.7506	0.2869
Seismic	0.9715	0.9698	0.9736	0.9720	**0.9793**	0.9696	0.9774	0.9774	0.9741
Spambase	0.7972	0.8019	**0.8141**	0.8046	0.6522	0.7945	0.7748	0.7744	0.7135
Shuttle	**0.6389**	0.6124	0.5998	0.6241	0.2154	0.5999	0.2439	0.2276	0.2191
Wilt	0.9724	0.9723	0.9723	0.9725	**0.9821**	0.9725	0.9719	0.9735	0.9719
Mean rank	3.8700	4.5300	4.0700	3.6000	6.5300	4.6000	5.4700	5.0000	**7.3300**

Brier scores of test set predictions are scaled to the best solution defined by a LASSO regression model estimated on the test set; largest values for each data set are indicated in bold

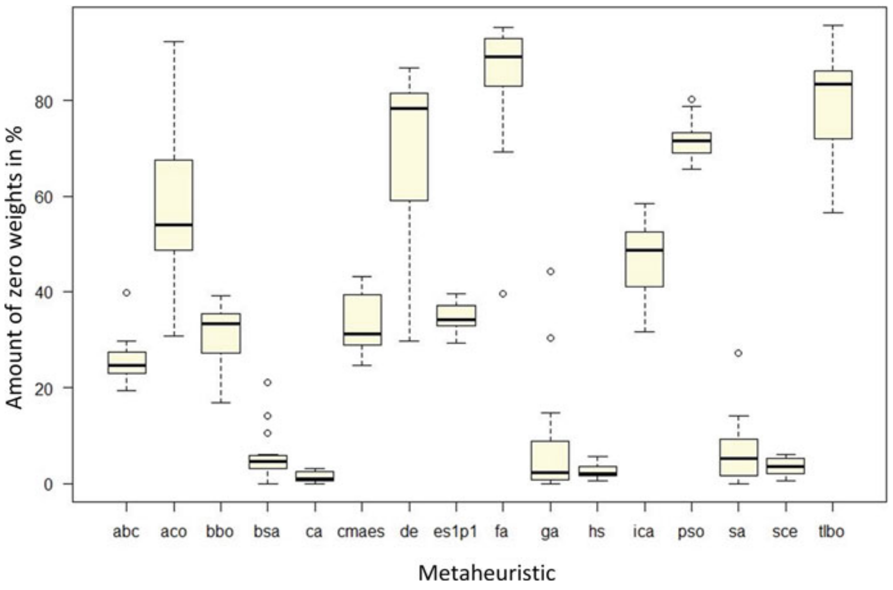


Fig. 19.5 Boxplots of percentage of zero weights of ensembles for each MH

the nine methods. Especially where there is overfitting in the best base model such as in shuttle, pima, or heart, techniques based on averaging show better results than techniques based on base model accuracy. However, for ensemble selection and LASSO regression, the optimization on test set predictions is recommended which explains their bad performance. Optimization on test set predictions could not be realized since there would have been no hold-out data sets available and the comparison to the metaheuristic ensembles fitted to the training sets would not have been meaningful. In case of training set optimization, ensembles based on averaging are clearly superior to ensemble selection and LASSO regression as they are less affected by overfitting of the base models. Nevertheless, it could also be advantageous to optimize metaheuristic ensembles on test set predictions to eliminate the effect of base model overfitting on the weight configuration.

19.5 Conclusions

In this paper, 16 metaheuristics were analysed and compared regarding their performance in determining weights of weighted average ensembles. In the beginning, the basic theory of these kinds of ensembles was presented as well as the metaheuristics with their respective principles of operation. This was followed by an experimental study in which, for 15 different data sets, libraries of base models were created. From these libraries, weighted average ensembles were defined by weights that were optimized by metaheuristics with respect to score measures of accuracy and diversity. These score measures were examined for differences among predictive accuracy, computation time, and percentage of zero weights of the underlying ensembles. Subsequently, the 16 metaheuristics were compared by each score as well as by computation time and percentage of zero weights of Brier score ensembles. Conclusively, the four metaheuristics that ranked highest among all comparisons were compared by predictive accuracy to simple average, best base model, ensemble selection, and LASSO regression.

In the comparison of the four deployed score measures, significant differences were detected in the accuracy and weight vector sparsity of ensembles. However, no significant difference was found in computation time. Considering the ranks, the Brier score was best in the accuracy comparison and the sparsity comparison. With view on computation time, the Brier score ranged to shortest required times, while the H-measure showed longest duration. In the comparison between the 16 metaheuristics, no significant differences were determined regarding ensemble fitness. The metaheuristics significantly differed only in computation time and sparsity. Regarding the ranks, SA, ES1P1, and CA required least computation time, while FA, ABC, and SCE required most. FA, TLBO, PSO, and DE yielded sparsest weight vectors, while CA, GA, and SCE performed worst in producing sparsity. In the final comparison between the metaheuristic ensembles, best base model, and other ensemble techniques (in Table 19.34), selected SA, ES1P1, CA, and BSA collectively outperform other techniques for seven out of the fifteen data sets with respect to the Brier score. Out of these seven instances, three were achieved by SA. Certainly, in case of overfitting of the best base model, the averaging techniques outperformed techniques based on base model accuracy. If no overfitting occurs in the best base model, the superiority of the metaheuristic ensembles is not evident.

Since no clear differences could be detected between the metaheuristics concerning accuracy and diversity, no metaheuristic is clearly preferable over another in that sense. The time-wise comparison (in Fig. 19.2) demonstrated that the metaheuristics differ by the speed at which they meet the termination criteria. The single-solution methods SA and ES1P1 seem to get stuck in local optima quicker and are less likely to escape from them which results in early abort and low computation time. Methods like FA demand long duration which implies that they escape easier from local optima and constantly change the best fitness function value. However, considering the ranks in the accuracy (see Table 19.25) and time comparison (see Fig. 19.2), a longer run does not necessarily result in higher predictive accuracy of the ensemble. The differences of the metaheuristics in producing weight vector sparsity (see Table 19.31) can be utilized in cases of low storage capacity. One might use a metaheuristic that combines only a few base models and allows for discarding unused models. The comparison of the score measures (see Table 19.20) clearly indicates to prefer the Brier score over the H-measure when the aim is an accurate ensemble. Also, the Brier score produces ensembles in shorter time (see Fig. 19.2) and leads to fewer base models than other measures and is, therefore, preferable.

The novelty of the MHs considered has already been challenged in previous work. In view of the observed results, one may draw on corresponding criticism and target it to the focal optimization task, asking, for example, whether the choice of the search mechanism in an ensemble selection framework really matters. In view of the observed results, it seems legitimate to formulate some concluding comments as follows: First, the ensemble selection literature appears to be largely unaware of the rich set of MHs available today. The choice of a heuristic or MH is rarely investigated or reflected. GA or PSO are usually the methods of choice, whereas other approaches are used, at best, rarely. A merit of our investigation is that it raises awareness among machine learners that the realm of MHs is probably as large as the space of learning methods and that, when facing an optimization problem such as ensemble selection, it might be sensible to spend time on thinking which MH might be a suitable approach. In particular, we have shown that differences across MHs in terms of accuracy, computing times, diversity, and sparsity exist and may be sizeable. In this regard, our study makes a first step towards a better understanding of MH features for ensemble selection, and eventually even other optimization tasks in machine learning. A related, second point is that novel approaches for ensemble selection—and maybe machine learning—often come as general purpose procedures. This paper is no exception in that we also assess our models on a set of data sets from different domains, implicitly claiming that they are able to solve problems in all these different domains. The development or advancement of a MH is more often pursued in the context of a given problem setting like routing, scheduling, etc. In demonstrating differences in the ability of a MH to produce accurate, or sparse, or fast, or diverse ensembles, our results call for a more problem-oriented approach to develop learning methods. Third, although the magnitudes of performance differences among MHs might not have been substantial across our data sets, we highlight that the approaches that machine learners routinely rely on may provide average performance and thus represent a questionable choice. This will come as no surprise to the operations research community, which has invested tremendous effort in further advancing the principles of heuristic search over decades and contributed a vast set of powerful algorithms. A somewhat critical remark to that community might be that we have a lot to learn when it comes to "selling" our methods. Deep learning and artificial intelligence are omnipresent. Reading about the latest MHs in the daily news is comparably less common.

Future research could aim at different aspects in metaheuristic ensembling. For instance, in the proposed work, to simplify matters, there were no efforts expended on tuning of metaheuristics towards higher accuracy. In future work an autocatalytic parameterization would be advisable; though, while this seems largely missing in the largest portion of the metaheuristics discussion anyway, this is more like a general research avenue to pursue. The metaheuristics were solely adjusted to produce results in a foreseeable amount of time. Hence, with respect to accuracy, the metaheuristics are assumed to be equally well tuned for solving the continuous optimization problem. That is, this could be the purpose of future research to support the validity of the results encountered here. Furthermore, the termination criteria could be reconsidered. The given criteria are geared to shorten computation times and could possibly leave too little time for algorithms to escape from local optima. In future research, computational effort could be measured and compared by the number of fitness function evaluations in a given maximum amount of time as recommended in [22] in order to achieve higher quality in time comparisons. Another detail which has not met consideration in this work is the probabilistic nature of metaheuristic procedures and the implicit randomness of their search results. This aspect could be approached by building ensembles in several runs for one metaheuristic-score combination and subsequent averaging of resulting samples. Lastly, the combinatorial problem of generating ensemble diversity was provisionally solved by metaheuristics built for continuous optimization problems. Existing metaheuristics for combinatorial optimization such as tabu search could easily be adapted to optimize diversity measures and thus could yield higher predictive performance.

A detailed study of the correlation between accuracy, computation time, and size of metaheuristic ensembles could directly follow the approach presented in this paper. A more valid comparison of metaheuristic ensembles and other techniques would have to be based on weights optimized on hold-out data predictions for each single procedure. One might also reconsider the application of the weighted average combination of classifiers and incorporate another ensembling method such as majority voting into a comparison of metaheuristics, instead. For any new strategy, the controlling of metaheuristic parameters as well as termination criteria should be accomplished at the very beginning.

References

1. Ali MZ, Awad NH, Suganthan PN, Duwairi RM, Reynolds RG (2016) A novel hybrid cultural algorithms framework with trajectory-based search for global numerical optimization. Information Sciences 334:219–249, https://doi.org/10.1016/j.ins.2015.11.032
2. Asuncion A, Newman D (2007) UCI machine learning repository. http://www.ics.uci.edu/~mlearn/MLRepository.html
3. Atashpaz-Gargari E, Lucas C (2007) Imperialist competitive algorithm: An algorithm for optimization inspired by imperialistic competition. In: IEEE Congress on Evolutionary Computation, IEEE, pp 4661–4667, http://dblp.uni-trier.de/db/conf/cec/cec2007.html#Atashpaz-GargariL07
4. Beyer HG, Schwefel HP (2002) Evolution strategies – a comprehensive introduction. Natural Computing 1(1):3–52, https://doi.org/10.1023/A:1015059928466
5. Blum C, Li X (2008) Swarm intelligence in optimization. In: Blum C, Merkle D (eds) Swarm Intelligence: Introduction and Applications, Springer, Berlin, Heidelberg, pp 43–85, https://doi.org/10.1007/978-3-540-74089-6_2
6. Blum C, Roli A (2003) Metaheuristics in combinatorial optimization: Overview and conceptual comparison. ACM Computing Surveys 35(3):268–308, http://doi.acm.org/10.1145/937503.937505
7. Caruana R, Niculescu-Mizil A, Crew G, Ksikes A (2004) Ensemble selection from libraries of models. In: Proceedings of the Twenty-first International Conference on Machine Learning (ICML), ACM, New York, pp 18, http://www.cs.cornell.edu/~caruana/ctp/ct.papers/caruana.icml04.icdm06long.pdf
8. Caserta M, Voß S (2010) Metaheuristics: Intelligent problem solving. In: Maniezzo V, Stützle T, Voß S (eds) Matheuristics: Hybridizing Metaheuristics and Mathematical Programming, Springer US, pp 1–38

9. Chen Y, Wong ML (2010) An ant colony optimization approach for stacking ensemble. In: Nature and Biologically Inspired Computing (NaBIC), 2010 Second World Congress on, pp 146–151
10. Chipperfield A, Fleming P, Pohlheim H, Fonseca C (1994) Genetic algorithm toolbox for use with matlab. Tech. rep., Department of Automatic Control and Systems Engineering, University of Sheffield
11. Civicioglu P (2013) Backtracking search optimization algorithm. https://de.mathworks.com/matlabcentral/fileexchange/44842-backtracking-search-optimization-algorithm, accessed: 2016-04-01
12. Civicioglu P (2013) Backtracking search optimization algorithm for numerical optimization problems. Applied Mathematics and Computation 219(15):8121–8144, https://doi.org/10.1016/j.amc.2013.02.017
13. Coletta LFS, Hruschka ER, Acharya A, Ghosh J (2013) Towards the use of metaheuristics for optimizing the combination of classifier and cluster ensembles. In: 2013 BRICS Congress on Computational Intelligence and 11th Brazilian Congress on Computational Intelligence, pp 483–488
14. Dietterich TG (2000) Ensemble methods in machine learning. In: Multiple Classifier Systems, Springer, Lecture Notes in Computer Science, vol 1857, pp 1–15, http://web.engr.oregonstate.edu/~tgd/publications/mcs-ensembles.pdf
15. Duan QY, Gupta VK, Sorooshian S (1993) Shuffled complex evolution approach for effective and efficient global minimization. Journal of Optimization Theory and Applications 76(3):501–521, https://doi.org/10.1007/BF00939380
16. Eberhart R, Kennedy J (1995) A new optimizer using particle swarm theory. In: Sixth International Symposium on Micro Machine and Human Science, IEEE, pp 39–43
17. Ekbal A, Saha S (2011) A multiobjective simulated annealing approach for classifier ensemble: Named entity recognition in Indian languages as case studies. Expert Systems with Applications 38(12):14760–14772, https://doi.org/10.1016/j.eswa.2011.05.004
18. Fawcett T (2006) An introduction to roc analysis. Pattern Recognition Letters 27(8):861–874, http://www.sciencedirect.com/science/article/B6V15-4HV747X-1/2/c1653cca4db4e94215437a482fcbecbb
19. Gabrys B, Ruta D (2006) Genetic algorithms in classifier fusion. Applied Soft Computing 6(4):337–347, https://doi.org/10.1016/j.asoc.2005.11.001
20. García S, Fernández A, Luengo J, Herrera F (2010) Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental analysis of power. Information Science 180(10):2044–2064, https://doi.org/10.1016/j.ins.2009.12.010
21. Geem Z, Kim J, Loganathan G (2001) A new heuristic optimization algorithm: Harmony search. Simulation 76(2):60–68
22. Gendreau M, Potvin JY (2010) Handbook of Metaheuristics, 2nd edn. Springer
23. Glover F (1994) Genetic algorithms and scatter search: Unsuspected potentials. Statistics and Computing 4:131–140
24. Glover F (2000) Fundamentals of scatter search and path relinking. Control and Cybernetics 29(3):653–684
25. Glover F, Kochenberger GA (2003) Handbook of Metaheuristics. Kluwer, Boston, http://opac.inria.fr/record=b1099522
26. Greistorfer P, Voß S (2005) Controlled pool maintenance for meta-heuristics. In: Rego C, Alidaee B (eds) Metaheuristic Optimization via Memory and Evolution, Kluwer, Boston, pp 387–424
27. Hand DJ (2009) Measuring classifier performance: A coherent alternative to the area under the roc curve. Machine Learning 77(1):103–123, https://doi.org/10.1007/s10994-009-5119-5
28. Hansen N (2006) The CMA evolution strategy: a comparing review. In: Lozano J, Larranaga P, Inza I, Bengoetxea E (eds) Towards a new evolutionary computation. Advances on estimation of distribution algorithms, Springer, pp 75–102

29. Hastie T, Tibshirani R, Friedman JH (2009) The Elements of Statistical Learning, 2nd edn. Springer, New York
30. Hernández-Orallo J, Flach PA, Ramirez CF (2011) Brier curves: a new cost-based visualisation of classifier performance. In: Getoor L, Scheffer T (eds) Proceedings of the 28th International Conference on Machine Learning, Omnipress, pp 585–592, http://dblp.uni-trier.de/db/conf/icml/icml2011.html#Hernandez-OralloFR11
31. Hosseini S, Khaled AA (2014) A survey on the imperialist competitive algorithm metaheuristic: Implementation in engineering domain and directions for future research. Applied Soft Computing 24:1078–1094, https://doi.org/10.1016/j.asoc.2014.08.024
32. Ingber L (1996) Adaptive simulated annealing (ASA): Lessons learned. Control and Cybernetics 25:33–54
33. Janikow CZ, Michalewicz Z (1991) An experimental comparison of binary and floating point representations in genetic algorithms. In: Belew RK, Booker LB (eds) Proceedings of the 4th International Conference on Genetic Algorithms, Morgan Kaufmann, pp 151–157–36, http://dblp.uni-trier.de/db/conf/icga/icga1991.html#JanikowM91
34. Jing Y, Xiaoqin Z, Shuiming Z, Shengli W (2013) Effective neural network ensemble approach for improving generalization performance. IEEE Transactions on Neural Networks and Learning Systems 24(6):878–887
35. Karaboga D, Basturk B (2007) Artificial bee colony (ABC) optimization algorithm for solving constrained optimization problems. In: Melin P, Castillo O, Aguilar LT, Kacprzyk J, Pedrycz W (eds) Foundations of Fuzzy Logic and Soft Computing: Proceedings of the 12th International Fuzzy Systems Association World Congress, IFSA 2007, Cancun, Mexico, June 18-21, 2007, Springer, Berlin, Heidelberg, pp 789–798, https://doi.org/10.1007/978-3-540-72950-1_77
36. Kirkpatrick S, Gelatt CD, Vecchi MP (1983) Optimization by simulated annealing. Science 220(4598):671–680
37. Lessmann S, Baesens B, Seow HV, Thomas LC (2015) Benchmarking state-of-the-art classification algorithms for credit scoring: An update of research. European Journal of Operational Research 247(1):124–136, http://www.sciencedirect.com/science/article/pii/S0377221715004208, https://doi.org/10.1016/j.ejor.2015.05.030
38. MATLAB (2010) version 7.10.0 (R2010a). The MathWorks Inc., Natick, Massachusetts
39. Mitchell M (1995) Genetic algorithms: An overview. Complexity 1(1):31–39, https://doi.org/10.1002/cplx.6130010108
40. Nabavi-Kerizi SH, Abadi M, Kabir E (2010) A PSO-based weighting method for linear combination of neural networks. Comput Electr Eng 36(5):886–894, https://doi.org/10.1016/j.compeleceng.2008.04.006
41. Ortiz GA (2012) (1+1)-evolutionary strategy. https://de.mathworks.com/matlabcentral/fileexchange/35800-1+1-evolution-strategy--es-, accessed: 2016-04-01
42. Palanisamy S, Kanmani S (2012) Classifier ensemble design using artificial bee colony based feature selection. International Journal of Computer Science Issues 9(2):522–529
43. Partalas I, Tsoumakas G, Vlahavas I (2010) An ensemble uncertainty aware measure for directed hill climbing ensemble pruning. Machine Learning 81(3):257–282, https://doi.org/10.1007/s10994-010-5172-0
44. Poli R, Kennedy J, Blackwell T (2007) Particle swarm optimization. Swarm Intelligence 1(1):33–57, https://doi.org/10.1007/s11721-007-0002-0
45. Quoos M, Pozniak-Koszalka I, Koszalka L, Kasprzak A (2015) Multiple classifier system with metaheuristic algorithms. In: Gervasi O, Murgante B, Misra S, Gavrilova LM, Rocha CAMA, Torre C, Taniar D, Apduhan OB (eds) Computational Science and Its Applications – ICCSA 2015: Proceedings of the 15th International Conference, Banff, AB, Canada, June 22-25, 2015, Part II, Springer, Cham, pp 43–54, https://doi.org/10.1007/978-3-319-21407-8_4
46. Rao RV, Savsani VJ, Vakharia DP (2011) Teaching-learning-based optimization: A novel method for constrained mechanical design optimization problems. Computer Aided Design 43(3):303–315, https://doi.org/10.1016/j.cad.2010.12.015

47. Rechenberg I (1970) Optimierung technischer Systeme nach Prinzipien der biologischen Evolution. Dissertation, Technische Universität Berlin
48. Resende M, Ribeiro C, Glover F, Marti R (2010) Scatter search and path relinking: Fundamentals, advances and applications. In: Handbook of Metaheuristics, Springer, New York, pp 87–107
49. Reynolds RG (1994) An introduction to cultural algorithms. In: Sebald AV, Fogel LJ (eds) Evolutionary Programming — Proceedings of the Third Annual Conference, World Scientific Press, San Diego, CA, USA, pp 131–139, http://ai.cs.wayne.edu/ai/availablePapersOnLine/IntroToCA.pdf
50. Schwefel HP (1975) Evolutionsstrategie und numerische Optimierung. Dissertation, Technische Universität Berlin
51. Segredo E, Lalla-Ruiz E, Hart E, BPaechter, Voß S (2016) Analysing the performance of migrating birds optimisation approaches for large scale continuous problems. Lecture Notes in Computer Science 9921:134–144
52. Shmueli G, Koppius OR (2011) Predictive analytics in information systems research. MIS Quarterly 35(3):553–572
53. Simon D (2008) Biogeography-based optimization. IEEE Transactions on Evolutionary Computation 12(6):702–713
54. Socha K, Dorigo M (2008) Ant colony optimization for continuous domains. European Journal of Operational Research 185(3):1155–1173, http://EconPapers.repec.org/RePEc:eee:ejores:v:185:y:2008:i:3:p:1155-1173
55. Sorensen K, Sevaux M, Glover F (2018) A history of metaheuristics. In: Martí R, Pardalos P, Resende M (eds) Handbook of Heuristics, Springer, Cham. https://doi.org/10.1007/978-3-319-07153-4_4-1
56. Storn R, Price K (1997) Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces. Journal of Global Optimization 11(4):341–359, https://doi.org/10.1023/A:1008202821328
57. Taghavi S, Sajedi H (2014) Ensemble selection using simulated annealing walking. International Journal of Advances in Computer Science & Its Applications 4(4):174–178
58. Tahir MA, Smith J (2010) Creating diverse nearest-neighbour ensembles using simultaneous metaheuristic feature selection. Pattern Recognition Letters 31(11):1470–1480. https://doi.org/10.1016/j.patrec.2010.01.030
59. Tang EK, Suganthan PN, Yao X (2006) An analysis of diversity measures. Machine Learning 65(1):247–271. https://doi.org/10.1007/s10994-006-9449-2
60. Tsoumakas G, Partalas I, Vlahavas I (2009) An ensemble pruning primer. In: Okun O, Valentini G (eds) Applications of Supervised and Unsupervised Ensemble Methods, Studies in Computational Intelligence, Springer, Berlin, pp 1–13, https://doi.org/10.1007/978-3-642-03999-7_1
61. Visentini I, Snidaro L, Foresti GL (2016) Diversity-aware classifier ensemble selection via f-score. Information Fusion 28:24–43, http://www.sciencedirect.com/science/article/pii/S1566253515000688
62. Weyland D (2010) A rigorous analysis of the harmony search algorithm: How the research community can be misled by a "novel" methodology. International Journal of Applied Metaheuristic Computing (IJAMC) 1(2):50–60
63. Yang XS (2009) Firefly algorithms for multimodal optimization. In: Watanabe O, Zeugmann T (eds) Stochastic Algorithms: Foundations and Applications: Proceedings of the 5th International Symposium, SAGA 2009, Sapporo, Japan, October 26-28, 2009., Springer, Berlin, Heidelberg, pp 169–178, https://doi.org/10.1007/978-3-642-04944-6_14
64. Yarpiz (2016) Yarpiz. http://yarpiz.com/category/metaheuristics, accessed: 2016-04-01
65. Yin PY, Glover F, Laguna M, Zhu JX (2010) Cyber swarm algorithms — improving particle swarm optimization using adaptive memory strategies. European Journal of Operational Research 201:377–389

66. Yu X, Gen M (2012) Introduction to Evolutionary Algorithms. Springer
67. Zhou ZH (2012) Ensemble Methods: Foundations and Algorithms. Chapman & Hall/CRC
68. Zhou ZH, Wu JX, Jiang Y, Chen SF (2001) Genetic algorithm based selective neural network ensemble. In: Proceedings of the 17th International Joint Conference on Artificial Intelligence - Volume 2, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, IJCAI'01, pp 797–802, http://dl.acm.org/citation.cfm?id=1642194.1642200

Chapter 20
A Multi-objective Meta-Analytic Method for Customer Churn Prediction

Mohammad Nazmul Haque, Natalie Jane de Vries, and Pablo Moscato

Abstract The term 'metaheuristic' was introduced in 1986 as a way to label 'a higher-level procedure designed to guide a lower-level heuristic or algorithm' to find solutions for tasks posed as mathematical optimization problems. Analogously, the term 'meta-analytics' can be used to refer to a higher-level procedure that guides ad hoc data analysis techniques. Heuristics that guide ensemble learning of heterogeneous classifier systems would be one of those procedures that can be referred to as 'meta-analytics'. In general, researchers use single-objective approaches for ensemble learning. In this contribution we investigate the use of a multi-objective evolutionary algorithm and we apply it to the problem of customer churn prediction. We compare the results with those of a symbolic regression-based approach. Each has its own merits. While the multi-objective approach excels at prediction, it lacks in interpretability for business insights. Oppositely, the symbolic regression-based approach has lower accuracy but can give business analysts some actionable tools. Depending on the nature of the business scenario, we recommend that both be employed together to maximize our understanding of consumer behaviour. High-quality individualized prediction based on multi-objective optimization can help a company to direct a message to a particular individual, while the results of a global symbolic regression-based approach may help large marketing campaigns or big changes in policies, cost structures and/or product offerings.

Keywords Churn · Customer churn prediction · Ensemble of classifiers · Ensemble learning · Genetic programming · Multi-objective ensemble · NSGA-II algorithm · Symbolic regression

M. N. Haque (✉) · N. J. de Vries · P. Moscato
School of Electrical Engineering and Computing, The University of Newcastle, Callaghan, NSW, Australia
e-mail: Mohammad.Haque@newcastle.edu.au; natalie.devries@newcastle.edu.au; Pablo.Moscato@newcastle.edu.au

P. Moscato, N. J. de Vries (eds.), *Business and Consumer Analytics: New Ideas*, https://doi.org/10.1007/978-3-030-06222-4_20

20.1 Introduction and Motivation

Since the early 1980s, machine learning research has been increasingly adopted to solve many problems in business and consumer analytics. The pace of the field is relentless with new breakthroughs being published almost on a daily basis. Needless to say, there is a wealth of information available on the Internet. Since the establishment of the World Wide Web in the early 1990s, knowledge boundaries have dissolved and people can now have access to software and source code from almost anywhere in the world just by browsing public domain repositories. The adoption of languages like Java, R, Python, etc. are also contributing to the development of large collections of machine learning and classification software which is 'top-of-the-shelf' and ready to use for data analytic objectives. The question for this particular field is then which classification algorithms would be most useful for a given problem when we have such an abundance of alternatives.

The path to answering this question is perhaps obvious. We need to conduct research in the area of meta-analytics, that is, methodologies which aim at guiding individual analytic methods to improve the final performance by exploiting synergies between the individual classifiers. A prime candidate for such research is the area of Ensemble Learning. Broadly, ensemble learning is the process by which multiple models, such as classifiers (as in our case) or experts, are strategically generated and combined to solve a particular computational intelligence problem.

We should be aware that the primary objective in data classification is to finally deliver a method that consistently predicts the right target when confronted with samples coming from a particular distribution. The system has been trained with an independent set of samples and thus good performance relative to these new samples is seen as a good indicator of its generalization capability. During the past decade, researchers have been increasingly accepting the notion that ensembles of classifiers can be a useful way to improve the generalization capability in problems related to prediction and regression.

In spite of the fact that theoretical results indicate that ensembles could outperform the generalization performance of their individual members (called 'base classifiers'), selecting such a set—a group that 'works together' to meet the desired theoretical criteria (combining all accurate and diverse base classifiers) given a particular classification problem—is not an easy task [11]. It may be, in a certain sense, what mathematicians describe as an 'ill posed' problem. However, it is clear that a meta-analytic procedure embodies the appropriate elements for selecting such a group. This provides a strong direction for future research to assure a consistent flow of new innovative designs for base classifiers. Beyond the first objective of classification/prediction, we are confronted with the objective of selecting a suitable set of classifiers. Analogous to the problem of over-fitting in machine learning, including too many base classifiers may achieve high-performance in a training set but yield poor generalization ability. This means that we would need to identify, for a given analytic request, a set of base classifiers that generalizes well and does not have too many members. This introduces another dimension that we will make clear in Sects. 20.6.3 and 20.8 that ends this chapter. For the purpose of interpretability in business and consumer insight it is also desirable that the classifiers are of a relatively small number. Each of the classifiers can be interpreted as a 'model' of our business and the variables in it can allow some action which will result in better decision-making and outcomes. Having too many models may obscure the knowledge that can be derived from the whole process.

From the discussion above, and from a meta-analytics perspective, we are dealing with a multi-objective scenario: that of trying to maximize the generalization capability of an ensemble of classifiers while at the same time reducing the number of members in its set. We are thus recognizing an emerging trend. There have been advances in single-objective approaches (mostly accuracy-based), but now some researchers are turning to multi-objective optimization for further improvements by generalizing the ensemble of classifiers [4, 11, 16, 18]. Some accuracy-based approaches may have some problems with classification problems which have classes that are very imbalanced, as, for instance, when we have to discriminate between two classes and one class only has 10% of the samples represented in the training set. This means that other types of objective guidance functions should be selected for multi-objective optimization (MOO). Our contribution presents a case study of this type of problem domain in which we show how a meta-analytic procedure can handle multiple conflicting criteria for the design of an ensemble of classifiers in a challenging business analytics problem.

20.2 Churn Prediction: An Important Issue for Customer Relationship Management

It is well-known among business and marketing managers that obtaining new customers is a more difficult feat than maintaining existing ones. It is also more costly and time-consuming for the business. Although customer relationship management (CRM) is not always easy either, hanging on to existing customers is usually more feasible than trying to attract new customers (who may not know about your business or product). This makes customer churn an interesting research area. A customer is a 'churner' when he/she leaves the company as a customer. This concept is more applicable to service companies when long-term and ongoing relationships between the company and the consumer are more common, as in the case of banks or phone or internet providers. Naturally, businesses would like to be able to predict these churners or, at the very least, differentiate them from the customers who remain (non-churners) so that certain insights can be made into why these customers are churning. This scenario provides a natural setting for applying multi-objective optimization. Specifically, we investigate the problem of forecasting churners and non-churners of a telecom services company, drawing on the Churn telecom service customer dataset, which is publicly available online.[1] Details of the dataset can be found in the dataset description in Chap. 26.

[1] churn.txt inside the compressed file at: http://dataminingconsultant.com/DKD2e_data_sets.zip.

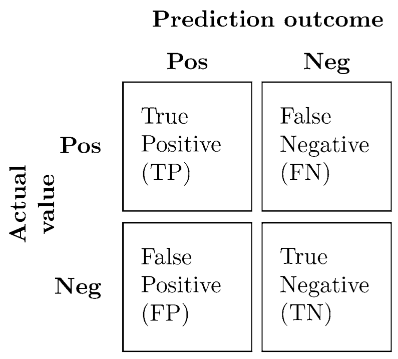


Fig. 20.1 A confusion matrix summarizes all possible outcomes in a binary-classification problem (TP, FP, FN and TN). In our case a TP means a churner predicted correctly to churn, a FP is a remaining customer that was predicted as a churner, a FN is a churner that was not predicted to churn and a TN is a remaining customer that was correctly predicted not to churn

Our objective is to predict 'churners' (those customers who left the company) and differentiate them from the 'non-churners' (those who remained loyal customers to the company). This dataset is a binary-class dataset containing 3333 samples and 20 features [24]. We split the dataset into training and testing sets using threefold cross validation (CV) of the original dataset (use 1 as the seed for randomly splitting with WEKA [13]). We used stratified-folds to keep the class distribution similar to the original dataset while separating them into training and testing folds. After doing the threefold CV, we get three pairs of training and testing fold datasets. The number of samples in each Train Folds is 2222 (1900, 322) and Test Folds is 1111 (950,161).

20.3 Classification Performance Measures

The importance of a classifier's prediction performance is critical. The prediction performance of a classifier is usually reported using a two row by two column matrix, widely known as a *confusion matrix* [29]. The four outcomes of a classification problem can be summarized in the confusion matrix as shown in Fig. 20.1.

Here, the outcomes are labelled as belonging to either the positive (Pos) or the negative (Neg) class. If the outcome from a prediction and the actual value is Pos, then it is called a true positive (TP); however, if the actual value is Neg, then it is said to be a false positive (FP). Conversely, when both predicted and actual outcomes are Neg, then it is denoted as a true negative (TN). It is denoted as a false negative (FN) when the prediction outcome is Neg, while the actual value is Pos.

$$Accuracy = \frac{(TP + TN)}{(TP + TN + FP + FN)} \tag{20.1}$$

$$Recall/Sensitivity = \frac{TP}{(TP + FN)} \tag{20.2}$$

$$Specificity = \frac{TN}{(TN + FP)} \tag{20.3}$$

$$Precision = \frac{TP}{(TP + FP)} \tag{20.4}$$

$$F\text{-}Measure = 2 \times \frac{(Precision \times Recall)}{(Precision + Recall)} \tag{20.5}$$

$$MCC = \frac{(TP \times TN) - (FP \times FN)}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}} \tag{20.6}$$

Different measures of classifier performance can be calculated from the confusion matrix, as shown in Eqs. (20.1)–(20.6). The classification accuracy ($Accu$) is one of the most significant metrics. Although classification accuracy is a measure used for comparing the prediction performance, it is not suitable for use with imbalanced data when the sample sizes of the two classes differ greatly. It only considers the proportion of correct classifications without considering the class distribution. This can result in the poor rating for the minority class being easily overwhelmed by the correct classification of the majority class. We note that identifying the minority classes can actually be an important outcome in many cases, such as fraud detection, cancer detection and intrusion detection. Thus, the more appropriate evaluation criteria are those that use all values from the confusion matrix for binary-classification problems. Other popular measures for comparing performance of different classifiers are based on *sensitivity* and *specificity*. Sensitivity (SEN) is the probability of predicting a positive outcome when the true state is positive, whereas specificity ($SPEC$) is the probability of predicting a negative outcome when the true state of a case is negative. While there is no perfect way of describing the confusion matrix of TP, FP, FN and TN by a single number, the Matthews Correlation Coefficient (MCC in Eq. (20.6)) quantifies the strength of the classifications using the confusion matrix. The MCC takes into account both the sensitivity and specificity, and can often provide a more balanced accuracy assessment of the model [10]. An MCC with a higher value indicates better predictions. It is commonly used as a performance measure for imbalanced datasets [15] in different fields of machine learning, including bioinformatics [14]. Notably, the MCC was chosen as a measure classifier performance in the initiative FDA MAQC-II led by the USA, which was based on microarray gene expression and genotyping data [27]. Further, for the analysis of the confusion matrix, the experimental results obtained by Jurman et al. [17] indicate that MCC is an ideal measure for practical classification. Thus, we have selected MCC as our measure of classification performance.

20.4 A Brief Introduction to Multi-objective Optimization in Ensemble Learning

The core principle of ensemble learning is to weigh several individual pattern classifiers, and then combine them to gain better accuracy than can be obtained by using each of them separately. In the machine learning paradigm, ensemble data mining methods strategically advance the power of committee methods, or combine models to achieve better prediction accuracy than any of the individual models could achieve. The basic goal when designing an ensemble is to use independent models so that the combined model will produce a better performance. Ensemble methods have been used by researchers from various disciplines such as pattern recognition, statistics and machine learning.

In addition to better prediction, in real-world scenarios we are often required to give solutions that optimize more than one objective function. Analogous to practising the art of 'feature engineering', applied mathematicians and computer scientist have to select objective functions that may best address the actual task they are required to perform. For some problems these functions are more natural, but in other cases the practitioners may need to 'engineer' which function or combination of those best suits the problem domain.

In some cases, the more useful objective functions are quite explicit but also rather conflicting. For example, we may be required to maximize a certain measure of performance while at the same time minimizing the cost to achieve this performance. At the same time we may need to maximize reliability while minimizing risk and execution time. In those cases, we typically cannot find a single final solution that is optimal for all objectives.

20.4.1 Multi-objective Optimization in a Nutshell

A multi-objective optimization problem (MOP) can be defined as the problem of finding optimal feasible solutions that optimize more than one conflicting objectives:

$$(\textbf{Minimize}) \left[f_1(\mathbf{x}), f_2(\mathbf{x}), \cdots, f_p(\mathbf{x})\right] \qquad \mathbf{x} \in \mathbf{S}, \tag{20.7}$$

where f_i, with $i = 1, 2, \cdots, p$ is the i-th objective function and $\mathbf{x}$ is a solution from the set of feasible solutions $\mathbf{S}$. In multi-objective optimization we are interested in more than one feasible solution for the optimization problem. The feasible solutions we seek are called *Pareto-optimal* solutions.

A solution $\mathbf{x}^* \in \mathbf{S}$ is called *Pareto-optimal* if there does not exist another solution that dominates it by virtue of having a smaller or equal objective value. A solution

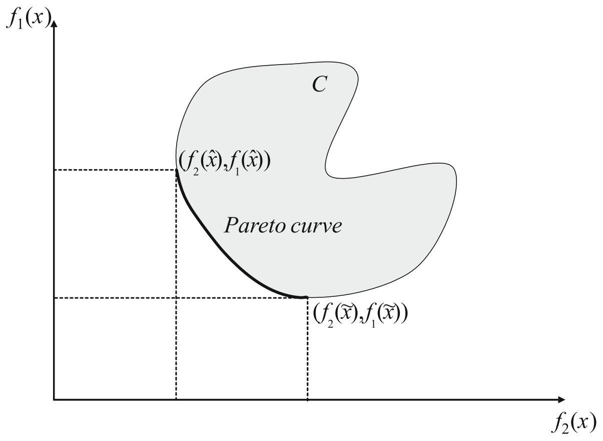

$f_1(x)$
C
$(f_2(\hat{x}),f_1(\hat{x}))$
Pareto curve
$(f_2(\tilde{x}),f_1(\tilde{x}))$
$f_2(x)$

Fig. 20.2 An example of Pareto-front. The figure is adapted from [3] with permission

is called *non-dominated* if all other solutions $\mathbf{x} \in \mathbf{S}$ have higher value for at least one objective function f_i or have the same value for all the objective functions [3].

For a two-objective problem we can plot the objective values of solutions (also denoted as *Pareto-front*) in a x-y plane. An example of a *Pareto curve* is shown in Fig. 20.2, where all the points on the Pareto curve between $\left(f_2\left(\hat{x}\right), f_1\left(\hat{x}\right)\right)$ and $(f_2\left(\tilde{x}\right), f1\left(\tilde{x}\right))$ are called *non-dominated* points.

20.5 Our Multi-objective Meta-Analytic Method for Churn Prediction

In this section we specify the different components of our approach: the different objective functions to be used, the type of classifiers to be employed and so forth.

We have chosen 28 heterogeneous base classifiers available in the WEKA data mining software, version 3.7 [13]. We only chose those single classifiers which are not meta-classifiers and are capable of handling the churn dataset. The list of classifiers is given in Table 20.1.

20.5.1 Objective Selection for Multi-objective Optimization

Most of the research on ensemble generation using multiple-objectives is confined to a single scalar function using a hyper-parameter [16]. It is a common practice to tune the hyper-parameter (λ) to balance between two commonly used objectives, Obj_A and Obj_B to get the scalar value of objectives Obj_{Sc} as:

$$Obj_{Sc} = Obj_A + \lambda Obj_B. \tag{20.8}$$

The conversion of multi-objective into single objective usually introduces some weakness. A predefined hyper-parameter is required to be set in order to achieve the right balance between Obj_A and Obj_B. Moreover, transforming this outcome into single-objective approach is limited to type of the problem. We do need the multi-objective problem to actually optimize relative to multiple conflicting objectives. Best individuals from a multi-objective evolution will form the Pareto front of a size equal to the number of objectives.

Table 20.1 List of 28 base classifiers to be used for the experiments

Type	Count	Base classifiers
Bayes	2	`BayesNet`
		`NaiveBayes`
Support vector machine	2	`SMO`
		`SPegasos`
Linear	3	`Logistic`
		`SimpleLogistic`
		`SGD`
Neural network	3	`MLPClassifier`
		`RBFNetwork`
		`VotedPerceptron`
Decision tree	7	`ADTree`
		`BFTree`
		`HoeffdingTree`
		`J48`
		`LADTree`
		`REPTree`
		`PART`
Decision stump	1	`DecisionStump`
Trees	4	`FT`
		`LMT`
		`RandomTree`
		`SimpleCart`
Nearest neighbours	1	`IBk`
Rule learner	3	`ConjunctiveRule`
		`JRip`
		`Ridor`
Decision table	1	`DecisionTable`
Feature interval	1	`VFI`

20.5.1.1 Definition of Objective Functions

In our case, we analyse three commonly used objective values in the literature, namely classification performance measure, diversity score and ensemble size. Objectives based on these values conflict with each other. We define the objective functions for our Multi-Objective Ensemble of Classifier (MO-EoC) as follows:

Classification Performance Measure: Utilizing the classification performance measure, we maximize the average Matthews Correlation Coefficient (MCC) score for the ensemble of classifier as described in [15]. Hence, the objective function can be written as:

Objective 1 ***MCC:*** *Given an ensemble $\mathbb{E}$ with k base classifiers. For each base classifier $\mathbb{C}_i$ in the ensemble*

$$\textbf{(Maximize)} \quad Obj_{mcc}(\mathbb{E}) : \frac{1}{k}\sum_{i=1}^{k} MCC\,(\mathbb{C}_i\,(\mathbb{T}_s) \in \mathbb{E}) \qquad (20.9)$$
$$subject\ to \quad MCC\,(\mathbb{E}) \geq 0.0,$$

where the MCC score is calculated using 60–40 split on the selected feature subset $\mathbb{T}_s$. A valid solution should not perform worse than a random predictor (with an MCC score equals to at least 0.0).

The outcome of optimizing this objective will find the ensemble combination providing a maximum average MCC score. Hence, the optimization of Obj_{mcc} function will be a maximization problem.

Diversity Measure of Base Classifiers: In the classification problem, the diversity score represents the agreement or disagreement among the base classifiers' decisions. The most diverse classifiers will predict different labels for the same input data. In the literature about the ensemble of classifiers, the diversity among base classifiers is not well defined as classification performance [31]. Furthermore, [19] experimented with available diversity measures and found that some of them may mislead the classification learning. This work divides the diversity measures into two categories: pairwise and non-pairwise diversity.

In the case of pairwise diversity measures, the score is calculated for the pair of base classifiers. The calculation of pairwise diversity measure becomes computationally expensive for ensembles created with a large number of base classifiers and also for large datasets. Q-statistics, correlation coefficient and k-statistics are commonly used pairwise diversity scores [1, 26]. On the other hand, the non-pairwise diversity score is calculated over a set of classifiers for training performance. The non-pairwise diversity measures can be calculated easily as a group for a large number of base classifiers in contrast to the pairwise measures which consider each pair of base classifiers. Kohavi–Wolpert variance, inter-rater agreement, generalized diversity and entropy are commonly used non-pairwise measures of diversity [5, 25]. In our case, we have 28 heterogeneous base classifiers, which is the largest ensemble of classifiers considered so far.

We use the widely used entropy score for ensemble of classifiers as our measure of diversity [19]. The value of diversity for an ensemble $\mathbb{E}$ on training dataset $\mathbb{T}_s$ is calculated as:

$$\frac{1}{m}\sum_{j=1}^{m}\frac{1}{(k-\lceil\frac{k}{2}\rceil)}\min\{l(z_j), k-l(z_j)\}. \tag{20.10}$$

Here, k denotes the number of base classifiers $\mathbb{C}$ in the ensemble, m is the number of samples in training dataset, $l(z_j)$ denotes the number of base classifiers that recognizes a sample z_j correctly and $\lceil\frac{k}{2}\rceil$ denotes the votes from base classifiers with same class label.

Objective 2 ***Diversity:*** *Given an ensemble combination $\mathbb{E}$ with k number of base classifiers, the diversity as per Eq.* (20.10) *is defined as*

$$(\textbf{Maximize}) \quad Obj_{div}(\mathbb{E}) : Diversity\,(\mathbb{E})\,, \tag{20.11}$$

where the $Diversity$ score is calculated on the training portion $\mathbb{T}_s$ of the dataset.

An ensemble will be assessed for the entropy score attained by the combination of base classifiers. Hence, the optimization of Obj_{div} function will also be a maximization problem.

Ensemble Size: In multi-objective optimization of ensemble classifiers, many researchers used ensemble size (the number of base classifiers in the ensemble combination) as an objective value [1, 8, 9, 23] to minimize it (ensemble size or cardinality). The objective function for minimizing size is defined as:

Objective 3 ***Size:*** *Given an ensemble combinations $\mathbb{E}$ and a value k identifying the length of the binary string that represents the combination of base classifiers, the objective function for the ensemble size is written as:*

$$(\textbf{Minimize}) \quad Obj_{sz}(\mathbb{E}) : |\mathbb{E}| \mapsto \sum_{i=1}^{k}\mathbb{E}_i\;[\forall_i, \mathbb{E}_i = 1] \tag{20.12}$$
$$subject\ to \quad |\mathbb{E}| \geq 2,$$

where the ensemble size is mapped into the number of elements $\mathbb{E}_i$ having non-zero values in the respective position in the string of base *classifiers. To be considered valid, a solution needs to have at least two base classifiers.*

This optimization of Obj_{sz} is a minimizing task which will reduce the size of the ensemble.

20.5.2 The MO-EoC Framework

A multi-objective evolutionary algorithm ($MOEA$) deals with mathematical optimization problems which require the optimization of more than one criterion simultaneously. MOEAs have been applied successfully in many areas where optimal decisions need to be taken in the presence of trade-offs between two or more conflicting objectives. Unlike in single-objective optimization, MOEA provides a number of Pareto-optimal solutions. As previously noted, a solution is called non-dominated if it is impossible to improve one objective value without degrading some other objective value. For this reason, all Pareto-optimal solutions are treated as equally good. However, a final solution is selected from the Pareto-optimal solutions depending on the subjective preference (trade-offs in satisfying different objectives) of the decision maker.

20.5.2.1 NSGA-II

Among many multi-objective optimization algorithms, we have chosen the widely used NSGA-II (Non-dominated Sorting Genetic Algorithm II) [7] which is an upgraded version of the NSGA algorithm [28]. The NSGA-II MOEA poses several advantages over its predecessor. It improves the NSGA by adopting a fast non-dominated sorting approach and a crowded comparison operator which helps the algorithm to run faster, facilitates the application of the elitist principle, emphasizes non-dominating solutions, maintains population diversity and converges to solutions close to the Pareto-optimal solutions. Hence, the NSGA-II algorithm can simultaneously handle the optimization of multiple objective functions and represents one of the leading Pareto-optimal solution algorithms. It has been successfully applied in various optimization settings, including recently in resource allocation [22], data classification [21], biomarker discovery [30] and biological network analysis [32].

Algorithm 1 provides a pseudocode of NSGA-II. At first a parent population Pop containing Pop_{sz} individuals is randomly initialized. Each individual encodes the problem with a string of size Prb_{sz}. Then, genetic variation operations of *selection*, *recombination* and *mutation* are applied to generate offspring to enter the population ($ChPop$) with the size of Pop_{sz}. The algorithm starts by merging both populations into one population ($MPop$). Then, the combined population is subjected to a sorting procedure with `FastNondominatedSort` function (as shown in Algorithm 2 in the Appendix) to identify non-dominated solutions. The

`SortByRankAndDistance` function aligns the population into a hierarchy of non-dominated Pareto fronts, assigning the best rank to a solution which is not dominated by any other solutions. The average distance between individuals in each front is calculated using `CrowdingDistanceAssignment` described in Algorithm 4 (in the Appendix). Then, a function for discriminating individuals in the population is used according to their rank which preserved the elitism. The `SelectParentsByRankAndDistance` is used for ordering Pareto-front solutions first by their dominance precedence and then by the distance within the front. Then, the new population of offspring is generated after applying genetic variation operators (recombination and mutation). These procedures are repeated until the stopping criteria are satisfied.

Algorithm 1: Pseudocode of `NSGA-II` algorithm

```
   Input: $Pop_{sz}$, $Prb_{sz}$, $R_\chi$, $R_\mu$
   Output: $ChPop$
1  $Pop \leftarrow$ InitializePopulation($Pop_{sz}$, $Prb_{sz}$)
2  EvaluateAgainstObjectiveFunctions($Pop$)
3  FastNondominatedSort($Pop$)
4  $SPop \leftarrow$ SelectParentsByRank($Pop$, $Pop_{sz}$)
5  $ChPop \leftarrow$ RecombinationAndMutation($SPop$, $R_\chi$, $R_\mu$)
6  while $\neg$StopCondition do
7      EvaluateAgainstObjectiveFunctions($ChPop$)
8      $MPop \leftarrow$ Merge($Pop$, $ChPop$)
9      $\mathbb{F} \leftarrow$ FastNondominatedSort($MPop$)
10     Parents $\leftarrow \emptyset$
11     $F_L \leftarrow \emptyset$
12     foreach $F_i \in \mathbb{F}$ do
13         CrowdingDistanceAssignment($F_i$)
14         if Size(Parents)+Size($F_i$) $> Pop_{sz}$ then
15             $F_L \leftarrow i$
16             Break()
17         else
18             Parents $\leftarrow$ Merge(Parents, $F_i$)
19         end if
20     end foreach
21     if Size(Parents) $< Pop_{sz}$ then
22         $F_L \leftarrow$ SortByRankAndDistance($F_L$)
23         for $P_1$ to $P_{Pop_{sz}-\text{Size}(F_L)}$ do
24             Parents $\leftarrow Pi$
25         end for
26     end if
27     $SPop \leftarrow$ SelectParentsByRankAndDistance(Parents, $Pop_{sz}$)
28     $Pop \leftarrow ChPop$
29     $ChPop \leftarrow$ RecombinationAndMutation($SPop$, $R_\chi$, $R_\mu$)
30 end while
31 return $ChPop$
```

Algorithm 2 shows how we transform the population into non-dominate sorting population. The crowding distance of the non-dominating solutions and their rank are calculated and preserved (lines 4–13). The ranking process of each solution (shown in lines 8–10) in the population is calculated based on their dominance depth. Solutions in the population get the value of *rank* and *crowdingDistance* attributes.

The process of `CrowdingDistanceAssignment` is shown in Algorithm 4 (also in the Appendix). The calculation of crowding distance is restricted by the size of the non-dominated solution front. It returns a positive infinity value for front size ≤ 3 (lines 2–4). Otherwise, we initialize each solution in the fronts with zero distance. For each objective, the crowding distance is calculated for the front (lines 9–20). Here, the front ($\mathbb{F}$) is sorted according to the objective value. For each solution in the front, we update the crowding distance based on its neighbourhood (lines 16–19). Generally, solutions which are far away (not crowded) from other solutions are given a higher rank. The ranking is produced this way to generate a diverse solution set. The procedure returns the front with the associated crowding distances based on the objectives.

Finally, the parent selection method based on the rank and distance is shown in Algorithm 3 (in the Appendix). The parent selection process starts with a randomly selected solution. Then it is compared against other $arity$ sized randomly selected candidates (lines 3–7) for rank and distance. The winning solution is preserved in the population of parents. The process is repeated until the number of parents has reached the arity value.

20.5.2.2 The MO-EoC Using NSGA-II

In our design, we used 28 base classifiers (listed in Table 20.1) from the WEKA data mining software suite [13] to create the ensemble combinations. The method named `EvaluateAgainstObjectiveFunctions` evaluates each pair of objectives listed in Sect. 20.5.1.1. We use a binary string to represent a solution and to encode a pair of objectives inside the multi-objective optimization algorithm. The NSGA-II implementation is taken from the `MOEA Framework` [12], version 2.7, available online.[2] We use the binary presentation of the solution with `Half-uniform-Recombination` (HUX) and `BitFlip` (BF) mutation operators as variation functions of the NSGA-II algorithm. In the case of the HUX recombination operation, half of the non-matching bits are swapped in between the two parents. The probability rate of mutation (R_μ) and recombination (R_χ) is kept unchanged from the default value in the MOEA Framework. Table 20.2 shows the parameter values used for the execution of the MO-EoC algorithm.

[2]http://moeaframework.org.

Table 20.2 Parameter settings of the proposed MO-EoC using NSGA-II

Parameter	Value
Individual type	Binary string
Individual length	28
Population size	100
Maximum evaluation	10,000
Recombination strategy	HUX
Recombination rate (R_χ)	0.75
Mutation strategy	BF
Mutation rate (R_μ)	0.10
A pair of objectives	$\{Obj_{mcc}, Obj_{div}, Obj_{sz}\}$

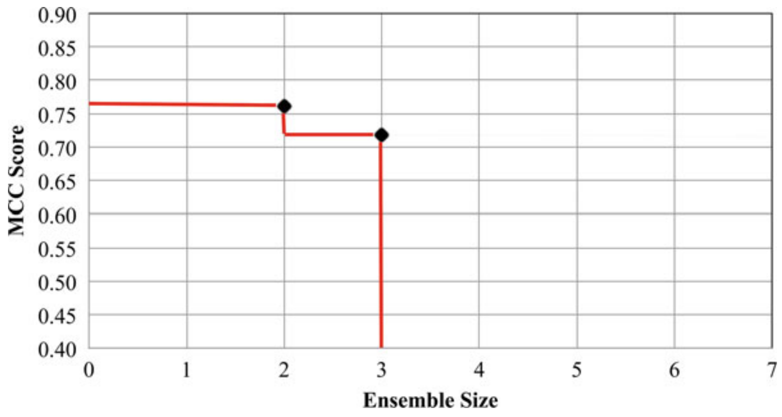


Fig. 20.3 Pareto optimal solutions for optimizing objective pairs of (MCC, Size) for Churn dataset

20.5.3 Computational Results

In each iteration of the NSGA-II algorithm, objectives are taken in pairs from $\{Obj_{mcc}, Obj_{div}, Obj_{sz}\}$ and each pair is evaluated for the Pareto front composed of non-dominated solutions. Then the goodness of those solutions is evaluated to decide the best objective pairs to use in the MO-EoC.

20.5.3.1 MCC vs Size

We evaluate the objective pairs (Obj_{mcc}, Obj_{size}) for our Churn detection dataset by first plotting the values of objective pairs for each of the Pareto-optimal solutions for the datasets as shown in Fig. 20.3. Here, the X-axis identifies the ensemble size and Y-axis identifies the MCC score of the non-dominating solution.

Figure 20.3 shows the scatter plot of objective scores pair (Obj_{mcc}, Obj_{size}) for *Churn* dataset. Here, for 30 independent runs, the experiment reveals only two solutions. One is for an ensemble size of 2 (with an MCC score of 0.72) and the other is for an ensemble of size 3 (with an MCC score of 0.76).

For this experiment, we found that the ensemble size and MCC score are not conflicting objectives and pairing them together did not produce any advantages for enhancing the MCC scores. No study has previously used MCC scores as a performance measure in evaluating the multi-objective ensemble of classifiers. We observe that in our pairwise optimization of the MCC score and ensemble size, the change of the MCC score for increasing the ensemble size is always parallel to the horizontal axis. Hence, the MCC score optimization of (Obj_{mcc}, Obj_{size}) does not demonstrate a conflict between the MCC objective and adhere to the conflicting objective of ensemble size objective in the multi-objective optimization.

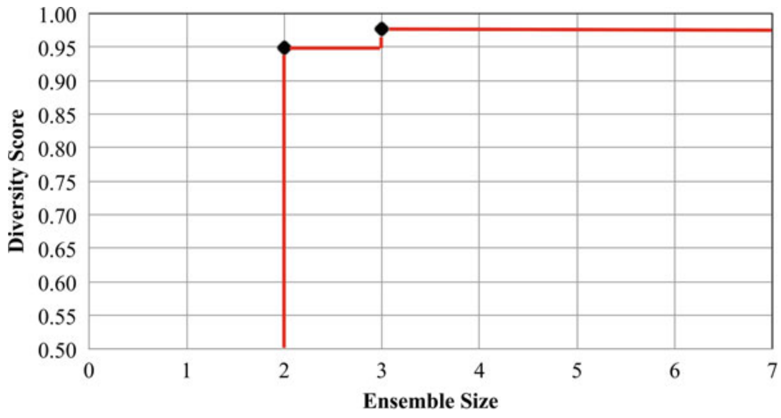


Fig. 20.4 Pareto optimal solutions for optimizing objective pairs of (Div, Size) for Churn dataset

20.5.3.2 Diversity vs Size

Now, we will evaluate the objective pairs (Obj_{div}, Obj_{size}) for the *Churn* dataset. We plot the Pareto-optimal solutions for the optimization of objective pairs in Fig. 20.4. Here, the X-axis identifies the ensemble size and the Y-axis identifies the diversity value of the non-dominating solutions.

Figure 20.4 shows the scatter plot of objective scores pair (Obj_{div}, Obj_{size}) for the churn dataset. Here, all solutions found in 30 repetition grouped into the ensemble of size 2 and 3 with two different diversity scores. The size three ensembles produced slightly better diversity scores than that of size two ensembles. Hence, the outcome has mostly been biased by the minimization of the ensemble size. The disagreement between the base classifiers decision in the solution ensemble with lower size will have more diversity score in compared with the disagreement by a single base classifier while in large ensemble. Aksela and Laaksonen [2] pointed out that pairwise optimization of diversity and ensemble size always leads to minimize the number of base classifiers. Our result also affirms their observation. Moreover, the optimization of (Obj_{div}, Obj_{size}) pair does not lead to a better generalization (MCC score of 0.586 and 0.598 for solution with 2 and 3 base classifiers, respectively). Therefore, the diversity or the ensemble size without pairing with an objective related to the performance measure will not lead to a better classification outcome.

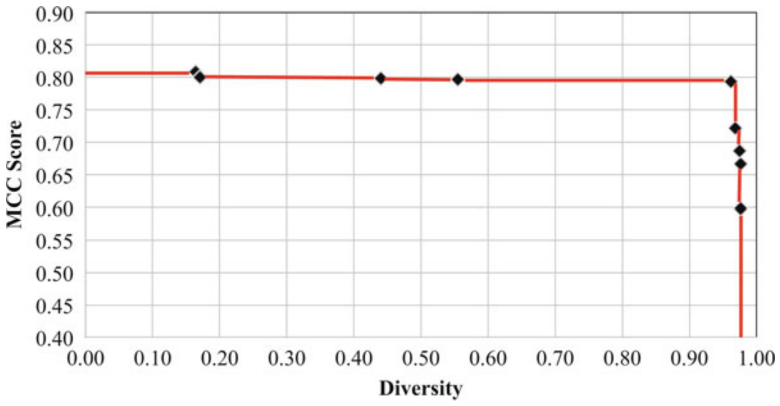


Fig. 20.5 Pareto optimal solutions for optimizing objective pairs of (MCC, Div) for Churn dataset

20.5.3.3 MCC vs Diversity

Now, we evaluate the objective pairs (Obj_{mcc}, Obj_{div}) for the *Churn* dataset. We plot the values of objective pairs for the Pareto-optimal solutions in Fig. 20.5. The X-axis identifies the Diversity score and Y-axis identifies the MCC scores achieved by non-dominating solutions.

Figure 20.5 shows the objective scores for optimizing the pair (Obj_{mcc}, Obj_{div}) in *Churn* dataset. As can be seen from the graph, similar MCC scores (in the range of 0.79–0.80) have been achieved for solutions with different diversity scores. Additionally, solutions with similar diversity scores (in the range of 0.96–0.97) have attained different MCC scores. It is also noticeable that, for Churn dataset, the diversity score of solutions varies in the certain range. Most importantly, lower diversified solutions (within the range) produced better MCC scores. Therefore, it is more prudent to choose a solution having higher MCC score but lower diversity value in case of solution selection from the Pareto-front.

The proposed method applies the genetic algorithm to evaluate three pairs of objectives in multi-objective settings: The first pair to optimize the MCC and the ensemble size, the second pairs of objectives are the ensemble size and Diversity scores, and the final is to optimize the MCC and diversity scores, respectively. The experimental outcomes revealed that the ensemble size or diversity without pairing with any performance measure does not have a significant effect on increasing predictive performances of the ensemble of classifiers. It is also observed that the pairwise optimization of the MCC and the diversity is the most suitable objective pair for ensemble combination search in multi-objective settings. So, we only consider the pair (Obj_{mcc}, Obj_{div}) as the best objectives for MO-EoC for classifying the *Churn* dataset.

20.6 Classifying Churners with MO-EoC Algorithm

We present the generalization performance of proposed MO-EoC for churner detection using the multi-objective settings in this section. The objectives used in the experiment are the maximization of both MCC and Diversity scores. The summary of threefold cross-validation experimental outcomes of MO-EoC algorithm in 30 independent runs is shown in Table 20.3.

The summary table shows the median MCC score is 0.79 with average of 0.74 for 30 independent runs. From the standard deviations it is evident that the performance of MO-EoC for churner prediction is stable for different classification measures (such as MCC, accuracy, precision and F-Measures).

Table 20.3 Summary statistics of the performances of MO-EoC algorithm for Churn dataset using threefold cross-validation of 30 independent runs

Statistic	MCC	Accu	Prec	FMeas
Minimum	0.53	89.38	0.89	0.89
First quartile	0.69	92.08	0.92	0.92
Third quartile	0.80	95.26	0.95	0.95
Mean	0.74	93.62	0.94	0.93
Median	0.79	95.17	0.95	0.95
Maximum	0.81	95.50	0.95	0.95
Variance	0.01	4.10	0.00	0.00
Stdev	0.07	2.02	0.02	0.02

The performances are measured in MCC, accuracy, precision and F-measure scores

To depict the distribution of predictive performances of MO-EoC found in 30 independent runs considering different measures, we create box-and-whisker plot in Fig. 20.6. The plot shows the performances considering different measures: (a) MCC, (b) Accuracy, (c) Precision and (d) F-Measure scores. From Fig. 20.6a, we found that 25th percentile of the MCC scores are above 0.7. For the accuracy score in Fig. 20.6b, the 25th percentile value is 92%. The spread of precision (Fig. 20.6c) and F-Measure (Fig. 20.6d) scores are very narrow. All of these results are apparent for the MO-EoC as a consistent method in churner classification.

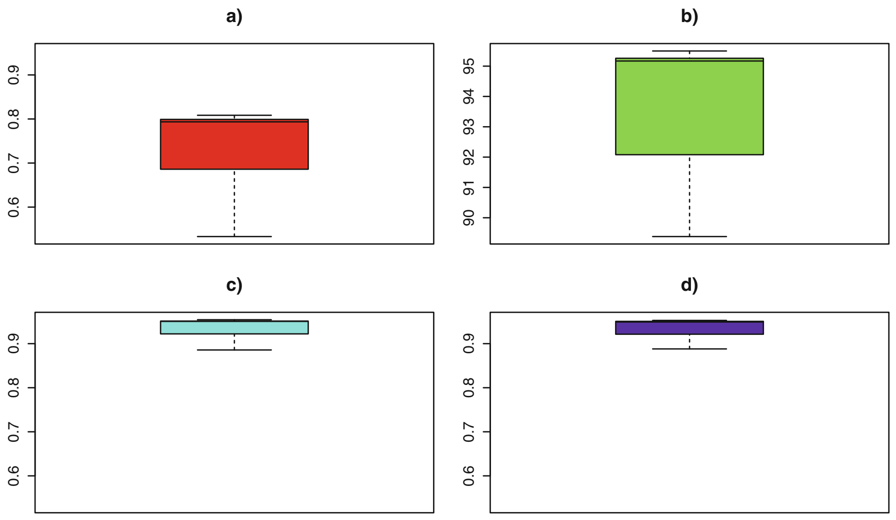


Fig. 20.6 Box-and-whisker plot to show the classification performances of the MO-EoC for Churn dataset using stratified 3-fold cross-validation of 30 independent runs. The performances are measured in (**a**) MCC, (**b**) Accuracy (%), (**c**) Precision and (**d**) F-measure scores

20.6.1 Churner Prediction Performances of Base Classifiers

Table 20.4 shows the classification performances (considering various measures) of base classifiers for threefold cross validation on Churn dataset. Here we report the churner prediction performances of base classifiers for Matthews correlation coefficient (MCC), Accuracy (Accu), Precision (Prec), F-Measure (FMeas), Area Under the Curve (AUC) [20], Sensitivity (SEN) and Specificity (SPEC) scores. The result summary shows the Maximum (Max.), Median and Minimum (Min.) value of each score for 28 base classifiers.

As can be seen from the table, `JRip` classifier attains the maximum MCC score (0.787) among 28 base classifiers. It achieves the best scores for all performance measures except for the AUC and SPEC measures. The `ADTree` classifier achieves the best AUC measure value of 0.899 and the `VFI` classifier achieves the best SPEC score with 0.829.

We plot the generalization performances of 28 base classifiers in Fig. 20.7 for MCC scores. From the graph it is clear that the `JRip` classifier achieves the best score. However, three well-known and widely used classifiers (namely `BFTree`, `REPTree` and `SimpleCart`) are unable to perform well in generalization of churn detection dataset. It indicates that a single best classifier cannot perform better in all

Table 20.4 Classification performances of 28 base classifiers using threefold cross validation on the Churn dataset

Base classifier	MCC	Accu	Prec	FMeas	AUC	SEN	SPEC
BayesNet	0.423	86.35	0.857	0.860	0.837	0.863	0.538
NaiveBayes	0.488	87.91	0.873	0.876	0.832	0.879	0.584
SMO	0.250	86.23	0.833	0.825	0.566	0.862	0.270
SPegasos	0.254	86.23	0.833	0.826	0.569	0.862	0.275
Logistic	0.324	86.41	0.840	0.843	0.806	0.864	0.368
SimpleLogistic	0.199	85.78	0.821	0.815	0.779	0.858	0.238
SGD	0.242	86.05	0.828	0.824	0.566	0.860	0.271
MLPClassifier	0.363	84.49	0.842	0.843	0.769	0.845	0.511
RBFNetwork	0.524	88.99	0.883	0.885	0.844	0.890	0.593
VotedPerceptron	0.118	85.75	0.864	0.795	0.521	0.857	0.161
ADTree	0.660	92.05	0.916	0.918	**0.899**	0.920	0.692
BFTree	0.000	85.51	0.731	0.788	0.500	0.855	0.145
HoeffdingTree	0.488	87.85	0.873	0.876	0.835	0.878	0.587
J48	0.733	93.85	0.936	0.935	0.885	0.938	0.713
LADTree	0.687	92.71	0.923	0.924	0.893	0.927	0.707
REPTree	0.000	85.51	0.731	0.788	0.500	0.855	0.145
Part	0.563	89.68	0.892	0.894	0.806	0.897	0.635
DecisionStump	0.294	85.87	0.831	0.836	0.601	0.859	0.352
FT	0.565	89.89	0.893	0.895	0.810	0.899	0.623
LMT	0.704	93.22	0.929	0.928	0.886	0.932	0.688
RandomTree	0.280	86.59	0.841	0.831	0.613	0.866	0.286
SimpleCart	0.000	85.51	0.731	0.788	0.500	0.855	0.145
IBk	0.214	83.05	0.807	0.816	0.591	0.830	0.351
ConjunctiveRule	0.374	86.68	0.848	0.853	0.770	0.867	0.436
JRip	**0.787**	**94.99**	**0.948**	**0.948**	0.863	**0.950**	0.778
Ridor	0.743	94.09	0.939	0.937	0.824	0.941	0.706
DecisionTable	0.530	89.02	0.884	0.886	0.844	0.890	0.605
VFI	0.023	18.51	0.782	0.123	0.681	0.185	**0.829**
Max	0.787	94.99	0.948	0.948	0.899	0.950	0.829
Median	0.368	86.50	0.853	0.848	0.792	0.865	0.525
Min	0.000	18.51	0.731	0.123	0.500	0.185	0.145

The best results for each of the metrics are shown in boldface

cases but for some cases. So, there is a need of the quest for a classification system that performs consistently well for all cases.

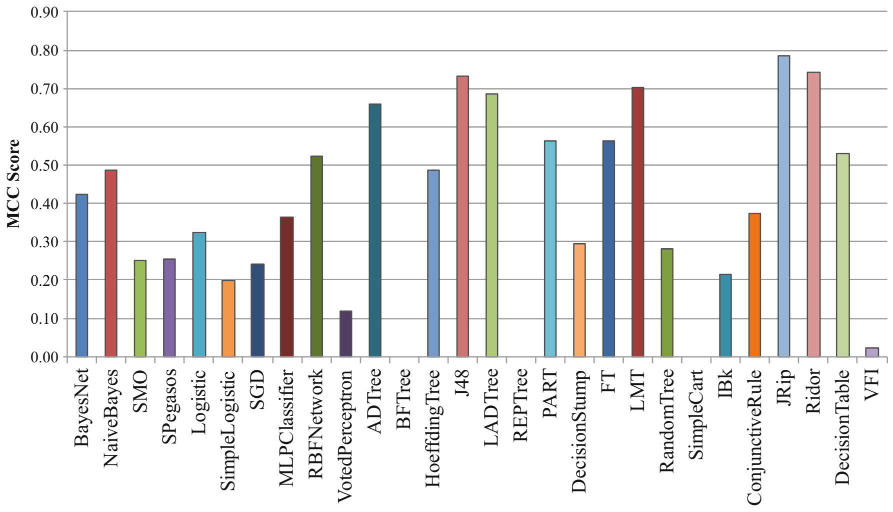


Fig. 20.7 Generalization performances of base classifiers on threefold cross-validation on Churn dataset for MCC scores

20.6.2 Churner Prediction Performances of State-of-the-Art Ensemble of Classifiers

Now we compare the classification performances of state-of-the-art ensemble of classifier algorithms available in WEKA software package. We examined the performance of `AdaBoostM1`, `Bagging`, `RandomForest`, `RandomCommittee` and `Stacking` ensemble of classifiers from WEKA. For comparing the performance of MO-EoC with those state-of-the-art ensemble methods, we used the average performance achieved by our approach. We kept the default parameters unchanged for those ensemble classifiers from the WEKA framework to make the results reproducible.

Table 20.5 shows the comparison of churn prediction performances using various measures for different ensemble of classifiers. It can be seen from the table that the `AdaBoostM1` performed best among state-of-the-art ensemble methods with 0.310 MCC score while considering the MCC measure. Considering the Accuracy, the `RandomCommittee` achieves the best accuracy of 86.44%. In terms of precision measure, `RandomForest` is the best performing ensemble. Finally, considering the F-Measure score, the `AdaBoostM1` opted as the best state-of-the-art ensemble with 0.839 score. Now we consider the average performance of MO-EoC in 30 independent runs and it clearly outperforms all state-of-the-art ensemble methods for all prediction measures used in the experiment.

Table 20.5 Comparison of classification results of MO-EoC (average) and other ensemble of classifiers

Ensemble classifier	MCC	Accu	Prec	FMeas
AdaBoostM1	0.310	86.35	0.837	0.839
Bagging	0.000	85.51	0.731	0.788
RandomForest	0.119	85.75	0.878	0.794
RandomCommittee	0.232	86.44	0.862	0.813
Stacking	0.000	85.51	0.731	0.788
MO-EoC (Avg.)	**0.737**	**93.618**	**0.936**	**0.935**

The best results for each of the metrics are shown in boldface.
All classifiers were tested in the test-set using threefold cross-validation. The performance is measured for the Matthews Correlation Coefficient (MCC), Accuracy (Accu), Precision (Prec) and F-Measure (FMeas) scores

20.6.3 Discussion

In general, the ensemble of classifiers achieves better generalization performances than its associated base classifiers. Now we examine this phenomenon for our proposed MO-EoC. Let us consider the best performances of base classifiers shown in Table 20.4. There we found that the best MCC score 0.787 is achieved by `JRip` which is nearly equal to the median MCC 0.79 of MO-EOC achieved in 30 independent runs. Besides this similarity, the standard deviation for different classification performance measures for MO-EoC is reasonably low (0.07 for MCC, 2% in Accu, 0.02 for both the Prec and F-Measures) for 30 independent runs. These low deviancy and better performances of MO-EoC are evident for its stability and reliability, respectively, in classification of churners from non-churners.

Now we would like to deeply analyse the results in Pareto-front for the optimization of the ensemble combinations by MO-EoC for churn dataset. First, we arbitrarily chose one set of Pareto-optimal solutions from 30 independent runs to observe the training versus testing performance for those solutions. Next we will analyse which base classifiers are appearing in those Pareto-optimal solutions.

In the chosen Pareto-front we find nine Pareto-optimal solutions shown in Table 20.6. Now we see how base classifiers appeared in the ensemble combinations of MO-EoC. We found many base classifiers are not selected in each of the nine ensemble combinations. `BayesNet`, `NaiveBayes`, `SMO`, `Logistics`, `IBk` are some of them. The `FT` and `Ridor` classifiers appear in most of the solutions (in seven out of nine solutions). Most of the Pareto-optimal solutions by MO-EOC are formed with three base classifiers (which are the smallest ensembles in size). The largest ensemble combination (marked as e6) has seven base classifiers. So we find that a small number of base classifiers are selected from 28 available base classifiers to formulate the ensemble combinations in the Pareto-front from MO-EoC.

Now we sort these nine ensemble combinations in order of their diversity score in training and plotted corresponding training MCC score (TrainMCC) with Testing MCC score (TestMCC) in Fig. 20.8. From this observation we found that the

Table 20.6 Variations of base classifiers appeared in nine Pareto-optimal solutions of a run of MO-EoC

Base classifier	e1	e2	e3	e4	e5	e6	e7	e8	e9	Count
BayesNet	0	0	0	0	0	0	0	0	0	0
NaiveBayes	0	0	0	0	0	0	0	0	0	0
SMO	0	0	0	0	0	0	0	0	0	0
SPegasos	0	1	0	0	0	0	1	0	1	3
Logistic	0	0	0	0	0	0	0	0	0	0
SimpleLogistic	0	0	0	0	0	0	0	0	0	0
SGD	0	0	0	0	0	0	0	0	0	0
MLPClassifier	0	0	0	0	0	1	0	0	0	1
RBFNetwork	0	0	0	0	0	0	0	0	0	0
VotedPerceptron	0	0	0	0	0	0	0	0	0	0
ADTree	0	0	0	0	0	0	0	0	0	0
BFTree	0	0	0	0	0	0	0	0	0	0
HoeffdingTree	0	0	0	0	0	0	0	0	0	0
J48	0	0	0	1	0	1	1	0	0	3
LADTree	0	0	0	0	0	0	0	0	0	0
REPTree	0	0	0	0	0	0	0	0	0	0
Part	0	0	0	0	0	0	0	0	0	0
DecisionStump	0	0	0	0	0	0	0	0	0	0
FT	0	1	1	1	1	1	1	0	1	7
LMT	1	0	0	0	1	1	0	1	0	4
RandomTree	0	0	0	0	0	0	0	0	0	0
SimpleCart	0	0	0	0	0	0	0	1	0	1
IBk	0	0	0	0	0	0	0	0	0	0
ConjunctiveRule	0	0	0	0	0	0	0	0	0	0
JRip	1	0	1	1	0	1	0	1	0	5
Ridor	1	0	1	1	1	1	0	1	1	7
DecisionTable	0	1	0	0	0	0	0	0	0	1
VFI	0	0	0	1	0	1	0	1	0	3
#Base classifier	**3**	**3**	**3**	**5**	**3**	**7**	**3**	**5**	**3**	

The number of base classifiers selected for each of the solutions is shown in boldface.
Nine solutions are marked as e1–e9 and selection of a base classifier is marked with 1 under the ensemble combination. The 'Count' column shows the number of times a base classifier has appeared in nine ensemble combinations

increment of diversity score has proportional effect with the training performance in MCC score but the testing performances have the opposite relationship. This asymmetric phenomenon found between the training and testing performances in this figure confirms that the MO-EoC is able to generalize the Churn prediction well.

Finally we conclude from the classification performance comparison between base classifiers, other state-of-the-art ensemble of classifiers and proposed MO-EoC for the churn prediction, it is apparent that the MO-EoC performed consistently

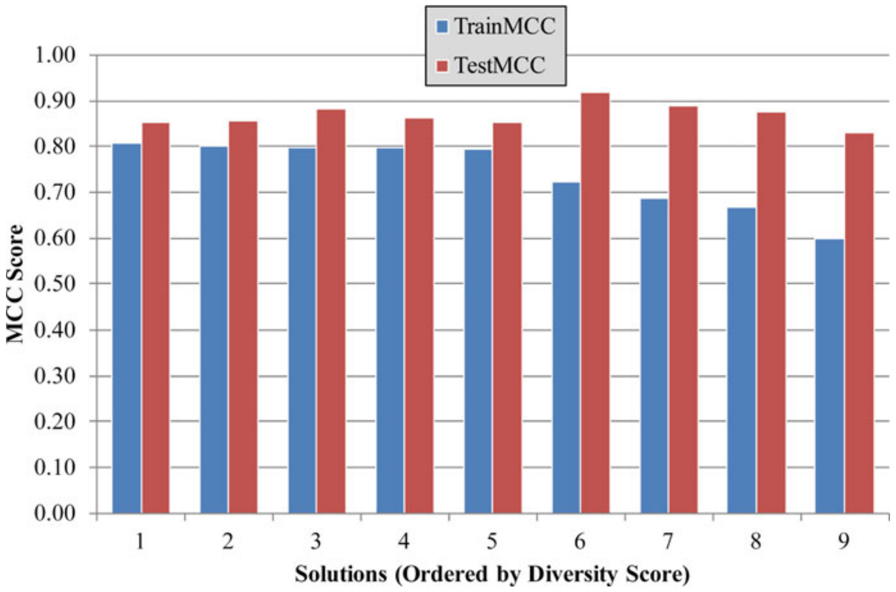


Fig. 20.8 Training and testing performance comparison of Pareto optimal solutions

in 30 independent runs. Moreover, the proposed multi-objective ensemble of classifier system finds ensemble combinations using few base classifiers from large available pool. Hence, the MO-EoC appeared as best method for churn classification compared to its base classifiers and other ensemble methods. We hope that it would perform consistently well for other classification problems in different areas.

20.7 Classifying Churners with Symbolic Regression Analysis

As an additional comparison method, we have used a symbolic regression approach, based on Genetic Programming, to classify and predict the churners from the non-churners. In symbolic regression, the objective is to find a model that fits a given set of input/output data without the user having to explicitly specify the model in advance. For this process we use a software tool called Eureqa from the company called Nutonian,[3] which is a mathematical symbolic regression software tool that has a very simple and useful interface and is free for academic use, thereby making it convenient for verifying the reproducibility of our results. The output of Eureqa is a Pareto optimality curve which shows several models. These models have a 'trade-off' between their capacity to fit the data well and their level of complexity. Eureqa has an ad hoc way of calculating the 'complexity' referred to as the 'size' of the

[3] www.nutonian.com.

models. A weighted function is calculated based on the number and the type of basic 'building blocks' that appear in model equation. Simple 'building blocks' such as addition, subtraction or the operation of introducing an input variable are seen as not very 'complex' and therefore the functions that use only a few of these arithmetic actions will have a relatively low 'complexity' score. Naturally, the opposite is true for more 'complex' building blocks such as quadratic functions. Further explanation of the 'building blocks' in Eureqa and how these have previously been selected by the authors can be found in [6].

Furthermore, the user needs to select the error metric that Eureqa will use as a guiding function to 'evolve' the model (which in turn will relate to the fitness functions used by the Genetic Programming approach). Consistent with their guidelines for binary classification tasks,[4] we used the 'Area under ROC Error' (AUC) as our guiding function, as we are indeed modelling a binary variable (churn or non-churn). To ensure that we will obtain a zero or one result, we 'force' a logistic function on the model. Finally, we select the row weight to equal the value $\frac{1}{occurrences(ChurnResult)}$ as the ratio between churners and non-churners is highly uneven. This step biases the evolutionary approach to consider the churners (of which there are very few samples) as they would provide a higher pay-off than predicting non-churners. This is a common and recommended practice to avoid having a very low number of True Positive predictions.

The threefold datasets were split at random into Training and Test sets. We obtain a Pareto frontier and then we select several models that we have found using only the samples from the training set and test their performance using the samples in the independent test set. For these models, we computed the same metrics (MCC and Accuracy) and we also give the 'Actual Churners Rate' to help us compare these results against the Multi-objective method described earlier in this chapter.

As stated, the Pareto optimality front trades off complexity for minimizing error (such a number is not user defined). To show the progression along this front, and for the purpose of comparison, we used four different models: the least performing model of the front (typically a function of just one variable), the best linear model (as the authors have previously done in [6]), the best model in which the variables still have a 100% positive or 100% negative (or both) variable sensitivity effect on the outcome variable (churn) and finally, the best (and therefore likely the most complex) model presented in the Pareto front.

For each fold, these four models are then tested in the equivalent test sets. To do this step we have to resort to a technique we have developed to address this issue. We first note that we used the AUC error metric to obtain the models in the Pareto frontier; this means that now we need a threshold value to classify the dataset into two distinct classes and evaluate its MCC and Accuracy. This threshold is model dependent and, unfortunately, Eureqa does not give this value to its users (although the value is expected to be provided in future releases as other users have already requested it too).

[4]http://formulize.nutonian.com/documentation/eureqa/tutorials/modeling-binary-outputs/.

We have reverse-engineered this threshold value using, again, Eureqa's functionality. We recall that we followed Eureqa's advice on the training set and evolved the argument of a logistic function. We will call this the 'argument function'. It is then evaluated on the test data, thus providing a value for each of the samples in the test data. We then use Eureqa on a task of fitting churn using these values and only the basic arithmetic actions of subtraction and addition of a constant and introduction of a variable while 'forcing' a logistic function over the whole equation which means we get a model that includes only one constant. The value thus obtained represents the threshold value that we need to use for the outcome of that model. This process was repeated for each of the twelve instances until we had rounded binary results for each that we could compare to the actual churn result in each of the test sets. After obtaining the classification (prediction) results, a confusion matrix is again used to calculate the Accuracy, the 'Actual Churner's Rate' and Matthews Correlation Coefficient (MCC) metrics and compare these to the other methods. This gives us a clear indication of which classification and churn prediction approach is of better quality.

Finally, unlike with the MO-EoC method, the symbolic regression analysis process provides the user with models and shows which variables are used in the prediction models. Therefore, we will conduct various ad hoc inspections to provide a greater understanding and interpretation of each of the models found and provide a brief discussion of this.

20.7.1 Classification Using Symbolic Regression Modelling

As explained in Sect.20.7, we used a symbolic regression modelling tool (Eureqa) to compare the MO-EoC method in terms of churn classification and churn prediction. The results of the evaluation metrics of this method are shown in Table 20.7.

The best results from the symbolic regression modelling process among the 12 instances are shown in bold.

20.7.2 Interpreting Symbolic Regression Results

Although it does not prove to be the most accurate method in terms of ACC and MCC metrics, one of the benefits of symbolic regression for churn prediction is the fact that the actual models and variables (features) used are identified and relationships between these features can be investigated to provide business managers with useful insights. Table 20.8 shows the top five most used features by Eureqa in each of the three folds. This shows us those variables that clearly have a strong relationship with our objective variable: churn.

In Table 20.8 we can see that the variable 'Customer Service Calls' is the most frequently occurring variable in all three folds, meaning it likely has a strong relationship with customer churn behaviour. Furthermore, whether or not the

Table 20.7 Classification results of Eureqa models in all three folds

Classification model	Accu	ACR	MCC
Fold 1 worst Eureqa model	0.689	0.311	0.052
Fold 1 best linear model	0.820	0.789	0.491
Fold 1 model with most 100% variable sensitivities	0.820	0.789	0.491
Fold 1 best Eureqa model	**0.890**	**0.764**	**0.611**
Fold 2 worst Eureqa model	0.590	0.171	0.062
Fold 2 best linear model	0.773	0.714	0.386
Fold 2 model with most 100% variable sensitivities	**0.865**	**0.876**	**0.606**
Fold 2 best Eureqa model	0.867	0.814	0.582
Fold 3 worst Eureqa model	0.572	0.584	0.108
Fold 3 best linear model	0.752	0.689	0.346
Fold 3 model with most 100% variable sensitivities	0.860	0.559	0.456
Fold 3 best Eureqa model	**0.869**	**0.789**	**0.575**

All four models for each fold were tested in the test set and the Accuracy (Accu), the 'Actual Churner's Rate' (ACR) and Matthews Correlation Coefficient (MCC) metrics were computed. The best results when taking all three metrics into account are shown in boldface

Table 20.8 The five most used variables by Eureqa for each fold

Fold 1	Fold 2	Fold 3
CustServ.Calls	CustServ.Calls	CustServ.Calls
Intl.Plan	Intl.Plan	Day.Mins
Day.Mins	Day.Charge	Intl.Plan
Eve.Mins	Eve.Charge	Eve.Mins
Intl.Charge	Intl.Charge	Intl.Charge

customer has an 'International Plan' comes in second place in two, and third place in one of the folds for being the most frequently used variable to predict churn. This provides insights to marketing managers in predicting those consumers who will likely churn, as with this information they are more likely to be able to identify them.

However, although it is interesting to know how many times certain features were included in the models, it is more interesting to know in what way they contributed, i.e., by exhibiting either a positive or a negative relationship with churn. Therefore, for each model, we used the variable sensitivity reports to inspect this. Eureqa's definition for the 'sensitivity' metrics it computes is 'The relative impact within this model that a variable has on the target variable'. As we have explained, we use this information to pick one of the models to investigate. For each of the folds, we picked the best model that still had only variables contributed either in a 100% positive or 100% negative way. As stated, we have twelve instances from taking four models from each of the three folds. However, the three 'worst' results were simply taken for evaluation and comparison purposes and do not provide much insight. We can state here that for each of the three folds, the 'worst' Eureqa results simply provided the logistic function of the variable 'Eve.charge'. As Table 20.7 shows, the results of this model were not very successful at predicting churners. Therefore we move on to investigate the other models.

In Tables 20.9, 20.10, 20.11, 20.12, 20.13 and 20.14 we show the variable sensitivity computations for the models of Eureqa. Firstly, in Table 20.9 we can see that the variable 'Day.Charge' has the highest sensitivity metric by far and is 100% positive. This may indicate to business managers that the higher the customer's 'Day Charge' was (for the telecom service), the more likely they are to churn.

Table 20.9 Variable sensitivity analysis for the best linear model of Fold 1

Variable	Sensitivity	% Positive	% Negative
Day.Charge	1.3729	100%	0%
Day.Mins	0.52689	0%	100%
CustServ.Calls	0.30847	100%	0%
Intl.Plan	0.30109	100%	0%

In this case, the best linear model was also the best model with still all 100% positive or negative sensitivities

Table 20.10 shows the sensitivity analysis for the best model of Fold 1. In this model, the variable 'International Plan' (which is a yes/no variable) has the highest sensitivity and only a positive effect on the predicted value of churn. This means that whenever this variable is higher (i.e., 'yes') churn is more likely to be higher (i.e., churner). However, we do note that some of these more complex models are extremely complicated and each variable individually does not predict the outcome. Rather, the models are 'fitted' by Eureqa to get a lower error metric. However, seeing whether the variables have a positive or negative influence on the total prediction of the model still provides insights into which features are more likely to lead to churners.

Table 20.11 provides the sensitivity analysis for the best linear model of Fold 2. As we can see, the feature with the highest sensitivity is 'Day Charge' again, like in Fold 1. Similar to the best model of Fold 1, here 'International Plan' is one of the variables with the highest sensitivities (here, the second one). Again, it has a fully positive effect which strengthens the finding that whether a customer has an international plan or not affects churn the most. If we could isolate this relationship appropriately, the company would be able to draw conclusions such as whether or not their International Plan was pleasing customers or whether it was receiving too much competition from other providers (which may be one of the reasons those consumers churned and left the company). However, again, like we have stated, these variables are inter-related in the complex models found and need to be taken into consideration together.

Table 20.10 Variable sensitivity analysis for the best model of Fold 1

Variable	Sensitivity	% Positive	% Negative
Intl.Plan	0.35172	100%	0%
VMail.Plan	0.32493	0%	100%
CustServ.Calls	0.21816	75%	25%
Eve.Mins	0.18969	100%	0%
Day.Mins	0.18901	82%	18%
Intl.Charge	0.011258	100%	0%

Table 20.11 Variable sensitivity analysis for the best linear model of Fold 2

Variable	Sensitivity	% Positive	% Negative
Day.Charge	0.62149	100%	0%
Intl.Plan	0.47438	100%	0%
CustServ.Calls	0.28836	100%	0%
Eve.Charge	0.24939	100%	0%
Night.Mins	0.13242	100%	0%
Intl.Charge	0.014844	100%	0%

Table 20.12 shows that again ‘Day Charge’ is the most sensitive variable in the model to predict churn. Also, ‘Customer Service Calls’ and ‘International Plan’ are again in the top three, with a mostly positive effect on the model outcome.

Next is the best linear model of Fold 3 shown in Table 20.13. Here a new variable has the highest ‘sensitivity’ value ‘Day Minutes’ (0.934 sensitivity) and is 100% positive. There are three other variables in this model all with much smaller sensitivity values.

Finally, Table 20.14 shows the sensitivity analysis for the best model found in Fold 3. Again ‘Day Minutes’ has the highest ‘sensitivity’ value, but this time with a split % of positive and negative influence on the model (54/46%, respectively). A mixture of other variables appears all with a 100% positive influence in the model. Interestingly, ‘International Plan’ and ‘Customer Service Calls’ have appeared frequently in the other models we have presented for predicting churn.

Table 20.12 Variable sensitivity analysis for the best model of Fold 2

Variable	Sensitivity	% Positive	% Negative
Day.Charge	0.50578	100%	0%
CustServ.Calls	0.47706	83%	17%
Intl.Plan	0.46105	100%	0%
Eve.Charge	0.25111	100%	0%
Night.Mins	0.13483	100%	0%
Intl.Charge	0.022098	100%	0%

Table 20.13 Variable sensitivity analysis for the best linear model of Fold 3

Variable	Sensitivity	% Positive	% Negative
Day.Mins	0.93535	100%	0%
Intl.Plan	0.26789	100%	0%
CustServ.Calls	0.26688	100%	0%
Intl.Mins	0.0043791	100%	0%

Seeing that several variables are repeatedly some of the most used and most positively ‘sensitive’ variables inside those models provides us with information to generate useful insights for churn prediction. We now know that some of these variables (e.g., Customer Service Calls, International Plan and Day Minutes) may

Table 20.14 Variable sensitivity analysis for the best model of Fold 3

Variable	Sensitivity	% Positive	% Negative
Day.Mins	0.58804	54%	46%
Intl.Plan	0.44497	100%	0%
CustServ.Calls	0.36845	100%	0%
Eve.Mins	0.21085	100%	0%
Night.Mins	0.11237	100%	0%
Intl.Charge	0.030137	100%	0%

play a more active role in predicting customer churn. We also know those variables that are used less and are therefore less contributing factors to successful churn prediction. For instance, ‘Account Length’, ‘Voicemail Message’, ‘Day Calls’, ‘Evening Calls’ and a few other variables are not used in any of the models in each of the threefolds. Although these may be important characteristics of the telecom service, in the predictive models we have found here, they are not needed to predict churners. This information could save time and effort for managers as they know where to place emphasis on pleasing customers, i.e., emphasizing those variables that are positively related to churn behaviour rather than those variables that do not affect churn. Besides investigating each of these variables, it shows that Eureqa is advantageous for churn prediction in terms of generating useful insights that can be turned into actions by managers.

20.8 Conclusions

In this chapter we have presented an ensemble learning method for classification and prediction. We have illustrated the effectiveness of this method on a customer churn dataset for the goal of predicting those customers who are going to churn. We have found good results in terms of accurate classification prediction with our evolutionary algorithm MO-EoC. Our approach appears to be the best method for churn classification compared to its base classifiers and other ensemble methods we have examined. Clearly, the predictive power of a dataset is directly related to the quality of the information that it provides (i.e., that can identify features relevant/correlated to causal variables that can explain the phenomenon in question). From this perspective, it is not unusual in the social sciences to ‘celebrate’ when prediction accuracies are above 0.6. In this case, having obtained MCC scores of 0.8 and above, we are very optimistic that the method is an important contribution.

However, one downside of our algorithm was the lack of interpretability it provides for decision makers. Therefore, we also included an analysis of the customer churn dataset using a symbolic regression approach (based on a commercial and academic-available-for-free genetic programming software which allows reproducibility). The analysis using this method obtained lower results in terms of ACC and MCC than the MO-EoC algorithm; however, our ad hoc analyses of the resulting models provide some actionable insights for business decision makers. As stated in Sect. 20.7.2, the variables ‘Customer Service Calls’, ‘International Plan’ and ‘Day Minutes’ were among the most likely variables to have an active role in predicting customer churn when no segmentation of the customer base is performed.

Finally, we may state that methods like these (the MO-EoC algorithm and symbolic regression analysis) together provide a comprehensive analysis and reliable outcomes for problems such as churn prediction. Accurate methods such as MO-EoC can be used to predict those customers likely to churn, and the further analysis using symbolic regression approaches can provide insight into how and what types of resources companies can invest in to accurately predict churners to prevent them from churning.

Acknowledgements Pablo Moscato acknowledges previous support from the Australian Research Council Future Fellowship FT120100060 and Australian Research Council Discovery Projects DP120102576 and DP140104183.

Appendix

This extended Appendix provides additional algorithms used in this chapter. Each of these algorithms intend to solve some sub-problems dealt in the main algorithm and readers are highly encouraged to investigate these algorithms for themselves for the continued journey and challenge for solving business and consumer analytics using NSGA-II.

Algorithm 2: Pseudocode of FastNon-dominatedSort

```
Input: The population $\mathbb{P}$
Output: Fronts $\mathbb{F}$
1  foreach $\mathbb{P}_i \in \mathbb{P}$ do
2      $\mathbb{R}$.Add($\mathbb{P}_i$)
3  end foreach
4  $rank \leftarrow 1$
5  while $\mathbb{R} \neq \phi$ do
6      $\mathbb{F} \leftarrow \phi$
7      foreach $\mathbb{R}_i \in \mathbb{R}$ do
8          $\mathbb{F}$.Add($\mathbb{R}_i$)
9      end foreach
10     foreach $F_i \in \mathbb{F}$ do
11         $\mathbb{R}$.Remove($F_i$)
12         $F_i$.setRank($rank$)
13     end foreach
14     $\mathbb{F} \leftarrow$ CrowdingDistanceAssignment($\mathbb{F}$)
15     $rank \leftarrow rank + 1$
16     $\mathbb{P}$.Add($\mathbb{F}$)
17 end while
18 return $\mathbb{P}$;
```

Algorithm 3: Pseudocode of SELECTPARENTSBYRANKANDDISTANCE

```
Input: Population $\mathbb{P}$, Tournament Size $arity$
Output: Parent Population Parents
1  for $si \leftarrow 1 : arity$ do
2      $\mathbb{P}_{win} \leftarrow$ getRandomSoln($\mathbb{P}$)
3      for $i \leftarrow 1$ : Size($arity$) do
4          $\mathbb{P}_{cand} \leftarrow$ getRandomSoln($\mathbb{P}$)
5          $flag \leftarrow$ CompareRankDist($\mathbb{P}_{win}$, $\mathbb{P}_{cand}$)
6          if $flag > 0$ then
7              $\mathbb{P}_{win} \leftarrow \mathbb{P}_{cand}$
8          end if
9      end for
10     $Parents_{si} \leftarrow \mathbb{P}_{win}$
11 end for
12 return Parents;
```

Algorithm 4: Pseudocode of CROWDINGDISTANCEASSIGNMENT

```
Input: Input Fronts $\mathbb{F}$
Output: Output Fronts $\mathbb{F}$
1  $n \leftarrow$ Size($\mathbb{F}$)
2  if $n \leq 3$ then
3      foreach $F_i \in \mathbb{F}$ do
4          $F_i$.setDistance($+\infty$)
5      end foreach
6  else
7      $nObj \leftarrow$ getNumOfObjs($F_1$)
8      foreach $F_i \in \mathbb{F}$ do
9          $F_i$.setDistance(0)
10     end foreach
11     foreach $Obj_i \in nObj$ do
12         $\mathbb{F}$.Sort($Obj_i$)
13         $Obj_{min} \leftarrow F_1$.getObj($Obj_i$)
14         $Obj_{max} \leftarrow F_n$.getObj($Obj_i$)
15         $F_1$.setDistance($+\infty$)
16         $F_n$.setDistance($+\infty$)
17         foreach $F_i \in \mathbb{F}$ do
18             $Dist_{cr} \leftarrow F_i$
19             $Dist_{Range} \leftarrow (Obj_{max} - Obj_{min})$
20             $Dist_{nbr} \leftarrow ( F_{i+1}$.getObj($Obj_i$) $- F_{i-1}$.getObj($Obj_i$))
21             $Dist_{cr} \leftarrow Dist_{cr} + ( Dist_{nbr} / Dist_{Range} )$
22             $F_i$.setDistance($Dist_{cr}$)
23         end foreach
24     end foreach
25 end if
26 return $\mathbb{F}$;
```

References

1. K. Ahmadian, A. Golestani, M. Analoui, and M.R. Jahed. Evolving ensemble of classifiers in low-dimensional spaces using multi-objective evolutionary approach. In *Computer and Information Science, 2007. ICIS 2007. 6th IEEE/ACIS International Conference on*, pages 217–222, July 2007.
2. Matti Aksela and Jorma Laaksonen. Using diversity of errors for selecting members of a committee classifier. *Pattern Recognition*, 39(4):608–623, April 2006.
3. Massimiliano Caramia and Paolo Dell'Olmo. *Multi-objective Management in Freight Logistics: Increasing Capacity, Service Level and Safety with Optimization Algorithms*, chapter Multi-objective Optimization, pages 11–36. Springer London, London, 2008.
4. Arjun Chandra and Xin Yao. Ensemble learning using multi-objective evolutionary algorithms. *Journal of Mathematical Modelling and Algorithms*, 5(4):417–445, 2006.
5. Chien-Yuan Chiu and B. Verma. Multi-objective evolutionary algorithm based optimization of neural network ensemble classifier. In *Signal Processing and Communication Systems (ICSPCS), 2014 8th International Conference on*, pages 1–5, Dec 2014.
6. Natalie Jane de Vries, Jamie Carlson, and Pablo Moscato. A data-driven approach to reverse engineering customer engagement models: Towards functional constructs. *PLoS ONE*, 9(7):e102768, 2014.
7. Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and TAMT Meyarivan. A fast and elitist multiobjective genetic algorithm: NSGA-II. *Evolutionary Computation, IEEE Transactions on*, 6(2):182–197, 2002.
8. E.M. Dos Santos, R. Sabourin, and P. Maupin. Single and Multi-Objective Genetic Algorithms for the Selection of Ensemble of Classifiers. *The 2006 IEEE International Joint Conference on Neural Network Proceedings*, pages 3070–3077, 2006.
9. Eulanda M. Dos Santos, Robert Sabourin, and Patrick Maupin. Pareto analysis for the selection of classifier ensembles. In *Proceedings of the 10th Annual Conference on Genetic and Evolutionary Computation*, GECCO '08, pages 681–688, New York, NY, USA, 2008. ACM.
10. R. Dutt and A.K. Madan. Predicting biological activity: Computational approach using novel distance based molecular descriptors. *Computers in Biology and Medicine*, 42(10):1026–1041, 2012.
11. Shenkai Gu, Ran Cheng, and Yaochu Jin. Multi-objective ensemble generation. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 5(5):234–245, 2015.
12. David Hadka. MOEA Framework: A Free and Open Source Java Framework for Multiobjective Optimization, 2014.
13. Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H Witten. The WEKA Data Mining Software: An Update. *SIGKDD Explorations Newsletter*, 11(1):10–18, 2009.
14. M. N. Haque, M. N. Noman, R. Berretta, and P. Moscato. Optimising weights for heterogeneous ensemble of classifiers with differential evolution. In *2016 IEEE Congress on Evolutionary Computation (CEC)*, pages 233–240, July 2016.
15. Mohammad Nazmul Haque, Nasimul Noman, Regina Berretta, and Pablo Moscato. Heterogeneous ensemble combination search using genetic algorithm for class imbalanced data classification. *PLoS ONE*, 11(1):e0146116, 01, 2016.
16. Yaochu Jin and B. Sendhoff. Pareto-based multiobjective machine learning: An overview and case studies. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 38(3):397–415, May 2008.
17. Giuseppe Jurman, Samantha Riccadonna, and Cesare Furlanello. A comparison of MCC and CEN error measures in multi-class prediction. *PLoS ONE*, 7(8):e41882, 08, 2012.
18. Gulshan Kumar and Krishan Kumar. The Use of Multi-Objective Genetic Algorithm Based Approach to Create Ensemble of ANN for Intrusion Detection. *International Journal of Intelligence Science*, 2(October):115–127, 2012.

19. Ludmila I. Kuncheva and Christopher J. Whitaker. Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy. *Machine Learning*, 51(2):181–207, 2003.
20. Charles X Ling, Jin Huang, and Harry Zhang. AUC: a statistically consistent and more discriminating measure than accuracy. In *IJCAI*, volume 3, pages 519–524, 2003.
21. Seema Mane, Shilpa Sonawani, and Sachin Sakhare. Hybrid multi-objective optimization approach for neural network classification using local search. In *Innovations in Computer Science and Engineering*, pages 171–179. Springer, 2016.
22. Anabel Martínez-Vargas, Josué Domínguez-Guerrero, Ángel G Andrade, Roberto Sepúlveda, and Oscar Montiel-Ross. Application of NSGA-II algorithm to the spectrum assignment problem in spectrum sharing networks. *Applied Soft Computing*, 39:188–198, 2016.
23. Tien Thanh Nguyen, A.W.-C. Liew, Xuan Cuong Pham, and Mai Phuong Nguyen. Optimization of ensemble classifier system based on multiple objectives genetic algorithm. In *Machine Learning and Cybernetics (ICMLC), 2014 International Conference on*, volume 1, pages 46–51, July 2014.
24. Ruba Obiedat, Mouhammd Alkasassbeh, Hossam Faris, and Osama Harfoushi. Customer churn prediction using a hybrid genetic programming approach. *Scientific Research and Essays*, 8(27):1289–1295, 2013.
25. A. Rahman and B. Verma. Cluster oriented ensemble classifiers using multi-objective evolutionary algorithm. In *Neural Networks (IJCNN), The 2013 International Joint Conference on*, pages 1–6, Aug 2013.
26. A. Santana, R.G.F. Soares, A.M.P. Canuto, and Marcilio C P de Souto. A dynamic classifier selection method to build ensembles using accuracy and diversity. In *Neural Networks, 2006. SBRN '06. Ninth Brazilian Symposium on*, pages 36–41, Oct 2006.
27. L Shi, G Campbell, WD Jones, F Campagne, S Walker, Z Su, et al. The MAQC-II project: A comprehensive study of common practices for the development and validation of microarray-based predictive models. *Nature biotechnology*, 2010.
28. Nidamarthi Srinivas and Kalyanmoy Deb. Multiobjective optimization using nondominated sorting in genetic algorithms. *Evolutionary computation*, 2(3):221–248, 1994.
29. Stephen V Stehman. Selecting and interpreting measures of thematic classification accuracy. *Remote sensing of Environment*, 62(1):77–89, 1997.
30. Fatemeh Vafaee. Using multi-objective optimization to identify dynamical network biomarkers as early-warning signals of complex diseases. *Scientific Reports*, 6:22023, 2016.
31. Zhi-Hua Zhou and Nan Li. Multi-information ensemble diversity. In *Multiple Classifier Systems: 9th International Workshop, MCS 2010, Cairo, Egypt, April 7–9, 2010. Proceedings*, pages 134–144. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010.
32. S Zickenrott, VE Angarica, BB Upadhyaya, and A Del Sol. Prediction of disease–gene–drug relationships following a differential network analysis. *Cell Death & Disease*, 7(1):e2040, 2016.

Chapter 21
Hotel Classification Using Meta-Analytics: A Case Study with Cohesive Clustering

Buyang Cao, Cesar Rego, and Fred Glover

Abstract We present a new clustering algorithm for handling complexities encountered in analysing data sets of hotel ratings and analyse its performance in a clustering case study. In the setting we address, business constraints and coordinates (among other individual attributes of objects) are unknown and only distances between objects are available to the clustering algorithm, a situation that arises in a wide range of clustering applications. Our algorithm constitutes an application of meta-analytics, in which we tailor a metaheuristic procedure to address a challenging problem at the intersection of predictive and prescriptive analytics. Our work builds on and extends the ideas of our clustering algorithm introduced in previous work which employs the Tabu Search metaheuristic to assure clusters exhibit a property we call cohesiveness. The special characteristics of the present hotel classification problem are handled by integrating our previous method with a new form of hierarchical clustering. Our computational analysis discloses that our algorithm obtains clusters that exhibit greater cohesiveness than those produced by the classical K-means method.

Keywords Tabu search · Predicted overall ratings · Simultaneous regression · Restrictive hierarchical clustering

B. Cao
China Intelligent Urbanization Co-Creation Center for High Density Region, School of Software Engineering, Tongji University, Shanghai, China
e-mail: caobuyang@tongji.edu.cn

C. Rego
School of Business Administration, University of Mississippi, Oxford, MS, USA
e-mail: crego@bus.olemiss.edu

F. Glover (✉)
Leeds School of Business and the College of Engineering & Applied Science, University of Colorado, Boulder, CO, USA
e-mail: glover@colorado.edu

P. Moscato, N. J. de Vries (eds.), *Business and Consumer Analytics: New Ideas*,
https://doi.org/10.1007/978-3-030-06222-4_21

21.1 Introduction

Clustering data is the optimization process of organizing objects into groups (or clusters) so that each cluster contains only objects that may be considered similar to each other as determined by specified criteria or an objective function. Clustering problems arise in a variety of domains and are prevalent in applications arising in the fields of engineering, science, and business. The literature on data clustering is extensive and surveys are frequently written to address new applications, data types, techniques, and algorithms. A comprehensive review of clustering methods is presented in [15]. In this study we examine the challenge of clustering hotels' data sets for quality assessment using customer feedback. In order to put our work in perspective we begin by briefly reviewing literature relevant to our purpose.

Nowadays as data-driven decision-making gets more advanced, clustering algorithms play a crucial role in analytical and decision-making. A useful reference highlighting this role is provided in the work of [12], who present a survey on clustering approaches in data mining. The authors additionally develop criteria to evaluate the quality of clusters and suggest procedures for the selection of solution methodologies.

In logistics applications, companies often undertake to create service areas so that resources can be utilized more efficiently and customers are provided with better services. In this setting [4] presents an algorithm based on Thiessen-polygons that proves highly effective in generating balanced and connected clusters. By the same token, customer segmentation is crucial for companies to gain deeper understanding of their customers' preferences and needs so that their products, services, and marketing efforts can be allocated more efficiently. A worthy example of such work is the study of [16] who examines clustering applications of customer segmentation for the case of a publishing company.

Another common application arises in the situation where organizations seek to cluster different employees into "employee pools" whose members share significant similarities in terms of their skill sets and working experiences. April et al. [2] discuss such applications where jobs are required to be grouped into common categories so that each category can be treated as a unified job classification.

Classifying customers according to their buying habits or finding communities in social networks are likewise key examples of big data applications requiring efficient handling of clustering problems. In this vein Wu et al. [23] propose a cluster-based methodology for solving "cold start" problems involving collaborative recommendation more effectively. An algorithm similar to K-means is used to cluster customers based upon their properties so that suitable products/services can be recommended to customers in each cluster. Mohebi et al. [17] extend this focus by reviewing clustering algorithms utilizing parallel approaches based on MapReduce architecture for big data and present a classification for identifying suitable algorithms and implementations.

In big data applications the problem of cleaning data and extracting relevant components from raw data sets has also received attention in recent literature. Real-world data sets inevitably contain some noise and the dimension (numbers of objects and attributes) encountered in these settings is often too large to be handled efficiently by traditional clustering algorithms. To address these complexities Cohen et al. [6] propose a methodology for K-means dimensionality reduction that has proved successful in certain business applications by cutting the computational time while keeping a desired precision. Strehl and Ghosh [20] introduce a similarity relationship measure defined on each pair in the data sample so that clusters are formed by reference to the similarity domain as opposed to explicitly using the original high-dimensional space. In the realm of time series analysis of noisy and high-dimensional data processes, Iorio et al. [14] present a parsimonious clustering method where P-spline smoothers are used to enable a clustering algorithm to operate on the estimated (spline) coefficients instead of directly on the raw data set.

In addition to the traditional (and widely popular) K-means algorithm, researchers have developed a variety of clustering approaches that are tailored to be effective for business use cases. A review of non-traditional clustering algorithms in this genre is provided by Brusco et al. [3], which surveys clustering algorithms ranging from model-based to metaheuristics-based methods. Reviews of swarm intelligence-based clustering algorithms can be found in [1] and [13]. Ozturk et al. [18] present an artificial bee colony metaheuristic algorithm to solve clustering problems. A comprehensive software package for the solution of bi-clustering problems found in biology is also developed by Foszner and Polański [7].

The algorithm presented in this paper follows in the tradition of the metaheuristic approaches by drawing upon a Tabu Search framework to implement a special form of a hierarchical clustering method, although as subsequently noted, we depart from previous metaheuristic efforts in several key respects introduced in [5] and extend them in multiple ways. In particular, our approach constitutes an application of meta-analytics where predictive analytics (in the form of a regression model) is used to correct possible noise in the data set and prescriptive analytics (carried out by a Tabu Search algorithm) is used to cluster the data.

The chapter is organized as follows: Section 21.2 describes the hotel rating clustering problem. In Sect. 21.3 we describe the corresponding clustering problem's objective function and the general solution procedure proposed by Cao et al. [5]. In order to solve the hotel rating clustering problem more effectively, we propose a restrictive hierarchical clustering algorithm in Sect. 21.4. Computational results documenting the effectiveness of our algorithm are presented in Sect. 21.5. Finally, we summarize our findings and present conclusions in Sect. 21.6.

21.2 Problem Background and Preliminary Considerations

TripAdvisor[1] is the most popular site visited by tourists worldwide, containing more than 200 million records of customer reviews covering hotels, scenic sites,

[1]https://www.tripadvisor.com.

restaurants, etc. in 149 countries. The customer reviews are created by real people and are valuable for customers, travel agencies, and hotel/restaurant owners. If such reviews additionally classified hotels into several groups based upon customers' ratings, the outcome would be very informative for potential customers whose preferences are influenced by such classifications. Here we define the category of a hotel according to its rating. Furthermore, such a clustering would be valuable to enable unrated hotels to be classified more effectively. This type of application motivates us to create clusters that exhibit a property we call *cohesiveness*, meaning that clusters are not only as compact as possible (by having their members tightly together) and exhibiting clear borders between them, but also as consistent as possible (by having their members belong to the same or closely related categories).

In order to protect against competition in some of these applications the attributes for objects to be clustered may not be available to analysts, and the only existing knowledge for clustering consists of the similarities or distances between objects. In these cases, determining meaningful cluster centroids is exceedingly difficult, yet such centroids are highly important to enable clustering algorithms to create clusters with desirable qualities. Cao et al. [5] presented a Tabu Search based algorithm to create cohesive clusters for such applications.

In this chapter we propose an algorithm based on this cohesive clustering approach but adapted to solve clustering or categorizing hotels according to customers' ratings more efficiently. The model is conceived from the standpoint of generating solutions that provide a baseline to classify unrated hotels or to provide references for potential customers. From this perspective, the goal is not to generate clusters (groups of hotels) in real time but rather to create a judicious collection of clusters with the aim of using them multiple times over an extended horizon as a means for future classification and reference.

21.3 Prescriptive Model and Procedure

We begin by describing the data set used in the present application and presenting the problem to be solved. We then provide a brief description of the Tabu Search framework, which lays the ground for the proposed algorithm presented next.

21.3.1 Problem Definition and Model

Consider a data set containing N_p objects (or elements), t_i $(i = 1, \ldots, N_p)$ each one defined by k attributes (or factors) represented by a valued $x(t_i) = (x_p(t_i), p = 1, \ldots, k)$. The problem is to build N_s clusters according to an objective function specified below. Hence, solutions are partitions of the set of N_p objects into N_s disjoint subsets. Let the collection of clusters $C(s)$ for $s = 1$ to N_s be a (proposed) solution to the problem. The similarity (or distance) of a pair (t_i, t_j) of elements of numerical attributes can be generally defined as

$$Score(t_i, t_j) = \sqrt{\sum_{p=1}^{k}(x_p(t_i) - x_p(t_j))^2}. \tag{21.1}$$

As noted in [5], $Score(t_i, t_j)$ is application specific; here, it is the Euclidean distance. As previously mentioned, we utilize the customer survey information posted on the TripAdvisor website to provide data for clustering hotels derived from customer ratings. In the present study, we use customer ratings of relevant aspects as attributes for categorizing a hotel. Traditionally, hotels are classified based upon overall customer ratings as explained subsequently. We use the same component ratings that create the overall ratings, but handle them in a different manner. The component ratings refer to the following seven variables or aspects:

- Value,
- Rooms,
- Location,
- Check in/front desk,
- Cleanliness,
- Service,
- Business service.

The rating for these aspects are discrete values ranging from 1 to 5, and are characterized for each hotel t_i by the vector $x(t_i) = (x_p(t_i), p = 1, \ldots, 7)$, where $x_p(t_i)$ is the rating value for the pth aspect mentioned above except the overall for hotel t_i. The customers' overall rating of a hotel is also part of the data set, though not used directly in our model. The derivation of the overall rating used in our model is obtained offline by an analytical process discussed in detail in Sect. 21.4. $Score(t_i, t_j)$ is then used to measure the proximity (or similarity) of hotels t_i and t_j.

Our clustering method is based on the following definitions and observations.

Let $n(s)$ be the number of elements in a cluster s. Then the number of pairs (t_i, t_j) or links in the cluster is determined by

$$NL(s) = \frac{n(s)(n(s) - 1)}{2}. \tag{21.2}$$

To evaluate the quality of the collection of clusters $C(s)$ for $s = 1$ to N_s we first compute the full value of a cluster as

$$FV(s) = \frac{1}{NL(s)} \sum_{\substack{t_i < t_j \\ t_i, t_j \in s}} Score(t_i, t_j). \tag{21.3}$$

We also make reference to the variance of a cluster s, which we define as

$$VAR(s) = \frac{1}{NL(s)} \sum_{\substack{t_i < t_j \\ t_i, t_j \in s}} (Score(t_i, t_j) - FV(s))^2. \tag{21.4}$$

Finally, our goal is to build clusters according to the following objective function:

$$ObjVal(C(s)) = min\left(\alpha \sum_{s\in C(s)} FV(s) + (1-\alpha) \sum_{s\in C(s)} VAR(s)\right), 0 \leq \alpha \leq 1, \tag{21.5}$$

where α is an adjustable nonnegative weight for the full value and variance of clusters, which allows for trade-offs between distributions and similarities of elements.

21.3.2 Tabu Search Framework

In common with [5], our algorithm makes use of Tabu Search. As a prelude to the description of our clustering procedure, we briefly describe the general Tabu Search framework.

Tabu Search (TS) is a metaheuristic algorithm which guides a lower-level heuristic search to escape the trap of local optimality, and has exhibited many successful applications for complex combinatorial problems [9]. Tabu Search forbids certain reverse search directions (temporarily classifying them as tabu) to avoid the trap of local optimality and to provide vigour to the search. The tabu restriction is embedded in the search procedure by using a special short term memory, which is accompanied by other longer term memories to provide strategies for intensifying and diversifying the search as noted below. The adaptive memory character of Tabu Search differentiates it from other metaheuristics such as genetic algorithms, simulated annealing, and swarm intelligence methods, although over time many strategies first proposed with Tabu Search have found their way into these other procedures. See, for example, Chapter 9 of [9].

A simple version of the TS algorithm proceeds as follows. A randomly or heuristically constructed solution is used as the starting point for succeeding iterations, in which the current solution is modified by moves to create a trajectory in neighbourhood space until a stopping criterion is satisfied. At each iteration, a *candidate list* of moves is generated by examining a subset of the full set of moves that lead to neighbouring solutions. Moves that lead back in the direction of recently visited solutions are designated tabu and made inaccessible. This mechanism is facilitated by introducing a *tabu tenure* which identifies the number of iterations that a solution attribute (such as the value assigned to a variable) should remain tabu. In the most rudimentary case, the tabu tenure has a fixed size, or varies randomly in size between fixed lower and upper bounds.

For example, in the case of a binary optimization problem, upon giving a binary variable x a new value (changing from 0 to 1 or from 1 to 0), the variable x is made tabu to prevent it from receiving its immediately preceding value for a duration of $TabuTenure$ iterations by setting $Tabu(x) = Iteration + TabuTenure$. Then, setting $Iteration = Iteration + 1$ at each iteration, x remains tabu as long as $Tabu(x) \geq Iteration$, and is no longer tabu when $Tabu(x) < Iteration$ (note:

if initially $Iteration = 1$ and $Tabu(x) = 0$ for all binary variables x, then all variables begin non-tabu). More generally, a record of tabu status keeps track of information such as $Tabu(x = v)$ to prevent a variable x from receiving a value v that was previously assigned to it for a specified tabu tenure. In our case, where moves consist of transferring an element from one cluster to another, we make the reverse move tabu by assigning a tabu tenure that prevents the element from returning to the cluster it came from.

Another feature of TS is the use of an *aspiration criterion* which overrules the tabu restriction if a certain condition is met. In our implementation we adopt the commonly used aspiration criterion of allowing a tabu move to be selected if it produces an objective value better than the best one obtained so far. Other types of aspiration rules also commonly used are assignment-specific, allowing a move to be made if the objective function created by assigning a variable a particular value (or assigning an element to a particular location or classification) is improved by reversing the assignment.

More advanced TS strategies make use of various kinds of frequency memory that identify the number of times a variable or element has received a particular assignment in high (or low) quality solutions of the past, as a means for encouraging or discouraging such assignments. A variant of such memory identifies the number of iterations that an assignment has remained active in solutions of a particular quality in the past. Similarly, more advanced TS methods keep records of high quality past solutions for the purpose of revisiting them, or incorporating subsets of their assignments in new solutions, in order to re-launch a solution trajectory. Such strategies, which reinforce solution features or revisit regions where good solutions were previously found, are called intensification strategies and are complemented with diversification strategies, which generate assignments that have never (or rarely) been made in the past, or that drive the search into regions that have never (or rarely) been visited in the past.[2] Particular strategies which were launched in association with Tabu Search for the goal of achieving useful intensification/diversification trade-offs have become extensively embodied in other metaheuristics to improve performance by finding better solutions or reducing solution time. Prevalent examples are the evolutionary scatter search and path relinking strategies (see, e.g., [8] and [10]). Integration of these approaches with other methods is described in [11] and [19].

21.3.3 Clustering Procedure

In the current study, we apply a simple form of tabu status based on a tabu tenure that is fixed for all moves. The basic moves and evaluations of the Tabu Search algorithm we employ in the present context are as follows:

[2]The terms "intensification" and "diversification", now widely used in many metaheuristic algorithms, were originally introduced in Tabu Search.

- **Assigning an element t_0 to a cluster:** If the target cluster is s_A, then the result is $\hat{s}_A = s_A \cup \{t_0\}$ and $n(\hat{s}_A) = n(s_A) + 1$.
- **Dropping an element t_0 from a cluster:** This move removes an element t_0 from its current cluster. Let s_0 be the cluster t_0 currently belongs to. After this move, we have $\hat{s}_0 = s_0 \backslash \{t_0\}$ and $n(\hat{s}_0) = n(s_0) - 1$.
- **Updating and evaluating:** The following updating and evaluating procedures serve to decide whether a move $m(t_0, s_A)$ is beneficial:

  $FV(\hat{s}_A)$—the new full value of cluster s_A after adding t_0;
  $VAR(\hat{s}_A)$—the new variance of cluster s_A after adding t_0;
  $FV(\hat{s}_0)$—the new full value of cluster s_0 after dropping t_0;
  $VAR(\hat{s}_0)$—the new variance of cluster s_0 after dropping t_0.

As a result of the foregoing operations, the move that drops t_0 from s_0 and adds it to s_A leads to the following change in the objective function value:

$$DropObjVal(t_0, s_0) = \alpha(FV(\hat{s}_0) - FV(s_0)) + (1-\alpha)(VAR(\hat{s}_0) - VAR(s_0)); \tag{21.6}$$

$$AddObjVal(t_0, s_A) = \alpha(FV(\hat{s}_A) - FV(s_A)) + (1-\alpha)(VAR(\hat{s}_A) - VAR(s_A)); \tag{21.7}$$

$$DelObjVal(t_0, s_A) = AddObjVal(t_0, s_A) - DropObjVal(t_0, s_0). \tag{21.8}$$

Under the present minimization objective, $DelObjVal < 0$ identifies an improving move, while $DelObjVal \geq 0$ identifies a non-improving move. Initially, when clusters are being created by assigning an element t_0 to a cluster s_A, the change in objective function value reduces to

$$DelObjVal(t_0, s_A) = AddObjVal(t_0, s_A). \tag{21.9}$$

The general Tabu Search clustering procedure proposed by Cao et al. [5] is as follows.

Tabu Search Clustering Procedure

Step 1: Build 2-Element Clusters

Perform the following until the desired number of clusters is obtained:

(a) For each t_0 that is not clustered, identify the 3 best (minimum) scoring matches:

$$(t_a, t_b, t_c) = \arg\min\{Score(t_0, t_a) + Score(t_0, t_b) + Score(t_0, t_c)\}. \tag{21.10}$$

(b) Select the pair t_u and t_v that satisfies

$$(t_u, t_v) = \underset{t_u, t_v \in \{t_a, t_b, t_c\}}{\arg\min} \{Score(t_0, t_u) + Score(t_0, t_v) + Score(t_u, t_v)\}, \tag{21.11}$$

where t_u and t_v is the possible pair combination of (t_a, t_b, t_c).

(c) Select t_1 to form a cluster with the first pair of elements (t_0, t_1) such that:

$$t_1 = \arg\min_{t_1 \in \{t_u, t_v\}} \{Score(t_0, t_1)\}. \tag{21.12}$$

Step 2: Build Initial Clusters
For the remaining unassigned elements, an element t_0 is assigned to the cluster s_A for which $AddObjVal(t_0, s_A)$ is minimum.

Step 3: Improvement Procedure
(a) Pick a move $m(t_0, s_A)$ from $CandidateList$ s.t. $DelObjVal(t_0, s_A)$ is the minimum objective value produced by any candidate move in $CandidateList$. (The candidate list creation strategy is described in Sect. 21.4.)
(b) Perform move $m(t_0, s_A)$ and update the clusters, Tabu status, and the objective function.
(c) If the objective function improves, set $loop = 0$. Otherwise, set $loop = loop + 1$.
(d) If $loop < tabuLoopLimit$ and stop criterion is not met, return to Step 3(a). Otherwise, if the stop criterion is not satisfied, perform Diversification Procedure (discussed in Sect. 21.4).
(e) If stop criterion is not met, return to Step 3(a). Otherwise output results and end the algorithm.

Alternative designs for the constructive phase in Steps 1 and 2 may be considered depending on the application. In Sect. 21.4 we present a constructive restrictive hierarchical clustering method that proved particularly advantageous in the present setting.

21.4 Supporting Algorithmic Components

Our algorithm makes use of a number of search strategies suitable to the application under consideration. These include a judicious selection of candidate moves for the local search and a diversification strategy aimed at exploring promising regions of diverse solutions. Provided that the input data for our algorithm does not readily support the type of hotel ratings available in the present data set, a preliminary data analysis and processing are necessary to generate adequate input data for the clustering algorithm.

21.4.1 Application Data Set and Algorithm Input Data

While building clusters, the definition of similarity between any pair of objects plays an important role. The purpose of the current study is to group (or cluster) hotels based on their quality as perceived by customers. As mentioned earlier the quality of a hotel in our data set is represented by seven individual aspect ratings and an overall rating, all given by customers on a scale 1–5 (worst to best).

Traditionally, hotels are clustered based upon their relative ratings typically given in the form of a similarity (or distance) matrix. On the other hand, ratings on individual attributes are not always disclosed to protect against competition. Therefore our clustering algorithm assumes no knowledge of individual aspect ratings. Since our algorithm by design creates clusters based on the input of a distance matrix, the computation of such input data is performed offline. Specially, we compute pairwise relationships between hotels obtained by the Euclidean distance derived from aspects' ratings. Therefore, the individual aspects' ratings are used to compute the Euclidean distance, defined by $Score(t_i, t_j)$, for any pair of hotels and given to the algorithm as the similarity measurement.

In order to validate the consistency of the aspects' ratings against the customers' overall rating for the corresponding hotels we developed a regression model (presented in Sect. 21.5) to predict individual overall ratings based upon the associated aspects' ratings. Our regression model reveals that the overall rating of a hotel can indeed be predicted according to the ratings for the seven individual aspects in the data set, excluding the customers' overall ratings. Therefore, we conclude that the pairwise relationship between hotels in our similarity matrix is consistent with the customers' overall ratings (as they derive from the same aspects ratings). We conjecture that a model that predicts overall ratings using all aspect ratings information is likely to reduce noise (e.g., due to rush or emotional factors) and so better explain the variance in overall customer satisfaction. Therefore, to evaluate the quality of clusters in terms of variance of overall ratings, we elected to use our *predicted overall ratings* in place of the original overall ratings.

21.4.2 Candidate List Construction

Since considering all eligible moves in a neighbourhood would entail excessive computational effort, a candidate list is employed that restricts attention to an appropriate subset of these moves. Also, some moves are likely not going to bring any improvement to the solution; therefore, anticipating potential non-improving moves is desirable.

In the present implementation, a move is included in the candidate list if:

$Score(t_0, t) < \delta \times avgDist$ (a predefined threshold for all existing elements t in a solution $C(s_A)$, as discussed below),

and

$$\frac{AddVal(t_0, s_A)}{n(s_A)} < \frac{1}{N_s} \sum_{s=1}^{N_s} \frac{AddVal(t_0, s)}{n(s)}, \tag{21.13}$$

where

$$AddVal(t_0, s) = \sum_{t \in C(s)} Score(t_0, t). \tag{21.14}$$

The threshold value is a parameter that can be adjusted as needed for specific application scenarios. In our study the threshold is defined by the average distance between all elements (denoted by $avgDist$ in the following discussions) weighted by a multiplier $0 < \delta \leq 1$, which can be adjusted to include more or less candidates. Our computational experiments demonstrate the efficacy of this candidate strategy to exclude moves that cannot contribute to improving the solution.

21.4.3 Diversification Mechanism

An effective search often requires a diversification strategy designed to obtain solutions that are significantly different from those yielded so far. Let N_p/N_s define the "baseline" number of elements in any cluster. A cluster will be called a *source cluster* if the number of elements exceeds N_p/N_s and will be called *target cluster* otherwise. An element t_0 is randomly selected from a source cluster and assigned to the target cluster s_A that possesses the minimal $DelObjVal(t_0, s_A)$ among all target clusters for element t_0. The process is repeated until there is no possible source cluster.

21.4.4 Restrictive Hierarchical Clustering Method

Our preliminary computational experiment disclosed that though the algorithm was able to create cohesive clusters based upon the input distances, the results did not achieve the goals we implicitly had in mind. We want each cluster to have a predicted overall rating that faithfully represents all elements (hotels) in that cluster. For instance, if the mean value of the predicted overall rating of a cluster is 3.5, then we should expect all elements in the cluster to have overall ratings relatively close to 3.5. In other words, we are seeking the minimum variance of predicted overall ratings, instead of the minimum variance of distances. Furthermore, clusters should ideally have a clear boundary that causes the mean values of overall ratings of different clusters to be far apart. However, the input data for the algorithm does not readily support an evaluation that permits the overall ratings to be differentiated in this fashion. In particular, our algorithm uses only distances defined by $Score(t_i, t_j)$ and as mentioned above does not acquire values for coordinates (individual aspect ratings) of the vectors. As a result, we do not have a convenient way to identify the centroid of a cluster. We discovered on further investigation that a variation on hierarchical clustering can compensate for this limitation and succeeds in creating relatively good initial solutions, where the mean values of predicted overall ratings are distant and clusters maintain clear boundaries. The elements of this strategy are embodied in the following *restrictive hierarchical clustering method*, which in our experiments is used in place of the constructive phase (Steps 1 and 2) of the Tabu Search procedure (presented in Sect. 21.3.3) for generating initial solutions.

Step 1: Create N_s Seed Points
For each cluster, we pick an initial seed point as follows (where N_s is the number of clusters to be built):

(a) Randomly pick the first seed point s_0, to create the starting seed point set $S = s_0$. Then repeat the following step until S contains N_s seed points.
(b) Pick a next seed point s_q, which satisfies

$$q = \arg\max_i \left\{ \sum_{s\in S} Score(t_i, s) \text{ and } t_i \text{ is qualified} \right\}, \tag{21.15}$$

where t_i is called qualified if $Score(t_i, s) >$ predefined *buffer distance*. In our case, the predefined buffer distance is $ratio \times avgDist$, where $ratio$ is a parameter that can be adjusted. The larger the $ratio$, the farther apart the seed points will be.

Step 2: Forming Clusters
For the rest of the elements (outside of S) clusters are built by a method similar to the hierarchical clustering method. Initially each seed point and each remaining point constitute a single-element cluster. Define the average distance $avgDist(r, s)$ between two clusters r and s to be average of the distances between their elements, i.e., two clusters r and s that yield a minimum value for $avgDist(r, s)$ are selected to be merged, provided that they do not both contain seed points. (By implication, each merged cluster will contain at most one seed point.) The process continues until all clusters contain a seed point, hence yielding precisely N_s clusters. We may express the process as follows. At any step of the merging procedure, let N_s-set denote the N_s clusters that contain seed points. Then at each step until all clusters belong to N_s-set, the following two merging options are feasible:

(a) Two clusters that are not in N_s-set;
(b) One cluster is in N_s-set and the other is not.

Step 3: Improvement
The initial solution is passed to the Improvement Procedure, described in Step 3 of the Tabu Search Procedure.

21.5 Predictive Model and Computational Experiments

An extensive computational analysis has been conducted to gauge the effectiveness of our clustering approach on the assessment and classification of hotels based upon quality scores as perceived by guests. We refer the reader to [21, 22] for related studies on the same data sets from TripAdvisor[3] used in our study. For comparison with pertinent approaches in the literature our algorithm is applied to the data sets without supervision. The C# programming language was used to code the algorithm and computational tests were carried out on a desktop computer with a 2.7 GHz Intel Core CPU, 8 GB of RAM using Windows 7 Enterprise operating system.

21.5.1 Predictive Model

As stated earlier our clustering algorithm takes a similarity (distance) matrix as input to cluster a data set. Our goal is to create a distance matrix that most accurately reflects the differences between hotels with respect to the relevant aspects and associated overall quality. As noted in Sect. 21.1, customer reviews for each hotel are given for the following aspect ratings: value, rooms, location, check in/front desk, cleanliness, service, business service, overall. Ratings range from 1 to 5. Before we run our clustering algorithm, we conduct the data analysis and create a predictive model to derive the overall rating for a hotel from the corresponding aspects' ratings. In this case we can conclude that it is rational to validate clusters upon the predictive overall ratings obtained from the individual aspect ratings as opposed to the given overall ratings.

In our study we apply the clustering algorithm to cluster the hotels disregarding their overall rating values. For a hotel i the aggregate rating for an individual aspect j (excluding overall rating) is given by

$$r_{ij} = \frac{1}{n_i} \sum_{w=1}^{n_i} r_{ijw}, \tag{21.16}$$

where

r_{ij}—the rating for aspect j of a hotel i,
r_{ijw}—the rating for aspect j of a hotel i posted by customer w,
n_i—the total number of valid reviewers for a hotel i.

We may reasonably assume that data may suffer from bias or random variations in customer reviews due to factors such as rush or emotional reasons. To address

[3] All data sets can be downloaded from http://times.cs.uiuc.edu/~wang296/Data.

this possibility a regression model relating overall rating (dependent variable) to all other aspect rating values (independent variables) is obtained by the following procedures, where SPSS was employed to conduct the data analysis. We observed the relationship between individual independent variables and the overall ratings by plotting the following figures, where the y-axis represents the overall rating, while the x-axis indicates the individual aspects.

From Fig. 21.1 we are able to identify the linear trend between individual aspects and the overall rating. The regression model is then obtained as follows:

$$\begin{aligned} Overall = & -0.5575 + 0.3115\ Values + 0.2505\ Rooms + 0.07334\ Location \\ & + 0.2136\ Cleanliness + 0.0283\ CheckInFrontDesk + 0.2214\ Service \\ & + 0.0128\ BusinessService. \end{aligned} \tag{21.17}$$

Using a representative portion of the data for prediction, we have the result shown in Fig. 21.2, where dashed lines depict the real overall ratings and solid lines the predicted ones. Table 21.1 summarizes the model, showing how much the variation in overall rating can be explained by the aspect ratings (overall fit) and their relative contribution to the total variance explained. Since our model uses multiple predictors, the Adjusted R^2 value is particularly relevant. We used *simultaneous regression* where all predictors (aspect ratings) were tested at once. All seven aspect ratings were included in the model as given by degrees of freedom (df1). We see from our Adjusted R^2 of 0.9347 that aspect ratings account for 93.47% of the total variability of the overall rating variable. Since values of R^2 and Adjusted R^2 are about the same, we can conclude that changes in the overall rating are impacted mostly by the aspect ratings. A regression model containing no predictors is known as an intercept-only model. The overall significance of the our model is given by the F-test. Since p-value (Sig. F Change in the table) is less than 0.01, we can reject the null hypothesis that the fit of the intercept-only model and our model is equal and conclude that the aspect rating variables in combination significantly predict the overall rating for a hotel.

21.5.2 Computational Experiments

From the results in Table 21.1 listed in Sect. 21.5.1, we conclude that overall rating values are reasonably correlated with the aspects and hence can be predicted from the individual aspect rating values by means of the regression model presented above. Therefore, we conclude that the individual aspect rating values accurately reflect the overall quality of the corresponding hotels and therefore can be used to cluster the hotels. In order to solve the corresponding clustering problem, Euclidean distances are employed to measure the similarity of any pair of hotels. For every two hotels i and j, we compute the distance $Score(i, j)$ by

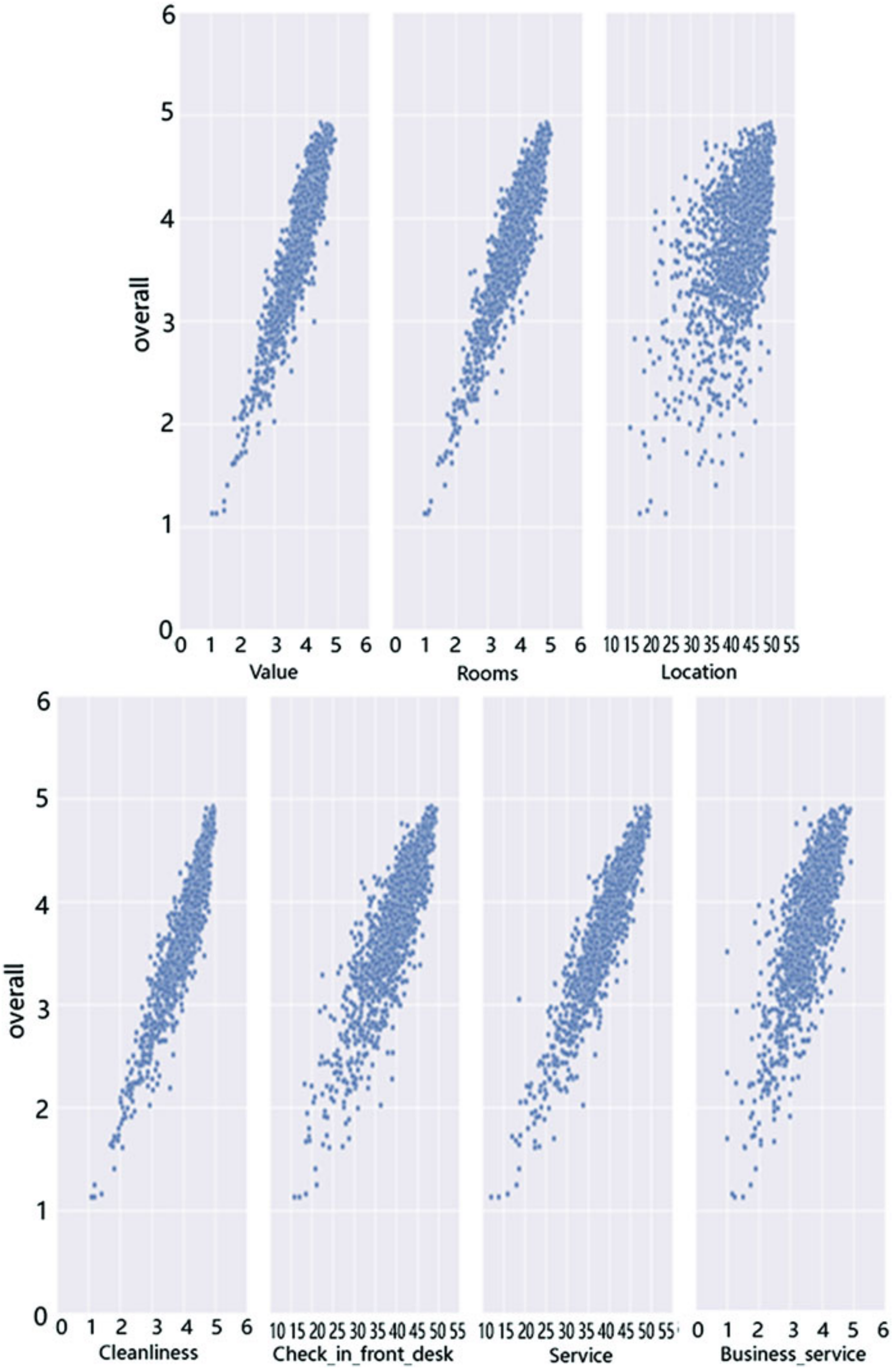


Fig. 21.1 The relationships between aspects and overall ratings for each hotel

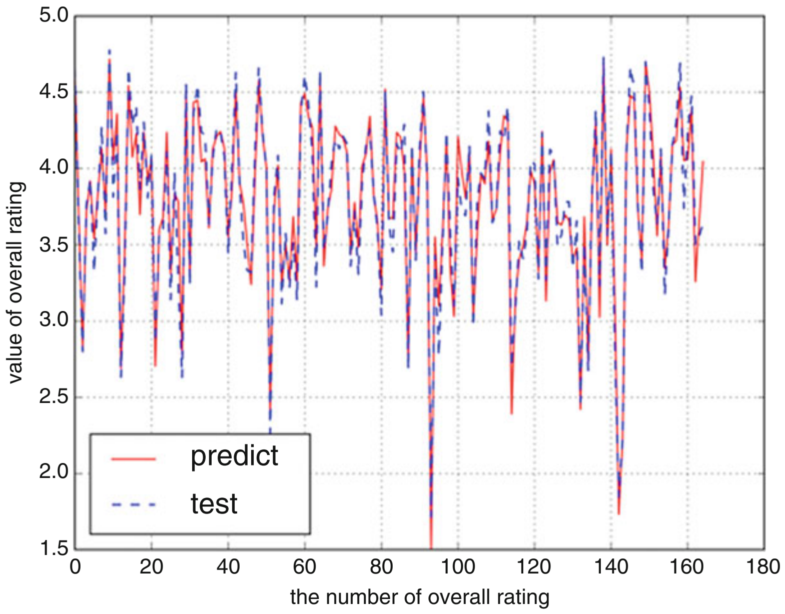


Fig. 21.2 Prediction against real overall rating

Table 21.1 Model summary

				Change statistics				
R	R^2	Adjusted R^2	Std error estimate	R^2 change	F change	df1	df2	Sig. F change
0.967	0.935	0.9347	0.16	0.935	2697.539	7	1311	0

$$Score(i, j) = \sqrt{\sum_{p=1}^{k} (r_{ip} - r_{jp})^2}, \tag{21.18}$$

where r_{ip} is the rating value for the pth aspect of hotel i. Though we understand that the aspects may have different impacts on the final (overall) rating, we treat all aspects equally; hence, the weighted Euclidean distance is not applied in the clustering procedure. Since all aspects are rated on the same scale 1–5, normalization is not required. However, in other cases normalization might be a necessary process in the data pre-processing.

In Sect. 21.3, we presented the objective function we are using, incorporating distances and variances to compute the objective, judge the quality of solution, and guide the solution procedure. As pointed out in [5], breaking ties between nearly equal move evaluations can be achieved by accounting for the variance, which in turn helps the search to pinpoint the truly relevant or most promising moves.

Based on the preceding discussion we conclude that a cluster should contain hotels whose overall ratings are as close as possible. In order to evaluate the algorithm more thoroughly, we sample the data set to form various testing data sets. In addition to seeking a clustering solution s that minimizes intra-cluster Euclidean distances, we also attempt to minimize the sum of the variance $SV(s)$ of each cluster s in $C(s)$ as follows:

$$SV(s) = \frac{1}{n(s)} \sum_{i \in C(s)} (r_i - \bar{r})^2, \tag{21.19}$$

where:

SC—the index set for the clusters created, $SC = \{1, \ldots, k\}$;
r_i—the predicted overall rating of a hotel i that is grouped into cluster s;
$\bar{r}$—the average predicted overall rating for all hotels included in cluster s;
$n(s)$—the number of hotels contained in cluster s.

A posterior procedure is created to compute $SV(s)$ for each cluster in the collection $C(s)$, $s \in SC$. The original input data set is used to calculate the predicted overall ratings, $\bar{r}$, and variance $SV(s)$ for each cluster in $C(s)$.

Tables 21.2 and 21.3 report comparative results for our cohesive clustering algorithm and for the K-means method, in which K represents the number of

Table 21.2 Computational results (K-means)

Problem			Number of hotels		
No.	Size	k	In clusters	$\bar{r}$ in clusters	$SV(s)$ in clusters
1	50	3	4, 10, 36	(1.682, 2.851, 3.821)	(0.0206, 0.0637, 0.1033)
		4	4, 8, 23, 15	(1.682, 2.777, 3.556, 4.14)	(0.0206, 0.0516, 0.0392, 0.038)
2	100	3	15, 45, 40	(2.394, 3.533, 4.161)	(0.2393, 0.0634, 0.032)
		4	5, 16, 39, 40	(1.749, 2.859, 3.601, 4.161)	(0.0343, 0.0548, 0.0377, 0.032)
3	150	3	32, 59, 59	(2.613, 3.55, 4.168)	(0.191, 0.0504, 0.0331)
		4	6, 28, 57, 59	(1.817, 2.814, 3.569, 4.168)	(0.0523, 0.0442, 0.0427, 0.0331)
4	300	3	42, 114, 144	(2.647, 3.607, 4.227)	(0.1874, 0.0452, 0.0412)
		4	12, 47, 105, 136	(2.054, 3.014, 3.697, 4.24)	(0.1106, 0.0453, 0.031, 0.0401)
5	500	3	65, 201, 234	(2.712, 3.616, 4.221)	(0.1517, 0.0432, 0.0373)
		4	44, 114, 181, 161	(2.554, 3.374, 3.867, 4.311)	(0.1447, 0.0432, 0.0294, 0.0251)
6	800	3	110, 310, 380	(2.67, 3.596, 4.216)	(0.1867, 0.0472, 0.0404)
		4	59, 173, 283, 285	(2.394, 3.27, 3.822, 4.286)	(0.1704, 0.0514, 0.0327, 0.0312)
7	1000	3	132, 377, 491	(2.664, 3.6, 4.21)	(0.1739, 0.051, 0.0417)
		4	97, 231, 347, 325	(2.519, 3.38, 3.894, 4.32)	(0.155, 0.0413, 0.0282, 0.0278)
8	1200	3	163, 439, 598	(2.665, 3.617, 4.237)	(0.1826, 0.0523, 0.0425)
		4	94, 267, 423, 416	(2.425, 3.309, 3.885, 4.33)	(0.168, 0.0556, 0.0304, 0.0288)
9	1500	3	192, 549, 758	(2.667, 3.634, 4.276)	(0.2, 0.0556, 0.0456)
		4	103, 332, 522, 553	(2.38, 3.3, 3.9, 4.36)	(0.1828, 0.0547, 0.0305, 0.0325)

Table 21.3 Computational results (our algorithm)

Problem			Number of hotels		
No.	$Size$	k	In clusters	$\bar{r}$ in clusters	$SV(s)$ in clusters
1	50	3	4, 10, 36	(1.682, 2.851, 3.821)	(0.0206, 0.0637, 0.1033)
		4	5, 4, 29, 12	(1.795, 2.763, 3.525, 4.213)	(0.0678, 0.0025, 0.0746, 0.0202)
2	100	3	6, 58, 36	(1.832, 3.439, 4.194)	(0.0632, 0.1394, 0.0248)
		4	5, 23, 25, 47	(1.749, 2.859, 3.601, 4.161)	(0.0343, 0.0747, 0.0083, 0.0395)
3	150	3	37, 55, 58	(2.681, 3.591, 4.177)	(0.1955, 0.0337, 0.0297)
		4	4, 33, 55, 58	(1.682, 2.802, 3.591, 4.177)	(0.0206, 0.0809, 0.0337, 0.0297)
4	300	3	9, 154, 137	(1.905, 3.467, 4.245)	(0.0567, 0.1435, 0.0363)
		4	2, 7, 154, 137	(1.545, 2.007, 3.457, 4.245)	(0.0009, 0.0252, 0.1435, 0.0364)
5	500	3	73, 156, 271	(2.764, 3.568, 4.179)	(0.1575, 0.0226, 0.0442)
		4	2, 71, 156, 271	(1.545, 2.8, 3.459, 4.179)	(0. 0009, 0.119, 0.0227, 0.0442)
6	800	3	29, 343, 428	(2.065, 3.383, 4.182)	(0.1255, 0.1081, 0.0452)
		4	15, 14, 404, 367	(1.8, 2.345, 3.459, 4.232)	(0.0952, 0.0067, 0.1242, 0.0353)
7	1000	3	35, 122, 843	(2.09, 2.92, 3.973)	(0.1024, 0.1246, 0.0396)
		4	27, 243, 356, 374	(1.997, 3.133, 3.815, 4.297)	(0.089, 0.0937, 0.0253, 0.0277)
8	1200	3	24, 497, 679	(1.816, 3.338, 4.202)	(0.0596, 0.1484, 0.0471)
		4	33, 302, 423, 442	(1.949, 3.153, 3.849, 4.324)	(0.09123, 0.098, 0.0255, 0.0264)
9	1500	3	28, 605, 867	(1.78, 3.35, 4.23)	(0.095, 0.1515, 0.0518)
		4	3, 25, 605, 867	(2.38, 3.3, 3.9, 4.36)	(0.0009, 0.0418, 0.1515, 0.0518)

clusters to be built. Since K-means could not take the variance into account while building clusters, we set α to 0.9. The K-means algorithm used in the computational experiment is embedded in R, a free software environment for statistical computing and graphics. We call K-means as kmeans(data, k, 1000, 10) to run it with a maximum of 1000 iterations and 10 restarts. Although other clustering algorithms are available, we use K-means as the benchmark because of its extensive use and publicized success in handling a wide range of real problems successfully. The problem size indicates the number of hotels to be clustered. The means and variances of predicted overall ratings for all clusters in the data set are listed together for the sake of clarity. The tables also show the number of hotels in each resulting cluster.

The parameters for our algorithm are set as follows:

- Tabu list size ($TabuTenure$): 6
- The maximum number of iterations (stopping criterion): 1000
- The number of iterations without improvement before starting diversification ($tabuLoopLimit$): 50
- The buffer distance multiplier ($ratio$): 0.9
- The threshold multiplier for the candidate list (δ): 1
- The weight for full value versus variance in a cluster (α): 0.9

Our empirical analysis showed that our clustering algorithm is robust when using values around those reported above. Therefore, our parameter settings have been fixed for all problem instances, i.e., no parameter adjustment is carried out and there is no human intervention during the computational experiments.

The computational times of our algorithm on the above computational environment and data set range from less than 1 s to approximately 270 s, depending on the number of objects to be clustered. These times are generally higher than those required by K-means, though still very reasonable. Our empirical analysis showed that a major portion of the computational time is spent on creating initial solutions where the restrictive hierarchical clustering method comes into play, since this approach requires pairwise comparisons to determine how two clusters should be merged. However, our algorithm possesses a structure that can be efficiently explored by parallel computing, which affords a means to reduce computational times.

Graphs of Fig. 21.3 depict the comparisons of the (sum of) variances of predicted overall ratings obtained by K-means and by our algorithm for solutions containing three and four clusters. In the graphs the x-axis presents the sizes of the underlying problems and the y-axis indicates the sum of variances of predicted overall ratings.

As discussed earlier, in order to provide the baseline for classifying hotel categories consistent with their relative differences in aspect and overall ratings we are motivated to build clusters of low variance. As such, the results in the tables and associated diagrams show that with a few exceptions our algorithm produces better solutions than K-means. These results also disclose that our algorithm produces clusters depicting a clear boundary, as represented by the relatively large differences of mean values of predicted overall ratings. This result is beneficial for categorizing new data in general. In the present situation it also makes it easier for travel information providers and travellers to offer more accurate information and retrieve more helpful information regarding hotel categories judged on the basis of their overall ratings.

The results also disclose that our algorithm has a useful ability to detect outliers, i.e., hotels whose overall ratings significantly differ from the average overall rating of the underlying data set. For example, the first cluster in problems 2, 3, 5, 7, and 8, for the instances with four clusters, identify a small number of hotels whose overall rating is significantly lower than the average overall rating of the data set. The ability of our approach to identify these hotels provides a basis for modifying or supplementing overall ratings to provide more meaningful information for travellers and hotel operators.

21.6 Conclusions

We adopt a clustering framework that has proved effective in logistics and cloud computing applications to create a new clustering algorithm for application to categorize hotels. Drawing on preliminary findings from our predictive model, we base our prescriptive clustering model upon Euclidean distances and test the quality of the clusters produced by our algorithm, accounting for total distances between hotels in each cluster and their variance relative to their predicted overall rating. As in our previous applications we organize our approach so that it only uses values (distances, in this case) representing similarities between elements in the clustering decision process, which is essential in contexts where aspects' ratings are not available and makes our algorithm appropriate for applications where numerical attributes are absent. In order to keep this flexibility while achieving our objective we introduce a restrictive hierarchical clustering method that efficiently captures the centroid of a cluster when vector coordinates are not available.

Our clustering algorithm has demonstrated its effectiveness in creating cohesive clusters as shown in our computational tests. The computational experiments disclose the algorithm is both robust and extensible. In addition, our method does not require extensive parameter tuning and is suitable for running on a backend to perform the corresponding tasks with little human intervention.

Future research will focus on enhancements of the algorithm to handle additional problem considerations and the use of parallel processing and big data technologies to improve computational efficiency.

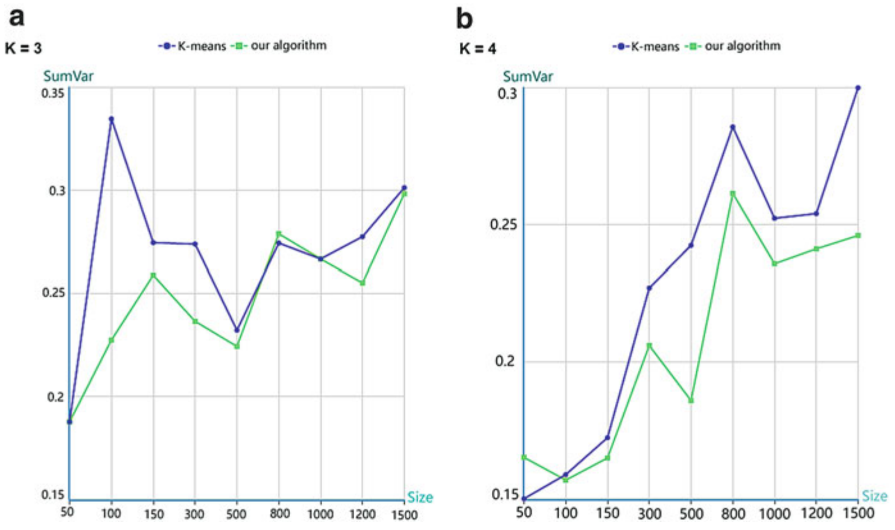


Fig. 21.3 The total variances obtained by both algorithms for (**a**) K = 3 and (**b**) K = 4

Acknowledgements We are indebted to a reviewer for insightful critical observations and suggestions that have helped to improve our chapter. The authors would like to thank our student team including Zheng Xu and Fang Yu for their efforts in implementing the algorithm, data preparation, and data analysis. This work was partially supported by the China Intelligent Urbanization Co-Creation Center [grant number CIUC20150011] and was also supported in part by the Key Laboratory of International Education Cooperation of Guangdong University of Technology.

References

1. Ajith Abraham, Swagatam Das, and Sandip Roy. Swarm intelligence algorithms for data clustering. In *Soft computing for knowledge discovery and data mining*, pages 279–313. Springer, 2008.
2. Jay April, Marco Better, Fred Glover, James P Kelly, and Gary Kochenberger. Strategic workforce optimization: Ensuring workforce readiness with OptForce^{TM}. Report P-20, OptTek Systems, Inc., 2014.
3. Michael J Brusco, Douglas Steinley, J Dennis Cradit, and Renu Singh. Emergent clustering methods for empirical OM research. *Journal of Operations Management*, 30(6):454–466, 2012.
4. Buyang Cao and Fred Glover. Creating balanced and connected clusters to improve service delivery routes in logistics planning. *Journal of Systems Science and Systems Engineering*, 19 (4):453–480, 2010.
5. Buyang Cao, Fred Glover, and Cesar Rego. A tabu search algorithm for cohesive clustering problems. *Journal of Heuristics*, 21(4):457–477, 2015.
6. Michael B Cohen, Sam Elder, Cameron Musco, Christopher Musco, and Madalina Persu. Dimensionality reduction for k-means clustering and low rank approximation. In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing*, pages 163–172. ACM, 2015.
7. Pawel Foszner and Andrzej Polański. Aspect analyzer distributed system for bi-clustering analysis. In *Man–Machine Interactions 4*, pages 411–420. Springer, 2016.
8. Fred Glover. A template for scatter search and path relinking. *Lecture notes in computer science*, 1363:13–54, 1998.
9. Fred Glover and Manuel Laguna. *Tabu Search*. Springer-Verlag, 1997.
10. Fred Glover, Manuel Laguna, and Rafael Martí. Fundamentals of scatter search and path relinking. *Control and cybernetics*, 29(3):653–684, 2000.
11. Fred Glover, Manuel Laguna, and Rafael Martí. New ideas and applications of scatter search and path relinking. In *New optimization techniques in engineering*, pages 367–383. Springer, 2004.
12. Johannes Grabmeier and Andreas Rudolph. Techniques of cluster algorithms in data mining. *Data Mining and Knowledge Discovery*, 6(4):303–360, 2002.
13. Tülin İnkaya, Sinan Kayalıgil, and Nur Evin Özdemirel. Swarm intelligence-based clustering algorithms: A survey. In *Unsupervised Learning Algorithms*, pages 303–341. Springer, 2016.
14. Carmela Iorio, Gianluca Frasso, Antonio D´Ambrosio, and Roberta Siciliano. Parsimonious time series clustering using p-splines. *Expert Systems with Applications*, 52:26–38, 2016.
15. Anil K Jain, M Narasimha Murty, and Patrick J Flynn. Data clustering: a review. *ACM computing surveys (CSUR)*, 31(3):264–323, 1999.
16. Gordon S Linoff and Michael JA Berry. *Data mining techniques: for marketing, sales, and customer relationship management*. John Wiley & Sons, 2011.
17. Amin Mohebi, Saeed Aghabozorgi, Teh Ying Wah, Tutut Herawan, and Ramin Yahyapour. Iterative big data clustering algorithms: a review. *Software: Practice and Experience*, 46(1): 107–129, 2016.
18. Celal Ozturk, Emrah Hancer, and Dervis Karaboga. Dynamic clustering with improved binary artificial bee colony algorithm. *Applied Soft Computing*, 28:69–80, 2015.
19. Mauricio GC Resende, Celso C Ribeiro, Fred Glover, and Rafael Martí. Scatter search and path-relinking: Fundamentals, advances, and applications. In *Handbook of metaheuristics*, pages 87–107. Springer, 2010.
20. Alexander Strehl and Joydeep Ghosh. Relationship-based clustering and visualization for high-dimensional data mining. *INFORMS Journal on Computing*, 15(2):208–230, 2003.
21. Hongning Wang, Yue Lu, and Chengxiang Zhai. Latent aspect rating analysis on review text data: a rating regression approach. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 783–792. ACM, 2010.

22. Hongning Wang, Yue Lu, and ChengXiang Zhai. Latent aspect rating analysis without aspect keyword supervision. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 618–626. ACM, 2011.
23. Hongchen Wu, Xinjun Wang, Zhaohui Peng, and Qingzhong Li. Div-clustering: exploring active users for social collaborative recommendation. *Journal of Network and Computer Applications*, 36(6):1642–1650, 2013.

Part VI
Data Science Applications in Travel and Fashion Analytics

Chapter 22
Fuzzy Clustering in Travel and Tourism Analytics

Pierpaolo D'Urso, Marta Disegna, and Riccardo Massari

Abstract Cluster analysis is one of the most common techniques used in tourism and marketing research to find homogeneous groups of units (i.e. tourists or consumers) according to a set of segmentation variables. Since it may be too strict to assume that each unit belongs to one cluster 100%, fuzzy clustering algorithms have been suggested in the market segmentation literature, which allow the same unit to be assigned to more than one cluster with a membership degree. Often, individual judgements, like emotions, motivations, or satisfactions, are captured making use of qualitative scales, such as Likert-type scales. Information collected using such scales are vague and uncertain mainly because respondents have to convert their judgements on a linguistic expression, often expressed in numerical values, the meaning of which can be different per respondent. To reduce this source of vagueness, ordinal variables can be transformed into fuzzy variables before the adoption of a clustering algorithm. This operation requires the modification of traditional fuzzy clustering algorithms in order to cope with the fuzzy nature of the segmentation variables. Two kinds of fuzzy clustering algorithms for fuzzy data are theoretically presented in this chapter, together with two empirical case studies.

Keywords Fuzzy clustering · Fuzzy market segmentation · Tourism research

P. D'Urso (✉) · R. Massari
Department of Social Sciences and Economics, Sapienza University of Roma, Roma, Italy
e-mail: pierpaolo.durso@uniroma1.it; riccardo.massari@uniroma1.it

M. Disegna
Accounting, Finance & Economics Department, Faculty of Management, Bournemouth University, Bournemouth, UK
e-mail: disegnam@bournemouth.ac.uk

P. Moscato, N. J. de Vries (eds.), *Business and Consumer Analytics: New Ideas*,
https://doi.org/10.1007/978-3-030-06222-4_22

22.1 Why Fuzzy Clustering for a Tourism Market?

The Nobel Prize winner in literature, Doris Lessing, in 2007 affirmed that "things are not quite so simple always as black and white". The world is not black or white and human experiences are not merely good or bad. People are usually not totally satisfied or totally dissatisfied with a product, an experience, a destination, or anything else. There are a few totally satisfied and totally dissatisfied people (or, in general, there are a few cases where respondents answer in an unequivocally extreme way) but they are not as common. Emotions, consumer behaviours, motivations, and human beings in general are better described by complex processes. Therefore, why when we describe, i.e. segment, a market using a cluster analysis do we force units to belong only to one cluster? Why do we arbitrarily assign a unit to one cluster when it is exactly "in the middle" of two or more clusters?

As emphasized in the market segmentation literature [3, 4, 31, 32], it is not always reasonable to assume that one unit belongs to one cluster. Consumers whose profiles mainly match one cluster do not necessarily have to be assigned 100% to that cluster [3]. Consumers who can be assigned to more than one cluster should not be solely assigned to one of them in an arbitrary way [32]. In all these examples, if consumers are allocated to only one cluster, the representation of the market is not realistic it is just a selective picture of the reality [4].

For example, suppose we are interested in finding out the favourite holiday destination of a group of people. In order to achieve this, we ask tourists to answer the following question: "Do you prefer beaches or mountains for your holidays?" As a result, in this case it is reasonable to expect that people are grouped into two clusters labelled "mountain" and "sea", but it is also reasonable to expect that some people are uncertain of the choice of one of these two holiday destinations, or that they are equally attracted to both. Performing a traditional (crisp) clustering method each unit is completely assigned to one cluster, while performing a fuzzy clustering method the strict assignment of units to only one cluster is relaxed and each tourist is only partially assigned to each cluster. Figure 22.1 displays how the "mountain" and "sea" groups look like after the adoption respectively of a crisp and fuzzy clustering algorithm ("mountain" group is coloured in green, while "sea" group is in blue). As a result, in the crisp market tourists assume a plain colour, while in the fuzzy market tourists are characterized by spots that represent their degree of membership to each of the two clusters. Concluding, fuzzy clustering methods would allow to better identify those tourists who are attracted by both destinations: in some cases, those tourists would like to enjoy a holiday in a destination surrounded by majestic mountain landscapes, while on other occasions they prefer to be on a warm beach with a cold drink in their hands.

The reasons why fuzzy clustering algorithms are adequate techniques for market segmentation are not only those listed above. Their adoption is also motivated by philosophical, economic, managerial, and marketing reasons. In the late nineteenth and early twentieth centuries the modernism philosophical movement came to light, characterized by concepts such as "absolute" and "universal" realities. In the late

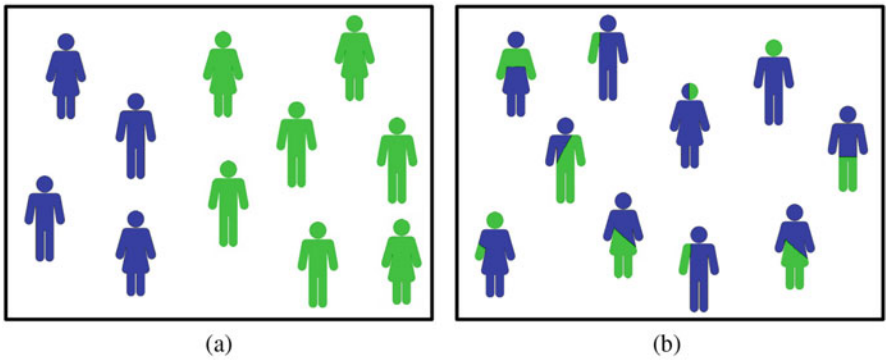


Fig. 22.1 Schematic representation of (**a**) crisp and (**b**) fuzzy markets

1960s and 1970s, some of the most eminent philosophers put into discussion these "modern" concepts, replacing them with concepts like de-realization, subjectivation, deconstruction, and hyper-reality. The postmodern era was born declaring the absence of an ultimate reality, an absolute truth, and, consequently, a universal perspective. In the same period, Zadeh presented his first work on fuzzy sets [40] as an answer to the emerging request of a new logic that is able to capture complex situations in which the traditional "true and false" dichotomy was not enough. In the late twentieth century, researchers in the marketing field started to investigate the characteristics of the new postmodern consumer and nowadays consumers' preferences, choices, and behaviours are studied under the postmodernism perspective (e.g. [21, 36]). In the early twenty-first century, the postmodern movement also reached researchers in the economic field. The rationality of a consumer's behaviour (in terms of preference relation associated with a demand function), at the basis of the revealed preference concept introduced by Samuelson [35], was put into discussion and re-formulated under an abstract framework. Due to the lack of information and human subjectivity, researchers have recognized the vague, i.e. fuzzy, nature of consumers' preferences, often not exact or not precise [23]. Therefore, a fuzzy conceptualization of both the consumer theory and the connected notions of preference and choice has been introduced (e.g. [14, 22]).

Finally, from a methodological point of view, fuzzy clustering algorithms have many advantages over more traditional (non-fuzzy) cluster algorithms [7, 27]. First, the fuzzy clustering methods are computationally more efficient because dramatic changes in the value of cluster membership are less likely to occur in estimation procedures [33]. Second, fuzzy clustering has been shown to be less affected by local optima problems [25]. Finally, the memberships for any given set of respondents indicate whether there is a second-best cluster almost as good as the best cluster, a result which traditional clustering methods cannot uncover [19].

In the following section, the inclusion of the fuzzy theory in each traditional step of a cluster analysis is described and discussed, while in Sect. 22.3 two case studies, in which fuzzy theory has been included in all steps, are illustrated.

22.2 Fuzzy Clustering

In academic marketing and tourist research, cluster analysis has been mainly conducted using categorical ordinal data [15, 16]. Categorical ordinal data can be classified into the following five different types depending on the presence/absence of thresholds, latent variables, and standardization process [29]:

1. the categorized metric (i.e. observable) variable with known thresholds, as, for instance, expenditure groups;
2. the categorized metric variable with unknown thresholds, for instance, expenditure groups like "low" and "high";
3. the categorized latent (i.e. non-observable) variable with unknown thresholds, obtained from the classification of an unmeasurable metric variable, e.g. friendliness into low-middle-high levels;
4. the semi-standardized discrete variable with ordered categories (e.g. classification into dead, handicapped, and sound mice in an experiment);
5. the unstandardized discrete variable with ordered categories, as the level of satisfaction.

For further insights about categorical ordinal data we refer the reader to Agresti [1].

After the selection of the set of categorical ordinal variables, i.e. the set of segmentation variables, the clustering process can be outlined with five further consecutive steps described in Fig. 22.2. In the following, the inclusion of fuzzy theory on each clustering step, when ordinal variables are adopted as segmentation variables, is presented and discussed.

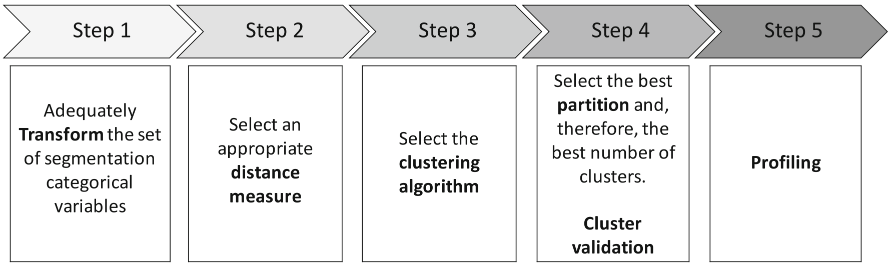

Step 1
Step 2
Step 3
Step 4
Step 5
Adequately **Transform** the set of segmentation categorical variables
Select an appropriate **distance measure**
Select the **clustering algorithm**
Select the best **partition** and, therefore, the best number of clusters.
Cluster validation
Profiling

Fig. 22.2 Steps that generally describe a cluster analysis for categorical ordinal variables

22.2.1 Step 1: Transformation of the Segmentation Variables

Likert-type scales are among the most commonly used categorical ordinal data in cluster analysis [13] and, according to the classification suggested by Kampen and Swyngedouw [29], they belong to Type 5, i.e. the unstandardized discrete variable with ordered categories. The popularity of Likert-type scales in tourist and marketing research is mainly due to the necessity to investigate personal opinions and emotions, as well as any other aspects that involve individual human spheres, using an easy-to-understand tool for respondents. Unfortunately, the use of this kind of variables is not without drawbacks. Essentially, respondents must convert their judgements on a scale where certain scale points might mean different things to different people. Therefore, data collected using Likert-type scales are vague, uncertain, imprecise, ambiguous, and the arbitrary assumption of equidistance between consecutive scale points make metric methods inappropriate. For a more comprehensive discussion of the methodological problems associated with Likert-type variables, and in general with ordinal answer formats, see [11, 29].

To overcome these drawbacks, the simple visual analogue scale or the fuzzy rating scale can be adopted instead of the traditional Likert-type scale [13, 20]. The visual analogue scale requires respondents to express their opinion indicating their positions along a continuous line between two end-points. On the other hand, the fuzzy rating scale is a modification of the traditional Likert-type scale that allows respondents to indicate both a preferred point on a scale and latitudes of acceptance on either side. Nowadays, these methods have not yet been extensively adopted in tourism and marketing research mainly for three reasons. Firstly, the mathematical and statistical methods/tools necessary to analyse the responses are quite complex. Secondly, when paper questionnaires are collected, an adequate software is required to decode the visual analogue scale and fuzzy rating scale answers and, at the moment, few institutions, universities, and organizations are willing to invest money in this. Finally, many academic researches are based on secondary data—i.e. data already collected by and available from other sources—which normally use traditional Likert-type scales.

A way to deal with the imprecision and vagueness inherent to Likert-type scales and, hence, to overcome their drawbacks consists of considering the Likert-type variables in a fuzzy framework [9], i.e. transforming the Likert-type scale levels into fuzzy variables. Coppi and D'Urso [5] suggested that the subjective evaluation recorded with a qualitative scale is best represented in a fuzzy framework, which reflects the uncertainty and the heterogeneity of individual evaluation. Furthermore, Hung and Yang [26] remarked that fuzzy sets suitably describe ambiguity and imprecision in natural language, connected to human perception.

A general class of fuzzy data is the LR fuzzy data [17, 41]. The k-th LR fuzzy variable ($k = 1, \dots, K$) observed on the i-th ($i = 1, \dots, I$) is denoted as $\tilde{x}_{ik} = (m_{1ik}, m_{2ik}, l_{ik}, r_{ik})_{LR}$ where m_{1ik} and m_{2ik} ($m_{2ik} > m_{1ik}$) are the left and right centres, respectively (the interval $[m_{1ik}, m_{2ik}]$ is usually referred to as the "core" of $\tilde{x}_{ik}$), and l_{ik} and r_{ik} the left and right spread, respectively. The membership function of each k-th LR fuzzy variable is defined as:

$$\mu_{\tilde{x}_{ik}}(u_{ik}) = \begin{cases} L\left(\frac{m_{1ik}-u_{ik}}{l_{ik}}\right) & , u_{ik} \leq m_{1ik}\ (l_{ik} > 0) \\ 1 & , m_{1ik} \leq u_{ik} \leq m_{2ik} \\ R\left(\frac{u_{ik}-m_{2ik}}{r_{ik}}\right) & , u_{ik} \geq m_{2ik}\ (r_{ik} > 0) \end{cases} \tag{22.1}$$

where L (and R) is a decreasing "shape" function from $\mathbb{R}^+$ to [0, 1] with $L(0) = 1$; $L(z_{ik}) < 1$ for all $z_{ik} = \frac{m_{1ik} - u_{ik}}{l_{ik}} > 0$, $\forall i, k$; $L(z_{ik}) > 0$ for all $z_{ik} < 1$, $\forall i, k$; $L(1) = 0$ (or $L(z_{ik}) > 0$ for all z_{ik} and $L(+\infty) = 0$). For the sake of completeness, Fig. 22.3 gives a representation of a general membership function for the k-th LR fuzzy variable.

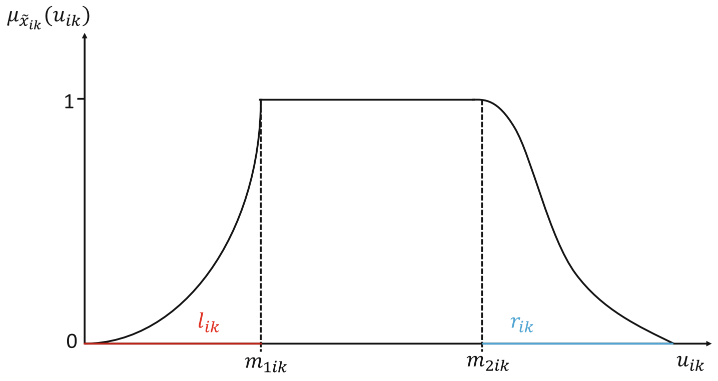


Fig. 22.3 LR membership function

The LR fuzzy variable $\tilde{x}_{ik}$ consists of an interval which runs from $(m_{1ik} - l_{ik})$ to $(m_{2ik} + r_{ik})$ and the membership functions give differential weights to the values in the interval, respectively, to the left and to the right of the left and right centres. The interval $[m_{1ik}, m_{2ik}]$ represents a range of plausible values for the corresponding value of the Likert-type scale, while the left and right spreads represent the uncertainty of the qualitative judgement. Finally, the collection of a set LR fuzzy variables into a matrix allows us to generate the LR fuzzy data matrix as follows:

$$\widetilde{\mathbf{X}} \equiv \{\tilde{x}_{ik} = (m_{1ik}, m_{2ik}, l_{ik}, r_{ik})_{LR} : i = 1, \dots, I;\ k = 1, \dots, K\}. \tag{22.2}$$

Two well-known LR fuzzy data, i.e. the trapezoidal and the triangular fuzzy data, are presented in the following. When L (and R) is of the form:

$$L(z_{ik}) = \begin{cases} 1 - z_{ik} & , 0 \le z_{ik} \le 1 \\ 0 & , \text{otherwise} \end{cases} \tag{22.3}$$

then $\widetilde{\mathbf{X}}$, as defined in (22.2), is a trapezoidal fuzzy data matrix whose elements have the following trapezoidal membership function:

$$\mu_{\tilde{x}_{ik}}(u_{ik}) = \begin{cases} 1 - \frac{m_{1ik}-u_{ik}}{l_{ik}} & , u_{ik} \leq m_{1ik} \; (l_{ik} > 0) \\ 1 & , m_{1ik} \leq u_{ik} \leq m_{2ik} \\ 1 - \frac{u_{ik}-m_{2ik}}{r_{ik}} & , u_{ik} \geq m_{2ik} \; (r_{ik} > 0). \end{cases} \tag{22.4}$$

The trapezoidal fuzzy data can therefore be expressed as $(m_{1ik} - l_{ik}, m_{1ik}, m_{2ik}, m_{2ik} + r_{ik})$, where $(m_{1ik} - l_{ik})$ and $(m_{2ik} + r_{ik})$ are the lower and upper bounds of the fuzzy data, respectively.

When $m_{1ik} = m_{2ik}$, the LR fuzzy data is characterized by a unique centre denoted as $\tilde{x}_{ik} = (m_{ik}, l_{ik}, r_{ik})_{LR}$. Consequently, the following particular case of LR fuzzy data matrix is defined:

$$\widetilde{\mathbf{X}} \equiv \{\tilde{x}_{ik} = (m_{ik}, l_{ik}, r_{ik})_{LR} : i = 1, \ldots, I; \; k = 1, \ldots, K\}. \tag{22.5}$$

Moreover, when L (and R) is of the form (22.3), $\widetilde{\mathbf{X}}$ is the triangular fuzzy data matrix, whose elements are characterized by the following triangular membership function:

$$\mu_{\tilde{x}_{ik}}(u_{ik}) = \begin{cases} 1 - \frac{m_{ik}-u_{ik}}{l_{ik}} & , u_{ik} \leq m_{ik} \; (l_{ik} > 0) \\ 1 - \frac{u_{ik}-m_{ik}}{r_{ik}} & , u_{ik} \geq m_{ik} \; (r_{ik} > 0). \end{cases} \tag{22.6}$$

In this case, the fuzzy triangular data can be expressed as $(m_{ik} - l_{ik}, m_{ik}, m_{ik} + r_{ik})$, where $(m_{ik} - l_{ik})$ and $(m_{ik} + r_{ik})$ are the lower and upper bounds of the fuzzy data, respectively.

The choice of the fuzzy coding of the Likert-type scale, the elicitation and specification of the membership function, as well as the analysis of robustness of the results obtained from a fuzzy data analysis are critical topics still considered as open problems in the literature [10, 11, 13].

22.2.2 Step 2: Distance for Fuzzy Data

When categorical ordinal variables are coded into fuzzy variables, it is necessary to consider both the distance for the centres and the distances for the spreads, since values are now represented by fuzzy data.

Several proximity measures (i.e. dissimilarity metrics, similarity measures) suitable for fuzzy data have been presented in the literature [e.g., 38, 39]. In the present chapter, the distance for fuzzy data proposed by Coppi et al.[6] is illustrated. The LR fuzzy data matrix as described in (22.5) will be considered from now on but the discussion can be easily generalized to any LR fuzzy data matrix (Eq. (22.2)).

Following Coppi et al. [6], the squared fuzzy distance between units i and the i', $d_F^2(\tilde{\mathbf{x}}_i, \tilde{\mathbf{x}}_{i'})$, can be computed as follows:

$$d_F^2(\tilde{\mathbf{x}}_i, \tilde{\mathbf{x}}_{i'}) = w_M^2 \|\mathbf{m}_i - \mathbf{m}_{i'}\|^2 + w_S^2 \left(\|\mathbf{l}_i - \mathbf{l}_{i'}\|^2 + \|\mathbf{r}_i - \mathbf{r}_{i'}\|^2 \right) \tag{22.7}$$

where $\tilde{\mathbf{x}}_i \equiv \{\tilde{x}_{ik} = (m_{ik}, l_{ik}, r_{ik})_{LR} : k = 1, \ldots, K\}$ denotes the fuzzy data vector for the ith unit; $\mathbf{m}_i$, $\mathbf{l}_i$, and $\mathbf{r}_i$ are the vectors of the centres and of the left and right spreads, respectively; $\|\mathbf{m}_i - \mathbf{m}_{i'}\|^2$ is the squared Euclidean distances between the centres; $\|\mathbf{l}_i - \mathbf{l}_{i'}\|^2$ and $\|\mathbf{r}_i - \mathbf{r}_{i'}\|^2$ are the squared Euclidean distances between the left and right spread, respectively; $w_M,\ w_S \geq 0$ are suitable weights for the centre and the spread components constrained by the following conditions: normalization condition $w_M + w_S = 1$; coherence condition $w_M \geq w_S \geq 0$.

Notice that the weights w_M and w_S can be either fixed a priori subjectively, considering external or subjective conditions (external weighting system), or objectively computed through a suitable data analysis procedure, e.g. a clustering procedure (internal weighting system). In the following analysis of fuzzy clustering we will refer to the internal weighting system.

22.2.3 Step 3: Fuzzy Clustering

The most popular fuzzy algorithms in the literature are the fuzzy C-means and the fuzzy C-medoids. Both methods have been extended to the "**bagged**" approach. In particular, D'Urso et al. [8] introduced the bagged fuzzy C-medoids clustering algorithm, while D'Urso et al. [10] proposed the bagged fuzzy C-means clustering algorithm. In the following, the fuzzy C-means, fuzzy C-medoids, bagged fuzzy C-means, and bagged fuzzy C-medoids clustering methods for fuzzy data are shown.

22.2.3.1 Fuzzy C-Means and Fuzzy C-Medoids Clustering for Fuzzy Data

The fuzzy C-means clustering algorithm for fuzzy data (FCMC-FD) [6] can be formalized as follows:

$$\begin{aligned} \min_{u_{ic}, \tilde{\mathbf{h}}_c, w_M} &: J_{FCMC-FD} \equiv \sum_{i=1}^{I} \sum_{c=1}^{C} u_{ic}^p d_F^2(\tilde{\mathbf{x}}_i, \tilde{\mathbf{h}}_c) = \\ & \sum_{i=1}^{I} \sum_{c=1}^{C} u_{ic}^p \left[w_M^2 \|\mathbf{m}_i - \mathbf{h}_c^M\|^2 + w_S^2 \left(\|\mathbf{l}_i - \mathbf{h}_c^L\|^2 + \|\mathbf{r}_i - \mathbf{h}_c^R\|^2 \right) \right] \\ s.t. & \sum_{c=1}^{C} u_{ic} = 1, \qquad u_{ic} \geq 0, \\ & w_M \geq w_S \geq 0;\ w_M + w_S = 1 \end{aligned} \tag{22.8}$$

where u_{ic} indicates the membership degree of the ith unit in the cth $(c = 1, \ldots, C)$ cluster; $p > 1$ is a weighting exponent that controls the fuzziness of the obtained partition; $d_F^2(\tilde{\mathbf{x}}_i, \tilde{\mathbf{h}}_c)$ represents the squared fuzzy distance between the ith unit and the prototype of the cth cluster, both expressed as fuzzy data vectors; $\tilde{\mathbf{h}}_c \equiv \{\tilde{h}_{ck} = (h_{ck}^M, h_{ck}^L, h_{ck}^R)_{LR} : k = 1, \ldots, K\}$ represents the fuzzy prototype of the cth cluster; $\mathbf{h}_c^M$, $\mathbf{h}_c^L$, and $\mathbf{h}_c^R$ represent, respectively, the centre and the left and right spreads of the c-th fuzzy prototype; $\|\mathbf{m}_i - \mathbf{h}_c^M\|^2$ is the squared Euclidean distances between the centres; $\|\mathbf{l}_i - \mathbf{h}_c^L\|^2$ and $\|\mathbf{r}_i - \mathbf{h}_c^R\|^2$ are the squared Euclidean distances between the left and right spread, respectively. For the iterative solutions with respect to $\tilde{\mathbf{h}}_c$, u_{ic}, w_M, and w_S see Coppi et al. [6].

The fuzzy C-medoids clustering algorithm for fuzzy data (FCMdC-FD) is very similar to the FCMC-FD algorithm, unless for the identification of the fuzzy prototypes of each cluster. In the FCMC-FD, the fuzzy prototypes, called centroids, are the weighted average, with weights equal to u_{ic}^p computed over all units, while in the FCMdC-FD, the fuzzy prototypes, called medoids, are observed units. Consequently, the FCMdC-FD clustering algorithm can be expressed as follows:

$$\begin{aligned}
&\min_{u_{ic}, \tilde{\mathbf{h}}_c, w_M} : J_{FCMdC-FD} \equiv \sum_{i=1}^{I} \sum_{c=1}^{C} u_{ic}^p d_F^2(\tilde{\mathbf{x}}_i, \hat{\tilde{\mathbf{x}}}_c) = \\
&\sum_{i=1}^{I} \sum_{c=1}^{C} u_{ic}^p \left[w_M^2 \|\mathbf{m}_i - \hat{\mathbf{m}}_c\|^2 + w_S^2 \left(\|\mathbf{l}_i - \hat{\mathbf{l}}_c\|^2 + \|\mathbf{r}_i - \hat{\mathbf{r}}_c\|^2 \right) \right] \\
&s.t. \sum_{c=1}^{C} u_{ic} = 1, \qquad u_{ic} \geq 0, \\
&w_M \geq w_S \geq 0; \; w_M + w_S = 1
\end{aligned} \tag{22.9}$$

where $d_F^2(\tilde{\mathbf{x}}_i, \hat{\tilde{\mathbf{x}}}_c)$ represents the squared fuzzy distance between the ith unit and the medoid of the cth cluster; $\hat{\tilde{\mathbf{x}}}_c \equiv \{\hat{\tilde{x}}_{ck} = (\hat{m}_{ck}, \hat{l}_{ck}, \hat{r}_{ck})_{LR} : k = 1, \ldots, K\}$ represents the fuzzy medoid of the cth cluster; $\hat{\mathbf{m}}_c$, $\hat{\mathbf{l}}_c$, and $\hat{\mathbf{r}}_c$ represent, respectively, the centre and the left and right spreads of the c-th fuzzy medoid; $\|\mathbf{m}_i - \hat{\mathbf{m}}_c\|^2$ is the squared Euclidean distances between the centres; $\|\mathbf{r}_i - \hat{\mathbf{r}}_c\|^2$ and $\|\mathbf{l}_i - \hat{\mathbf{l}}_c\|^2$ are the squared Euclidean distances between the left and right spread, respectively.

The iterative solutions with respect to u_{ic}, w_M, and w_S are similar to those of the FCMC-FD clustering algorithm.

22.2.3.2 Bagged Fuzzy *C*-Means and Fuzzy *C*-Medoids Clustering for Fuzzy Data

The basic idea of the bagged clustering algorithm, firstly introduced by Leisch [28], is to combine partitioning and hierarchical algorithms to overcome many of the limitations that arise performing the cluster analysis choosing only one of these two algorithms. Briefly, the bagged clustering algorithm starts drawing with replacement from the same dataset B bootstrap samples. The partitioning algorithm is then performed on the B bootstrap samples obtaining $(B \cdot C)$ centres, where C is the initial number of centres selected for the partitioning method. Finally, a hierarchical method is applied to the $(B \cdot C)$ centres in order to group the centres and, consequently, the units closest to them.

In the traditional bagged cluster algorithm, Leisch [28] suggested the use of the k-means as partitioning algorithm, but nothing prevents users from the adoption of any other partitioning algorithms. Taking advantage of the ease of which the bagged clustering algorithm can be integrated and modified, D'Urso et al. [8] suggested the bagged fuzzy C-medoids for fuzzy data (BFCMdC-FD), and D'Urso et al. [10] suggested the bagged fuzzy C-means for fuzzy data (BFCMC-FD).

Basically, the three important innovations introduced in the fuzzy version of the bagged clustering algorithms are: (1) the variables used for the clustering process are coded into fuzzy data before the creation of the B bootstrap samples; (2) the FCMC-FD or the FCMdC-FD is selected as partitioning algorithm; (3) each unit is assigned in a fuzzy manner to a cluster based on the membership degree of the unit to centres selected in the partitioning step.

Figure 22.4 schematically describes the steps of the BFCMC-FD/BFCMdC-FD algorithm.

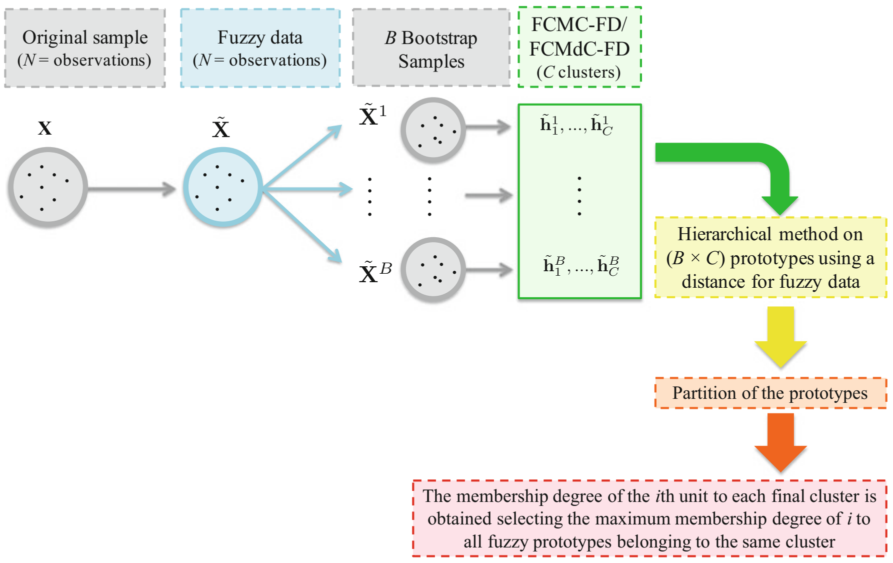

Original sample
(N = observations)
Fuzzy data
(N = observations)
B Bootstrap Samples
FCMC-FD/ FCMdC-FD
(C clusters)
$\mathbf{X}$
$\tilde{\mathbf{X}}$
$\tilde{\mathbf{X}}^1$
$\tilde{\mathbf{X}}^B$
$\tilde{\mathbf{h}}_1^1, \ldots, \tilde{\mathbf{h}}_C^1$
$\tilde{\mathbf{h}}_1^B, \ldots, \tilde{\mathbf{h}}_C^B$
Hierarchical method on ($B \times C$) prototypes using a distance for fuzzy data
Partition of the prototypes
The membership degree of the ith unit to each final cluster is obtained selecting the maximum membership degree of i to all fuzzy prototypes belonging to the same cluster

Fig. 22.4 The process of the fuzzy bagged cluster analysis

22.2.4 Step 4: Cluster Validation

An important step in cluster analysis regards the identification of the best cluster partition. Internal validity measures provide some useful guidelines in this respect (as suggested by Handl et al. [24]). An internal validity measure takes a cluster partition and the underlying dataset as the input, and uses information intrinsic to the data to assess the quality of the clustering. These measures take into account one or more of the following features of the obtained partition: (1) compactness, assessing cluster homogeneity, usually by looking at the intra-cluster variance; (2) connectedness, assessing whether observations are placed in the same cluster as their nearest neighbours in the data space; (3) separation, evaluating the separation between clusters. A number of different measures that combine these features are available in the literature. The most popular are the *Dunn index* [18], the *Davies-Bouldin index*[12], and the *silhouette index* [34].

When fuzzy cluster analysis is performed, suitable internal validity measures have been suggested in the literature [2, 37]. See also D'Urso et al. [7] for a comprehensive review on cluster validity measures. However, when original data is coded into fuzzy data, these kinds of measures must be suitably adapted in order to take the fuzzy nature of both units and final prototypes into consideration.

An example of a compactness measure for clustering of fuzzy data is the within-cluster variability (V_w) [11], which allows differences to be detected in the compactness across clusters and, if any, to individuate the main source of these differences, i.e. the centres or the spreads. Working with fuzzy data, $\tilde{\mathbf{x}}_i$, and fuzzy prototypes, $\tilde{\mathbf{h}}_c$, V_w is computed as the sum between the variability due to the centres (V_w^M) and the variability due to the spreads (V_w^S) as follows:

$$\begin{aligned}
V_w &= \frac{\sum_{i=1}^{N}\sum_{c=1}^{C} u_{ic}^p d_F^2(\tilde{\mathbf{x}}_i, \tilde{\mathbf{h}}_c)}{N} \\
&= \frac{\sum_{i=1}^{N} u_{ic}^p \left[w_M^2 \|\mathbf{m}_i - \mathbf{h}_c^M\|^2 + w_S^2\left(\|\mathbf{l}_i - \mathbf{h}_c^L\|^2 + \|\mathbf{r}_i - \mathbf{h}_c^R\|^2\right)\right]}{N} \\
&= \frac{\sum_{i=1}^{N} u_{ic}^p w_M^2 \|\mathbf{m}_i - \mathbf{h}_c^M\|^2}{N} + \frac{\sum_{i=1}^{N} u_{ic}^p \left[w_S^2\left(\|\mathbf{l}_i - \mathbf{h}_c^L\|^2 + \|\mathbf{r}_i - \mathbf{h}_c^R\|^2\right)\right]}{N} \\
&= V_w^M + V_w^S
\end{aligned} \tag{22.10}$$

where N is the number of units.

Another validity criterion useful for fuzzy clustering is the fuzzy silhouette (FS) coefficient [2], suitably adapted for fuzzy data. This index is computed as the weighted average of the individual silhouettes s_i as follows:

$$FS = \frac{\sum_{i=1}^{N}(u_{ir} - u_{iq})^{\beta} \cdot s_i}{\sum_{i=1}^{N}(u_{ir} - u_{iq})^{\beta}} \qquad s_i = \frac{(b_i - a_i)}{max\{b_i, a_i\}} \tag{22.11}$$

where s_i is the silhouette width of the i-th unit, and it measures the closeness of the i-th object to the objects in the highest membership cluster, with respect to the (fuzzy) distance to objects in other clusters; a_i is the average (fuzzy) distance of object i to all other objects belonging to its highest membership cluster and b_i is the minimum (over clusters) average (fuzzy) distance q_{iq} $(q = 1, \ldots, k)$ of object i to all objects belonging to another cluster q $(q \neq r)$; $(u_{ir} - u_{iq})^{\beta}$ is the weight attached to the silhouette width of the i-th unit; u_{ir} and u_{iq} are the first and second largest elements of the i-th row of the fuzzy partition matrix $\mathbf{U} = \{u_{ic}; i = 1, \ldots, N, \ c = 1, \ldots, K\}$; $\beta \geq 0$ is an optional user defined weighting coefficient. The traditional (crisp) silhouette coefficient is obtained as a particular case of FS by setting $\beta = 0$. The higher the value of FS, the better the assignment of the objects to the clusters, implying that the intra-cluster distance reaches its minimum value and, simultaneously, the inter-cluster distance its maximum value.

A further combined validity measure suitable for fuzzy clustering is the Xie and Beni cluster validity index (XB) [37] where the ratio of compactness to the separation of clusters is computed. Using this measure, the best partition is the one that allows the minimization of XB, indicating that the clusters are overall compact and separate from each other. In order to cope with the fuzzy nature of both units and prototypes, the XB for FCMC-FD is expressed as follows:

$$XB = \frac{\sum_{i=1}^{N}\sum_{c=1}^{C} u_{ic}^{p} d_F^2(\tilde{\mathbf{x}}_i, \tilde{\mathbf{h}}_c)}{\min_{c \neq c'} d_F^2(\tilde{\mathbf{h}}_c, \tilde{\mathbf{h}}'_c)}. \tag{22.12}$$

The extension of XB to the FCMdC-FD clustering algorithm can be obtained by substituting the fuzzy centroid $\tilde{\mathbf{h}}_c$ with the fuzzy medoid $\hat{\tilde{\mathbf{x}}}_c$.

22.2.5 Step 5: Profiling

Once an optimal partition is obtained, the further and final step is to profile the final clusters. Clusters profiling generally consist of describing clusters using a set of available information different from the one used to segment, by means of descriptive statistics such as the mean, median, variance, and preferably even more information if available, computed for each cluster. In tourism marketing, socio-economics, demographic, and travelling characteristics collected through the surveys are some common variables involved in the profiling step.

When a fuzzy clustering method is performed, the common approach to profile each cluster is to assign each unit to the cluster with the highest membership degree, adopting a "defuzzification" procedure and/or specifying a cut-off point for membership degree. Then, the profiling is performed on the "hard" partition obtained. Since the essence of any fuzzy clustering method is to allow units to simultaneously belong to different clusters quantifying the degree of membership of each unit to each cluster, the defuzzification procedure seems contradictory in itself. To overcome this drawback, capturing at the same time the vagueness of the allocation of each unit to all clusters, it is possible to make use of the membership degree to weigh any statistics. For instance, the weighted proportion and the weighted average values can be computed and the results are easy to interpret by practitioners and researchers in social sciences since they are very similar to the results obtained after the adoption of any more traditional cluster methods [e.g., 8, 10, 11].

22.3 Fuzzy Clustering for Tourism Segmentation: Examples

Two datasets are studied in this section. The first one is drawn from the annual inbound survey "International Tourism in Italy", conducted by the Bank of Italy since 1996. The second one is related to a survey on Chinese perceptions of Western Europe and resulted from a study conducted in Beijing in 2012.

22.3.1 Case 1: Travellers' Satisfaction Towards a Tourist Destination (D'Urso et al. [11])

The main aim of the survey of this first study is to monitor travel expenditure and length of stay of inbound and outbound visitors to/from Italy. Tourists have been interviewed face-to-face at the end of their trip, when they are returning to their place of habitual residence. Interviews have been conducted at different moments of the day, during both working days and holidays, and month by month, with a fixed number of interviews per each period of survey, at different national borders (including highways, railway, airports, and harbours). The stratified sampling method has been applied using different types of stratified variables per each type of frontier. In the questionnaire, socio-demographic characteristics of the interviewees (gender, age, employment status, and country of origin); information regarding the trip (such as number of cities visited during the trip, party group, motives, travel expenditure); and levels of satisfaction with different aspects of the destination have been collected. A sample of about 1000 international visitors who spent a holiday in South-Tyrol (Northern Italy) in 2010 and 2011 has been analysed through the FCMC-FD algorithm using the set of satisfaction variables, measured by means of a 10-point Likert-type scale, as segmentation variables.

Firstly, the set of Likert-type variables have been transformed, using the process presented in Sect. 22.2.1 of this chapter, into triangular fuzzy variables and Fig. 22.5 graphically displays the fuzzy recoding. For instance, the value 1 ("very unsatisfied") in the Likert-type scale corresponds to a fuzzy data in the range [1, 2.5], while the value 10 ("very satisfied") in the Likert-type scale corresponds to a fuzzy data in the range [8.5, 10]. In both cases, the degree of vagueness/uncertainty, computed as the difference between the upper and lower bounds, is equal to 2.5 but it is relevant to observe that the degree of vagueness becomes smaller the more the mid-values are observed, e.g. the values 5 and 6 while in the Likert-type scale both have a degree of vagueness of 1. This fuzzy recoding has been chosen since usually respondents know the difference between values 5 and 6, i.e. these values are little vague: the former corresponds to a mild negative judgement, while the latter corresponds to a mild positive judgement. On the contrary, respondents might not really understand/appreciate the difference between 1 and 2 (both implying a strong negative judgement), or between 9 and 10 (both implying a strong positive judgement), i.e. these values incorporate a higher degree of uncertainty [11].

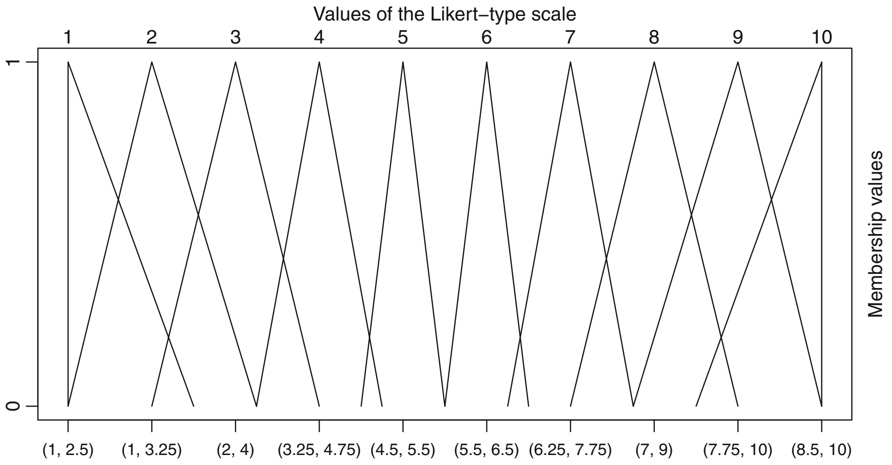


Fig. 22.5 Linguistic satisfaction terms in the form of fuzzy data [11]

After performing the FCMC-FD algorithm, employing the distance suggested by Coppi et al. [6], the XB cluster validity index (see Eq. (22.12)) has been adopted and the three-clusters solution has been identified as the best partition. The within-clusters variability of the three clusters, as well as the variabilities according to the centres and the spreads, has been computed following Eq. (22.10) and presented in Table 22.1. Results show that all clusters are equally compact since there are very little differences emerging between clusters.

Table 22.1 Variability within clusters according to the components of the fuzzy data (centres and spreads) (Data source: [11])

Within variability index	Cluster 1	Cluster 2	Cluster 3
Centres (V_w^M)	0.777	0.719	0.785
Spread (V_w^S)	0.427	0.415	0.386
Total (V_w)	1.204	1.134	1.171

Table 22.2 shows the values of the fuzzy centroids for the final clusters solution together with the rank (deceasing order, i.e. from the most (1) to the least (10) satisfactory) of the ten aspects describing the destination helpful to further understand the differences in satisfaction among the three clusters. Notice that a fuzzy centroid is a vector of K (where K indicates the number of variables used for the clustering process) fuzzy data each of which, in our case, are described by three values, i.e. lower, centre, and upper values.

Figure 22.6 reports the final fuzzy centroids represented by radar plots. The black solid line represents the centres of the fuzzy centroids, the green dashed line represents the lower values (inner lines), and the red dashed line the upper values

Table 22.2 Rank of the different aspects of the visited destination for each cluster [11]

Satisfaction	"Unfulfilled"	Rank	"With reservations"	Rank	"Enthusiasts"	Rank
Friendliness	(7.423, 7.916, 8.330)	2	(7.956, 8.619, 9.065)	2	(8.472, 9.300, 9.584)	6
Art	(7.223, 7.643, 8.012)	4	(7.791, 8.396, 8.844)	4	(8.492, 9.327, 9.599)	4
Landscape	(7.499, 8.002, 8.397)	1	(8.146, 8.863, 9.267)	1	(8.733, 9.644, 9.817)	1
Accommodation	(7.188, 7.611, 7.997)	7	(7.727, 8.321, 8.784)	7	(8.496, 9.335, 9.593)	3
Food and beverages	(7.219, 7.639, 8.010)	5	(7.782, 8.385, 8.839)	5	(8.479, 9.310, 9.577)	5
Prices	(5.845, 6.038, 6.226)	10	(6.412, 6.717, 7.001)	10	(7.525, 8.088, 8.458)	10
Products sold	(7.031, 7.402, 7.755)	9	(7.446, 7.946, 8.395)	9	(8.164, 8.892, 9.267)	9
Information	(7.149, 7.559, 7.935)	8	(7.675, 8.251, 8.711)	8	(8.439, 9.258, 9.573)	8
Safety	(7.244, 7.689, 8.072)	3	(7.893, 8.541, 8.976)	3	(8.572, 9.437, 9.697)	2
Overall	(7.209, 7.627, 8.033)	6	(7.746, 8.338, 8.879)	6	(8.448, 9.269, 9.663)	7

Note: In brackets are reported the lower bounds, the centres, and the upper bounds of the fuzzy data

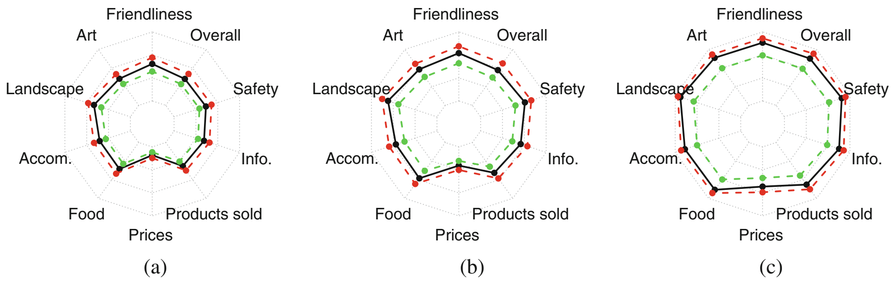


Fig. 22.6 Radar plots of fuzzy centroids per each cluster. Centres are represented with the black solid lines, upper and lower bounds are represented with red and green dashed lines, respectively [11]. (**a**) Unfulfilled. (**b**) With reservations. (**c**) Enthusiast

(outer lines) of the fuzzy centroids. By looking into these plots, the first Fig. 22.6a and the second Fig. 22.6b clusters can be labelled, respectively, the "Unfulfilled" and the "Enthusiasts" clusters, since they grouped people who are less and more satisfied with the investigated aspects, respectively. Likewise, the third cluster Fig. 22.6c has been labelled "With reservations" since it groups visitors who are neither much nor little satisfied with the entire set of variables used in the clustering process. From the inspection of both Table 22.2 and Fig. 22.6, it is possible to affirm that tourists are, in general, not satisfied with prices and cost of living ("Prices") of the destination. In fact, the values of the centres for the variable "Prices" are the lowest among the other variables for each cluster. Furthermore, also the uncertainty (spreads) related to this variable is very low.

To represent how the respondents are grouped in the three final clusters according to their membership degree, it is possible to make use of a suitable graph called ternary plot (Fig. 22.7). In particular, looking at this graph it is possible to observe that there is a considerable group of respondents who belong to the "Enthusiasts" with a high membership degree (between 60% and 80%) and, simultaneously, to the "With reservations" with a membership degree between 10% and 30%, and to the "Unfulfilled" with a membership degree lower than 20%. Similarly, there is a group of respondents who mainly belong to the "Unfulfilled" (membership degree between 60% and 80%) and simultaneously belong to the "With reservations" and to the "Enthusiasts", despite with a membership degree lower than 30%. Conversely, only a few respondents belong to the "With reservations" cluster with a high membership degree (between 60% and 80%), indicating that only a few tourists are undecided regarding the overall satisfaction against of tourist destination analysed.

To profile the clusters, socio-demographic characteristics—age, gender, employment status, and country of origin—and trip characteristics—purpose of the trip and expenditure behaviour—have been used. In this step, the weighted relative frequencies have been calculated. The clusters are therefore characterized by "virtual" tourists made up of the fusion of the common features of different observed tourists belonging to each cluster with a specific degree of membership. In this respect, the weighted sample size of the three clusters can be computed and the results suggest that tourists have been equally divided among "Enthusiasts" (35%), "With reservations" (31%), and "Unfulfilled" (34%). Finally, in order to verify whether the membership to different clusters depends on each of the profiling variable, the Chi-square tests have been calculated on the weighted relative frequencies tables (Table 22.3). As for the socio-demographic characteristics, the only significant difference between clusters is due to the country of origin: the "Unfulfilled" group is made up of a higher proportion of Austrian tourists than the other two

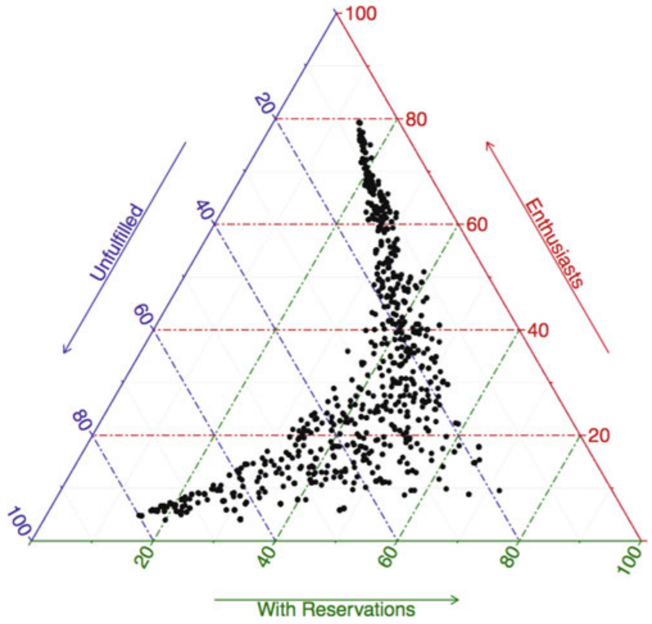


Fig. 22.7 Ternary plot [11]

clusters; the "Enthusiasts" grouped more German tourists and tourists from other European countries than the other two groups; while the "With reservations" presents the highest percentage of tourists from countries outside Europe. Regarding the travelling characteristics, the "Unfulfilled" group has the highest proportion of tourists travelling alone, who are visiting Italy for the first time, and who are travelling for business or other personal motivations (not including holidays). The "Enthusiasts" comprises the highest proportion of tourists who had already visited Italy before and those who are travelling for leisure purposes (mountain holidays, cultural holidays, other kinds of holidays). Therefore, these findings suggest that the satisfaction with the destination is a feeling highly influenced by the time that tourists spend to actually visit the destination, i.e. higher the visiting time, higher the level of satisfaction. As a consequence, holidays travellers are more satisfied with South Tyrol than business travellers since they have more time to enjoy what this destination offers. Finally, with regard to the travel expenditure behaviour, the "Enthusiasts" have the highest proportion of visitors who spend on accommodation, transportation, food and beverage, and shopping, while the "Unfulfilled" have the lowest proportion in each of these expenditure items. Therefore, as expected, it seems that the higher the level of satisfaction with the destination, the higher the willingness to consume goods and services offered by the destination.

This analysis helps practitioners to obtain comprehensible and easy-to-read results, since they appear similar to the results of other more traditional crisp clustering techniques, which can be employed for the creation of future management and marketing strategies, and the development and maintenance of a competitive advantage for the tourist destination against competitors.

22.3.2 *Case 2: Travellers' Perceptions of the Image of a Tourist Destination (D'Urso et al. [10])*

Chinese perceptions of Western Europe have been captured through a survey conducted in Beijing in 2012. Overall, 21 image attributes have been used to measure the tourism product generally offered to Chinese travellers such as attractive scenery and natural attractions and cultural/historical attractions, and the more general images of Western Europe such as cities with modern technology, quality accommodation, and quality tourist services. These attributes have been quantified through a 7-point Likert-type scale, where the value 1 means "offers very little" and the value 7 means "offers very much". Socio-demographics, travelling characteristics, and information sources most likely to use to plan a trip to Western Europe (TV, radio advertising, guidebook, internet search engine, etc.) were also collected. A sample of 328 respondents has been analysed by means of the BFCM-FD algorithm using the set of image variables as segmentation variables.

Firstly, the set of Likert-type variables have been transformed into fuzzy variables adopting the fuzzy coding suggested by Kazemifard et al.[30] and represented in Fig. 22.8.

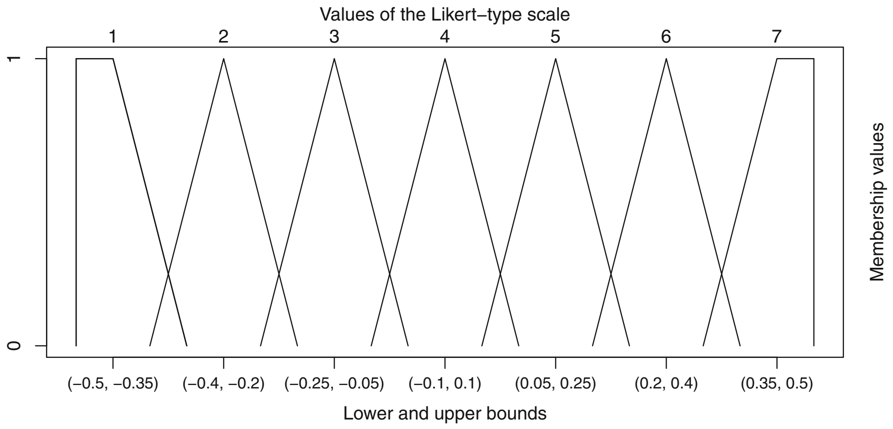


Fig. 22.8 Fuzzy recoding of the 7-point Likert-type variables [10]

Table 22.3 Socio-demographic characteristics of the visitors and travelling characteristics (percentage values) [11]

Variables	Sample	"Unfulfilled"	"Enthusiasts"	"With reservations"	*p*-Value
Socio-demographic characteristics					
Male	68.91	70.74	66.13	69.60	
Age					
Less than 35 years old	21.16	22.39	19.74	21.25	
35–44 years old	28.59	26.57	31.07	28.33	
45–64 years old	36.41	35.52	37.22	36.54	
More than 64 years old	13.84	15.52	11.97	13.88	
Employment status					
Self-employed	11.57	11.38	11.61	11.68	
Clerk	16.20	14.67	17.74	16.52	
Other employee	53.72	53.89	52.58	54.42	
Retired	12.47	14.07	11.29	11.97	
Other	6.04	5.99	6.78	5.41	
Country of origin					***
Austria	21.06	29.85	14.84	18.13	
Germany	50.85	42.99	56.45	53.26	
Other EU countries	21.46	20.59	22.58	21.25	
Outside EU	6.63	6.57	6.13	7.36	
Trip characteristics					
Visit alone	23.97	31.14	18.71	21.81	***
Only one cities visited in this trip	84.05	86.97	81.61	83.57	
Number of times in Italy before					***
Zero	23.97	31.14	17.10	23.23	
Up to five times	24.87	22.15	27.10	25.50	
More than five times	51.15	46.71	55.80	51.27	
Main purpose of travel					***
Mountain holiday	46.14	39.70	50.48	48.43	
Cultural holiday	18.86	19.40	19.29	17.95	
Other kind of holiday	11.03	9.55	12.86	10.83	
Other personal motivations	13.44	17.92	9.97	12.25	
Business	10.53	13.43	7.40	10.54	
Expenditure behaviour					
Accommodation	84.25	73.05	93.55	86.67	***
Transportation	71.51	58.98	83.97	72.52	***
Food and beverages	83.35	77.91	87.70	84.70	***
Shopping	72.52	68.66	76.77	72.44	*
Other services	35.31	31.94	37.10	36.93	

Note: Significance of the Chi-square test was reported. All test results are not significant unless indicated otherwise: *Significant at $p \leq 0.1$, ***Significant at $p \leq 0.01$

The distance between two units has been computed using Eq. (22.7), as explained in Sect. 22.2.2, and the BFCM-FD has been performed. Figure 22.9 graphically displays the output of the clustering algorithm. Specifically, the bottom panel shows the dendrogram along with the best partition obtained through the investigation of the average silhouette index value calculated following Eq. (22.11) for any partition between 2 and 20 (see Fig. 22.9, top panel). The peaks in the silhouette series suggest that the Chinese travellers can be segmented into two groups followed by the four-clusters solution. Since the four-clusters solution allows us to obtain a more precise and detailed characterization of the market segments in comparison to the two-clusters solution, this partition has been further investigated.

In order to label, describe, and identify the four final clusters, the weighted average values of the image attributes have been calculated and graphically displayed in Fig. 22.10. The weighted average values of the image attributes suggest that cluster

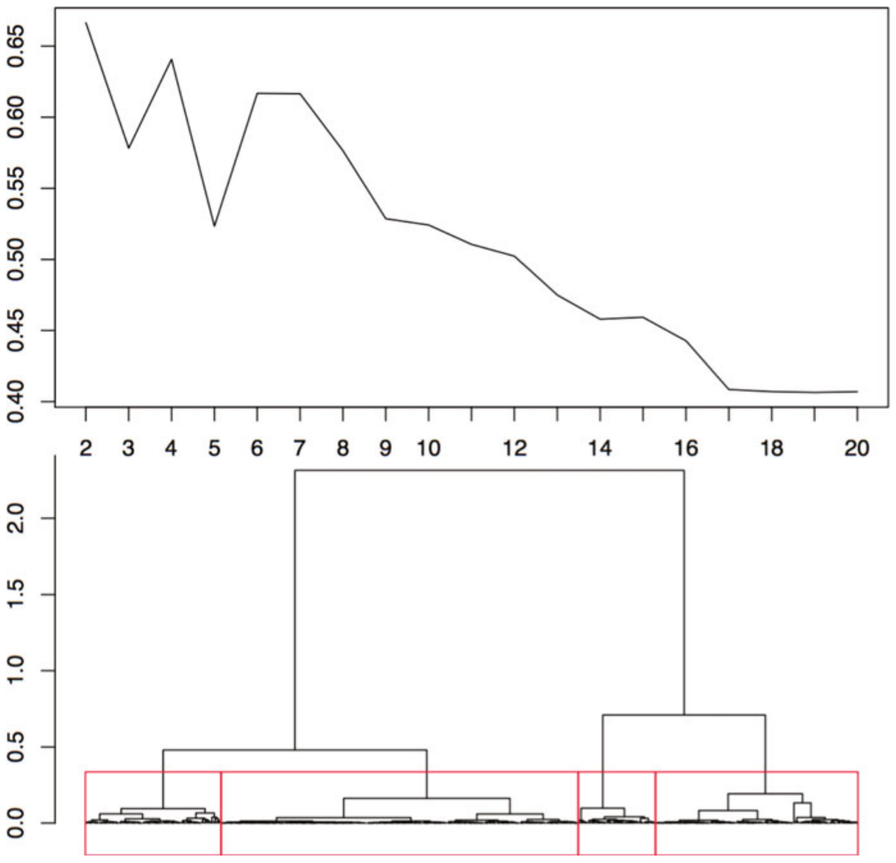


Fig. 22.9 Values of the silhouette index per each cluster partition from 2 to 20 (top panel) and dendrogram (bottom panel) (Data source: [10])

Table 22.4 Characteristics and preferences of the travellers, and characteristics of the trip (percentage values) [10]

Variables	Sample	"Apathetics"	"Moderates"	"Enthusiasts"	"Admirers"	p-Value
Female	57.32	43.90	54.63	63.44	75.56	***
Individual monthly income equal to 7000 RMB or less	67.08	68.29	65.42	64.13	75.00	
Single	61.06	60.98	62.75	66.30	46.67	
University degree or less	62.46	56.10	58.49	69.57	68.89	
Between 18 and 25 years old	51.53	52.44	51.40	56.52	40.00	
Full-time employee	54.29	58.02	51.85	50.00	62.22	
Trip characteristics						
3–5 star hotel is the preferred type of accommodation	42.64	41.46	35.19	44.57	59.09	*
First-timer in Western Europe	76.95	75.64	72.90	83.70	75.00	
The estimated duration of the next trip to Western Europe is less than 2 weeks	62.58	62.96	64.81	58.70	64.44	
Party group of the next trip to Western Europe: family or partner	60.37	63.29	64.49	50.00	66.67	
Main Purpose of travel						
Visiting friends and relative	3.96	3.66	4.63	2.15	6.67	
Study	19.21	19.51	20.37	21.51	11.11	
Work	5.18	2.44	9.26	4.30	2.22	
Holiday	83.54	78.05	82.41	84.95	93.33	
What destinations are you most likely to visit?						
UK	55.49	46.34	51.85	64.52	62.22	*
Italy	54.57	58.54	51.85	54.84	53.33	
Belgium	13.41	15.85	12.96	8.60	20.00	
Portugal	9.45	4.88	12.96	7.53	13.33	
France	72.56	70.73	65.74	74.19	88.89	**
Switzerland	53.05	46.34	52.78	51.61	68.89	
Ireland	17.99	14.63	20.37	16.13	22.22	
Netherlands	30.79	30.49	33.33	29.03	28.89	
Germany	39.63	29.27	39.81	41.94	53.33	*
Spain	39.33	32.93	45.37	37.63	40.00	

(continued)

Table 22.4 (continued)

Variables	Sample	"Apathetics"	"Moderates"	"Enthusiasts"	"Admirers"	*p*-Value
Austria	22.87	14.63	25.00	25.81	26.67	
Greece	50.30	37.80	47.22	56.99	66.67	***
What information source are you likely to use to plan your trip to Western Europe?						
TV or radio advertising	15.85	19.51	17.59	13.98	8.89	
Guidebook	33.84	24.39	44.44	29.03	35.56	**
Internet search engine	77.13	76.83	75.93	84.95	64.44	*
Travel agency	44.51	42.68	38.89	51.61	46.67	
Travel forums and blogs	47.56	35.37	50.00	53.76	51.11	*
Special magazine	29.88	26.83	32.41	31.18	26.67	

Notes: All Chi-square tests calculated are not significant unless indicated otherwise: *Significant at $p \leq 0.1$, **Significant at $p \leq 0.05$, ***Significant at $p \leq 0.01$

4 is a niche cluster grouping of Chinese travellers who believe more than the other travellers that Western Europe offers all the image attributes considered. Therefore, cluster 4 has been labelled the "Admirers" group. Chinese travellers who think that Western Europe "offers very little" in general, i.e. travellers whose average scores for the 21 attributes considered are very low, have been grouped in cluster 3, labelled the "Apathetics". Clusters 1 and 2 group respondents who rated the set of attributes with values on average always in the middle between the average score registered for the "Apathetics" and the "Admirers". Therefore, cluster 1 and cluster 2 have been respectively labelled as the "Enthusiasts" and the "Moderates".

In order to profile the four final clusters, the weighted proportion of the socio-demographic and travel characteristics of a possible trip to Western Europe have been calculated. The results of the Chi-square tests performed between each profiling variable and the membership to the different clusters are presented in Table 22.4. This analysis suggests, for instance, that the higher the proportion of females in the clusters, the higher the perception of what Western Europe offers on all the image attributes considered. The "Enthusiasts" are likely to use the Internet as a source of travel information and they prefer the UK as a future destination to visit, unlike the "Admirers" who prefer to visit France. Finally, the "Moderates" are likely to use guidebooks while the "Apathetics" use travel forums and blogs as source of travel information less than other groups.

Furthermore, it is particularly interesting to notice that Chinese tourists are attracted by safe, secure, and clean destinations. As a consequence, these attributes can be effectively used by European countries for destination advertising and promotion in China. Contrary, the new generation of Chinese tourists is not really interested in "festivals, events and shows", i.e. cultural-historical itineraries. The younger generation is instead more attracted by destinations made famous, thanks to movies, music, sports, or personal idols and European destinations need to promote festivals, events, and shows relevant for these kind of tourists. In conclusion, the

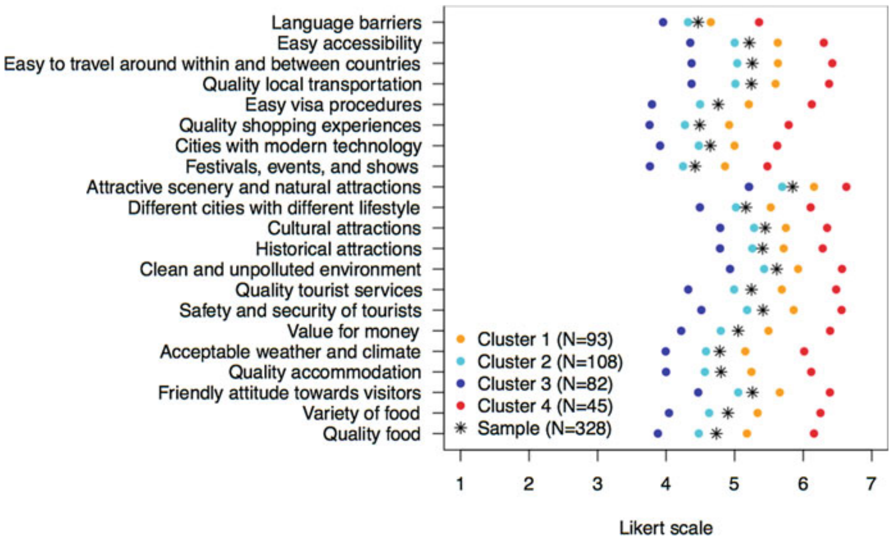


Fig. 22.10 Weighted mean of the segmentation variable, i.e. image attributes [10]

results can be used to monitor the evolution of the image of Western Europe in China and to assist destination marketers in selecting the appropriate image associations for both destination differentiation and positioning purposes.

22.4 Conclusions

In this chapter, an effective way to deal with many of the disadvantages arising when Likert-type variables are used as segmentation variables in cluster analysis is presented. The procedure here illustrated regards the inclusion of the fuzzy theory at each step of the clustering algorithm. Firstly, the transformation of Likert-type variables into fuzzy variables is presented as a way to overcome most of the problems arising in using this kind of categorical ordinal variables in cluster analysis. The LR fuzzy data, i.e. one of the most common family of fuzzy data, has therefore been presented. One of the main consequences in transforming the segmentation variables in fuzzy variables is the necessity to modify accordingly the distance/dissimilarity measure used in the clustering algorithm. Among the different distance measures suggested in the literature, the distance for fuzzy data proposed by Coppi et al. [6] is presented in this chapter. With regard to the clustering algorithm, the two most common fuzzy clustering algorithms, i.e. the fuzzy C-means and the fuzzy C-medoids, together with two of their recent generalizations, i.e. the bagged fuzzy C-means and the bagged fuzzy C-medoids, have been theoretically discussed. In the presentation of these fuzzy clustering algorithms, the fuzzy nature of the segmentation variables, as well as of the prototypes/medoids, has been highlighted. Furthermore, due to the fuzzy nature of the resulting prototypes/medoids, suitable generalizations of a few common validity indexes have been presented. To profile the final partition preserving the idea that each unit can belong to more than one cluster simultaneously, the final membership degrees matrix has been suggested as a weight for any statistics, as, for instance, in the calculation of the weighted means and the weighted proportions necessary to describe the final clusters. Finally, the applicability and effectiveness of the fuzzy clustering procedure suggested here is illustrated through two applications drawn from the tourism fields.

References

1. Agresti, A. (1990) Categorical data analysis. Wiley Series in Probability and Mathematical Statistics: New York
2. Campello, R. J., and Hruschka, E. R. (2006) A fuzzy extension of the silhouette width criterion for cluster analysis. Fuzzy Sets and Systems, 157:2858–2875
3. Chaturvedi, A. and Carroll, J. D. and Green, P. E. and Rotondo, J. A. (1997) A feature-based approach to market segmentation via overlapping k-centroids clustering. Journal of Marketing Research, 34:370–377
4. Chiang, W.-Y. (2011) Establishment and application of fuzzy decision rules: an empirical case of the air passenger market in Taiwan. International Journal of Tourism Research, 13:447–456
5. Coppi, R., and D'urso, P. (2002) Fuzzy K-means clustering models for triangular fuzzy time trajectories. Statistical Methods and Applications, 11:21–40
6. Coppi, R., D'Urso, P., and Giordani, P. (2012) Fuzzy and possibilistic clustering for fuzzy data. Computational Statistics & Data Analysis, 56:915–927
7. D'Urso, P. (2014) Fuzzy clustering, in: C. Hennig, M. Meila, F. Murtagh, R. Rocci (Eds.), Handbook of Cluster Analysis, Chapman and Hall
8. D'Urso, P., De Giovanni, L., Disegna, M., and Massari, R. (2013) Bagged Clustering and its application to tourism market segmentation. Expert Systems with Applications, 40:4944–4956
9. D'Urso, P., De Giovanni, L., and Massari, R. (2014) Self-organizing maps for imprecise data. Fuzzy Sets and Systems, 237:63–89
10. D'Urso, P., Disegna, M., Massari, R. and Prayag, G. (2015) Bagged fuzzy clustering for fuzzy data: An application to a tourism market. Knowledge-Based Systems, 73:335–346
11. D'Urso, P., Disegna, M., Massari, R. and Osti, L. (2016) Fuzzy segmentation of postmodern tourists. Tourism Management, 55:297–308
12. Davies, D.L. and Bouldin, D.W. (1979) A cluster separation measure. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2:224–227
13. de la Rosa de Sá, S., Gil, M.A., González-Rodríguez, G., López, M.T. and Lubiano, M.A. (2015) Fuzzy rating scale-based questionnaires and their statistical analysis, IEEE Transactions on Fuzzy Systems, 23(1):111–126
14. De Wilde, P. (2004) Fuzzy utility and equilibria. IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics, 34(4): 1774–1785
15. Dolnicar, S. (2002) A Review of Data–Driven Market Segmentation in Tourism. Journal of Travel & Tourism Marketing, 11(1):1–22
16. Dolnicar, S. and Grün, B. (2007) How constrained a response: A comparison of binary, ordinal and metric answer formats. Journal of Retailing and Consumer Services, 14:108–122
17. Dubois, D. and Prade, H. (1988) Possibility theory. Plenum press, New York

18. Dunn, J.C. (1974) Well separated clusters and fuzzy partitions. Journal of Cybernetics, 4:95–104
19. Everitt, B.S. and Landau, S. and Leese, M. (2001) Cluster analysis (4th edition). Arnold Press: London
20. Gil, M.A. and González-Rodríguez, G. (2012) Fuzzy *vs*. Likert Scale in Statistics. E. Trillas et al. (Eds.): Combining Experimentation and Theory, STUDFUZZ 271:407–420
21. Goneos-Malka, A., Strasheim, A., and Grobler, A. (2014) Conventionalists, Connectors, Technoisseurs and Mobilarti: differential profiles of mobile marketing segments based on phone features and postmodern characteristics of consumers. Journal of Retailing and Consumer Services, 21:905–916
22. Georgescu, I. (2007) Similarity of fuzzy choice functions. Fuzzy Sets and Systems, 158(12): 1314–1326
23. Georgescu, I. (2010) Arrow index of a fuzzy choice function. Fundamenta Informaticae, 99(3):245–261
24. Handl, J., Knowles, J. and Kell, D. B. (2005) Computational Cluster Validation in Post-Genomic Data Analysis. Bioinformatics, 21(15):3201–3212
25. Heiser, Willem J. and Groenen, Patrick J. F. (1997). Cluster differences scaling with a within-clusters loss component and a fuzzy successive approximation strategy to avoid local minima. Psychometrika, 62(1): 63–83
26. Hung, W.-L. and Yang, M.-S. (2005) Fuzzy clustering on LR-type fuzzy numbers with an application to Taiwanese tea evaluation. Fuzzy Sets and Systems, 150:561–577
27. Hwang, Heungsun and DeSarbo, Wayne S. and Takane, Yoshio (2007). Fuzzy clusterwise generalized structured component analysis. Psychometrika, 72(2):181–198
28. Leisch, F. (1999) Bagged clustering. Working paper 51 SFB adaptive information systems and modelling in economics and management science WU Vienna University of Economics and Business
29. Kampen, J. and Swyngedouw, M. (2000) The ordinal controversy revisited. Quality and Quantity 34(1): 87–102
30. Kazemifard, M., Zaeri, A., Ghasem-Aghaee, N., Nematbakhsh, M. A., and Mardukhi, F. (2011) Fuzzy Emotional COCOMO II Software Cost Estimation (FECSCE) using Multi-Agent Systems, Applied Soft Computing, 11(2):2260–2270
31. Kotler, P. (1988) Marketing Management. 6th edition Prentice-Hall: Englewood Cliffs, NJ
32. Li, X. and Meng, F. and Uysal, M. and Mihalik, B. (2013) Understanding China's long-haul outbound travel market: An overlapped segmentation approach. Journal of Business Research, 66:786–793
33. McBratney, A. B. and Moore, A. W. (1985). Application of fuzzy sets to climatic classification. Agricultural and Forest Meteorology, 35(1):165–185.
34. Rousseeuw, P.J. (1987) Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. Journal of Computational and Applied Mathematics, 20:53–65
35. Samuelson, P. A. (1938) A note on the pure theory of consumer's behaviour. Economica, 61–71
36. Wang, D., Niu, Y., Lu, L., and Qian, J. (2015) Tourism spatial organization of historical streets e a postmodern perspective: the examples of Pingjiang Road and Shantang Street, Suzhou, China. Tourism Management, 48: 370–385
37. Xie, X. and Beni, G. (1991) A validity measure for fuzzy clustering. IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI), 13:841–847
38. Yang, M., and Ko, C. (1996) On a class of fuzzy c-numbers clustering procedures for fuzzy data. Fuzzy Sets and Systems, 84:49–60
39. Yang, M.S., and Liu, H.H. (1999) Fuzzy clustering procedures for conical fuzzy vector data. Fuzzy Sets and Systems, 106:189–200
40. Zadeh, L. (1965) Fuzzy sets. Information and control, 8:338–353
41. Zimmermann, H. J. (2011) Fuzzy set theory and its applications. Kluwer Academic Press, Norwell

Chapter 23
Towards Personalized Data-Driven Bundle Design with QoS Constraint

Mustafa Mısır and Hoong Chuin Lau

Abstract In this paper, we study the bundle design problem for offering personalized bundles of services using historical consumer redemption data. The problem studied here is for an operator managing multiple service providers, each responsible for an attraction, in a leisure park. Given the specific structure of interactions between service providers, consumers and the operator, a bundle of services is beneficial for the operator when the bundle is underutilized by service consumers. Such revenue structure is commonly seen in the cable television and leisure industries, creating strong incentives for the operator to design bundles containing lots of not-so-popular services. However, as customers might choose to bypass a bundle completely if it is not sufficiently attractive, we need to impose a quality of service (QoS) constraint on the lower bound of the perceived attractiveness. In this paper, we make two major contributions (1) recognizing the inherent differences in customer preferences, we propose an approach for detecting different user classes, and for each user class, make an appropriate bundle recommendation; and (2) in order to make the bundling scheme even more adaptive to unknown customer preferences, we propose a dynamic bundling strategy, which allows customers to "trade in" any number of undesirable services dynamically so that they can be replaced by an alternative set of services. A step to generate fixed or static bundles is also studied. The pros and cons of different bundling strategies are illustrated using a real-world dataset collected from a large leisure park operator in Asia that manages a large collection of attraction providers.

This work was done while the author was a post-doc at the Singapore Management University.

M. Mısır (✉)
Nanjing University of Aeronautics and Astronautics, College of Computer Science and Technology, Nanjing, Jiangsu, China
e-mail: mmisir@nuaa.edu.cn

H. C. Lau
Singapore Management University, School of Information Systems, Singapore, Singapore
e-mail: hclau@smu.edu.sg

P. Moscato, N. J. de Vries (eds.), *Business and Consumer Analytics: New Ideas*,
https://doi.org/10.1007/978-3-030-06222-4_23

Keywords Bundling · Dynamic recommendation · Static recommendation · Customer segmentation · Recommender systems · Matrix factorization

23.1 Introduction

Product development [27] is an important part of service innovation and marketing today. While product development could refer to designing new products from scratch or releasing an upgraded/modified version of an existing product, it could also refer to combining existing products as packages that we see in travel, financial, healthcare, information and telecommunications services. The latter process is commonly known as product bundling, which may entail both design and pricing aspects. For example, bundling of services are commonly found in vacation packages: bundles of airline tickets with hotels and car rental, as well as TV channel subscriptions.

This work focuses on the bundle design [19] problem, which is a well-studied problem (see related work in Sect. 23.2.1). The bundle design problem in this chapter is concerned with designing bundles of attractions for a large theme park. Typically in a theme park, there are many attractions not all of which are interesting to a given visitor—much like not all TV channels are appealing to a particular viewer. Hence, it is an interesting problem to be able to offer a given visitor an appropriate bundle of attractions that fits his/her profile. This bundle is then sold as a ticket comprising the subset of attractions that can be redeemed by the ticket holder.

Our work falls under a larger category of consumer analytics for the hospitality industry. It thus offers a methodology for the travel industry in terms of generating good bundle options for tourists in general. With the increasing number of both domestic and international travellers every year, there is a need to better understand demands of customers in order to better serve and provide a more enjoyable experience. In the past, many business decisions were made based on the "gut feeling" of decision makers, which might not always be the best as important information is not usually readily available. With the advancement of technology, especially with the emergence of data analytics, these critical business insights may be obtained from data, which allows these decision makers to make better business decisions in a timely manner.

Our goal is to develop strategies for designing bundles for profit maximization with a quality of service (QoS) constraint for the park operator. The target bundle design problem has unique characteristics differentiating from the existing bundling problems in terms of profit. The profit gained from a customer/visitor depends on his/her visit preferences, thus per-visitor profit varies among the visitors who bought the same bundle with the same price. While designing bundles, we explicitly consider the existence of multiple user classes/segments, and disparate bundling strategies typically exist for different user classes. The identification of user classes is achieved through analysing historical consumption patterns when customers were free to choose among all available services. These consumption patterns turn out to be the major difficulty in designing profitable bundles, as the perceived attractiveness of a particular service usually depends on the other included services in the bundle. The QoS constraint is used to maintain a certain level of this interactive attractiveness while designing profitable bundles.

We propose a data-driven approach benefiting from recommender systems (RSs) to offer personalized bundles of services. RSs refer to a broad range of software systems or tools that provide suggestions for items that a user might be interested in consuming [40]. The underlying techniques of RSs aim to produce high-quality prediction on ratings users might have for potential candidate items. To achieve such an objective, it is essential to have ways to reliably predict users' own preferences, which usually implies that a RS must have access to either a user's past choices or consumptions. Considering a user's own preferences as basis, a RS can be designed to generate its predictions using either other user's choices (the collaborative filtering approach [50] which is discussed in Sect. 23.2.2) or just candidate item's description/content (the content-based filtering approach [53]). Given the huge online footprint of users, service (e.g. retail and entertainment) providers have access to large amounts of user service consumption records. With such information, highly effective personalized RSs can be created for a wide variety of items including but not limited to movies, books, music, research papers and even friendships.

The target bundle design problem relates to the three-tier business model shown in Fig. 23.1. Examples of such business models include cable networks and leisure parks. The first is the tier of customers, who consume services from the given bundle based on their own preferences and limitations, like leisure park visitors; second, the tier of service providers who rent resources from the operator and provide services to customers, like the service provider of a roller coaster; and third, the operator, who

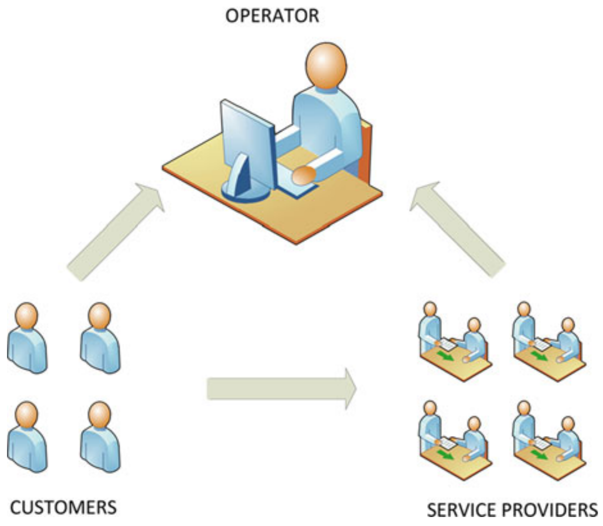


Fig. 23.1 A three-tier business model including customers, the service providers and the operator

Table 23.1 An example of four visitors who bought a bundle of four attractions where each attraction is managed by a service provider (+: visited, −: unvisited)

	a_1	a_2	a_3	a_4	Price (p)	Cost (c_j)	Profit ($p - c_j$)
v_1	+	−	−	+	40$	17$	23$
v_2	+	+	−	−	40$	13$	27$
v_3	−	+	−	−	40$	8$	32$
v_4	−	+	+	+	40$	27$	13$

owns the resources required for running services (e.g. land for physical services, or bandwidth for digital services) like the owner of the whole leisure park. To receive services from providers, customers can either pay the full rate directly to providers or they may purchase a pre-constructed bundle containing that service from the operator. When purchasing a bundle, a customer needs to pay the bundle price upfront (which is usually discounted from the total bundle value), but does not need to pay again for any service included in the bundle. As customers do not pay providers directly, providers will seek reimbursement at a pre-determined rate from the operator using verifiable service records. Table 23.1 gives a numerical example of the interaction between operator, services providers and visitors. Consider that a $p = 40\$$ bundle of four attractions are bought by four visitors. Whenever one of these visitors visits a particular attraction, a predetermined cost occurs to the operator and assumes that these four attractions cost (c_j) 17$, 13$, 8$ and 27$, respectively. Thus, although each visitor bought the same bundle with the same price, the operator's profit ($p - c_j$) from each visitor varies, i.e. 23$, 27$, 32$ and 13$.

We are interested in addressing this problem of designing service bundles for the operator. The operator receives income from two sources: (1) the fixed income from renting resources to service providers, and (2) the variable income from selling bundles. As the operator receives full payment for the bundle upfront, and only needs to reimburse a service provider if a customer uses that service, the operator actually enjoys additional income if a customer buys the bundle but utilizes relatively few services. Thus, solely from the perspective of the operator, having bundles of unpopular services seems like a profitable strategy. However, from the service providers' perspective, each attraction should be visited as much as possible so that each service provider who is responsible from each attraction can make a high profit. Otherwise, the service providers will not be willing to rent the attractions considering that they will not be able to reach their expected profit. Consequently, the operator will face the similar problem of making limited profit or losing money due to the issues of the service providers. Yet, both the operator's and the service providers' financial situation mainly depend on the park visitors. If the park visitors enjoy their time in the park through trying various attractions, each attraction can have a profitable business. This results in an increasing interest of renting the attractions by the service providers, which help the operator to make more income from the attractions' rents. This means that when constructing a service bundle, the operator may not simply include all the unpopular services, as the customers will simply refuse to buy the bundle if it is not attractive enough. We view this requirement as the Quality-of-Service (QoS) constraint the bundle needs to satisfy.

After formally formulating the bundle design problem that provides the mathematical foundation for quantifying the trade-off between operator profit and QoS constraint, we make two major contributions in designing the RS for the bundle design problem. First, we design a *static* segment-specific bundling strategy, in which a fixed bundle is constructed for a given customer (or a customer group) according to the segment s/he belongs. The static bundling falls into the category of the traditional bundle design. Having a certain level of flexibility of the bundles can make the bundling idea even more attractive for the customers. Thus, secondly, in order to grant even more flexibility to the consumer, we propose a *dynamic* bundling strategy where a customer has the option to choose to *trade in* any services s/he does not intend to utilize in the current bundle, and receive an alternative replacement which s/he may choose to keep or skip.

The remainder of the chapter is organized as follows. Section 23.2 provides reviews both bundle design and collaborative filtering. The bundle design problem studied here is explained in Sect. 23.3. The design details of the data-driven bundle design approach are provided in Sect. 23.4. Section 23.5 presents computational results using a real-world leisure park data. The conclusions and suggestions for future research are discussed in Sect. 23.6.

23.2 Background

The main goal of *"bundling"* is to deliver more profitable products in bundles than selling each of the products in a separate way. *Collaborative filtering* is a field of recommender systems. It is motivated to help group preferences. If two users have common preferences on certain items, their preferences are expected to be similar on other items and thus inform predictions.

These two subjects are now related in the literature. Recommender systems have also been used to address the bundling problems. In [59], a recommender system was devised to offer k best packages by extending the item-based recommendation idea using approximation algorithms. A graphical model was proposed to detect the consumers' unknown preferences that are used to predict user-specific best possible TV channel bundles in [17]. A bundle recommender system that reduces the item set first for discovering an optimal bundle for e-commerce was designed in [66]. Crowdsourced data was incorporated to deliver personalized travel packages in [61, 62]. In [9], a collaborative filtering based recommender system that utilizes personalized demand functions and price modelling was developed, focusing on both suppliers and consumers. Additionally, various studies [30, 51] were performed to recommend travel packages from a set of existing ones.

This section covers some fundamentals on bundling and collaborative filtering as related to our present work.

23.2.1 Bundling

The purposed goal of “bundling” comes with some advantages [44] including price discrimination, cost saving and potential entry deterrence. The goal is aimed to be achieved by attracting and satisfying consumers in terms of experiencing bundles and their costs. These objectives have been investigated under three perspectives in the literature [55]. These perspectives relate to *suppliers*, *consumers* and *competition*.

Suppliers seek ways to decrease all kinds of costs such as inventory costs [18] and marginal costs [54] while increasing profits and sales. In [20], the profitability of bundling was examined for a supplier that leads the complete market, considering the negotiations between the intermediaries and a competing firm. Another profitability work [16] was carried out under different conditions including the consumers’ preferences, product diversity and rival existence. In [58], the supplier selection problem was studied from a bundling perspective for notebook manufacturers in Taiwan. The goal is to determine the best possible bundle of suppliers for notebook production depending on various cost and quality measures. The effects of heterogeneity of the bundled products with a risk analysis were investigated in [44]. In [37], a constrained-based adaptive bundling strategy is proposed to offer product bundles by taking into account the newly introduced constraints/rules and the changes on products’ availability. Bundles of information products were analysed from the suppliers’ perspectives with the products’ complementarities and substitutabilities in [36]. In [34], bundles of sensor data, referring to applications of the “Internet of Things”, coming from multiple suppliers were offered with a pricing strategy. Pricing strategies on mobile tariffs as bundles were discussed in [12].

Consumers look for bundles to pay less when the bundled products are separately bought and to consider bundles that are interesting in terms of the correlation/complementarity between the bundled items. The way of discounts offered by bundles of automobiles with optional extras was studied considering the consumers’ feedback in [25]. In [1], the factors affecting the consumers’ decisions on buying a particular bundle were studied under the light of the bundle size, the bundles’ uniqueness and similarities. The relation between the decrease on the consumers valuations for the information goods and the number of goods was examined in [22]. A consumer preference analysis on choosing between bundles and single products was performed in terms of search and assembly costs in [24]. The consumers’ evaluations were analysed on the individual products of a bundle considering price discount and product complementarity in [46]. The bundles in the telecom industry of Turkey were studied to understand the actions of consumers on buying bundles and their future bundle choices in [52]. In [13], the consumers’ behaviour was reviewed on their food bundle preferences focusing on fruits and vegetables, relating to their health issues.

Competition [8] is another factor that needs to be taken into account if required. From this perspective, bundling studies are studied for the *monopolist*, *duopolist*, *oligopolist* and *perfect competition* environments. Monopoly refers to the markets where there is only one supplier for a particular product. For the duopolist markets, there are two competing suppliers for a single product. In the case of oligopoly, more than two strong yet limited number of suppliers are available. The perfect competition occurs when many suppliers and consumers exist. To give a number research works concentrating on the competition aspect, in [54], bundle design and pricing were studied for the monopolist settings. In [6], the effects of bundling on social welfare were studied for monopolists. In [15], an equilibrium theory is studied for the profitability of the bundles considering the duopoly and perfect competition markets. For the oligopoly markets, the effects of discounts on the bundled products considering the products' interrelations were analysed in [21]. An analysis for two-good bundle pricing is performed under oligopoly in [45].

Referring to the form of bundling [49], the studies could be classified as pure or mixed bundling [35]. Pure bundling is for the firms that sell only bundled products while mixed bundling [35] is valid when a firm sells both bundled and separate products. For the case when there is no bundling, "unbundling" is the term used in the literature.

In this chapter, we study the bundle design problem of a multi-product monopolist (no competition) for pure bundling, yet also applicable to mixed bundling, without bundle-related supplier costs, where product interactiveness are taken into account.

23.2.1.1 Item Interactivity

One of the critical aspects for bundle design is interactiveness of items. Interactiveness is considered in three ways, namely complements, substitutes and independent. Among them, complementarity and substitutability are studied mostly as interactiveness indicators. The traditional approach to specify whether two products are complementary or substitutable is to check the sales of products regarding their price changes. For two complementing products, a decrease in the price of one product is expected to cause an increase in the sales of the other product. For two substitutable products, however, increase in the price of one product is expected to cause an increase in the sales of the other product. In other words, the complementary products are interesting together for the consumers while the substitutable products are expected to be sold as alternatives [64]. For instance, a bike and a bike tyre can be considered complementary while two different bikes are substitutable since either one of them could be bought. Yet, changes on the prices and sales are unable to provide an ultimate criterion to talk about complementarity and substitutability for a given pair of products [47]. It is possible to see some products that are complementary for some consumers while they are substitutes for some others [28]. Following the same bike example, a road bike and a mountain bike can be complementary if a consumer has a particular interest of using them for different purposes, i.e. driving on a regular road or off-road driving. These bikes can be substitutable for another consumer if the only purpose is riding a bike.

The products that are complementary for some consumers, while substitutes for some others, require more complex bundling approaches than in the case where it exists a strict interactiveness difference. The bundling problem we are working on has an additional characteristic than when complementarity and substitutability can be favourable or unfavourable, is unknown. This makes our problem even more challenging. However, the bundles of complementing products (a detailed discussion is given in Sect. 23.5.1) are usually assumed more attractive to the customers than the bundles of substitutable products in the existing studies.

23.2.1.2 Size of the Bundles

Although the effects of bundles' sizes are a crucial subject to research, most of the bundling studies focus on the cases where bundle size is only two [14, 32]. In spite of the strong conclusions derived in these papers, working on the smallest bundles limits the applicability of these studies to the real-world scenarios. Unlike these studies, thousands of information goods are considered to form bundles in [7]. They investigated various topics including market segmentation, interactiveness and profit analysis regarding the bundle size.

Related to this chapter, in [33], the bundling operation is handled by generating a Markov Random Field (MRF) based on the visit transition frequencies of a given dataset. MRF is an undirected graphical model with Markov property. For our target problem domain, each node refers to an attraction and edges between nodes indicate the relation between attractions. Attractions that are disconnected or without a directly connecting edge are considered independent or unrelated. In other words, unconnected attractions are the ones which are not visited together or visited together a few times only. A MRF generated based on a relatively large historical visit dataset is able to reflect popular attractions and their level of popularity to be visited together with other attractions. Weights calculated for attraction subsets indicate how strong this popularity level is. These weights are also used to evaluate the popularity of each attraction when a set of certain attractions available in the form of conditional probabilities. In the aforementioned study, this information is utilized as the indicator of attractions' attractiveness. While attractiveness level shows the probability of an attraction being visited, it is used to calculate the cost that incurs to the main leisure park operator. Here, suggesting highly attractive bundles means that visitors will be highly satisfied while the operator will pay a large amount to the service providers and vice versa. Due to this inverse relationship between attractiveness and cost, the problem is designed as the knapsack problem [31]. The only constraint is set as the attractiveness level so that a resulting bundle should have a certain attractiveness level at least. Since the attractiveness of attractions is not constant or fixed but varies depending on the other attractions included in the bundle, the knapsack problem is considered with interactiveness.

23.2.2 Collaborative Filtering

Collaborative filtering (CF) [50] became a popular field of research mainly after the Netflix challenge [10]. The aim of this competition was to predict a set of users' preferences on a group of movies in the form of ratings/scores when limited rating data is available. Similarly, the Amazon.com [29] like datasets have been used for evaluating various CF methods on customer-item matrices involving the customers' item preferences. In those datasets, there is no dependent relation among both the users and items. Differently, in our case, visitors can be dependent when they move in groups and attractions are highly correlated which affects both revenue and QoS. In addition, one of the tested data forms here consists of both ordinal and availability information together while the current CF literature focuses on single-aspect data like movie ratings as in Netflix.

The motivation behind CF is related to the similarity levels between different users with respect to their preferences. If two users have common preferences on certain items, their preferences are expected to be similar on other items. Thus, user-based similarity is the straightforward choice for CF. Determining similarities between items [41] have also been used to perform predictions. A prediction process is concerned with either *matrix completion* or *cold start* [42]. The matrix completion task is targeted when all the users and items are known at some level. In other words, each user should have his/her score at least on one item and each item should be evaluated at least by one user. In the cold start case, the goal is to make predictions for unknown or new users and new items.

23.2.2.1 Memory-Based and Model-Based Collaborative Filtering

Existing CF algorithms are studied under two categories including *memory-based* CF and *model-based* CF. The memory-based methods predict unknown elements directly using available sparse data. The nearest-neighbour algorithm is the primarily studied memory-based approach. The idea is to determine missing matrix values by using a predetermined number of similar users or items.

Model-based algorithms build a model which approximates to a given matrix. Matrix factorization (MF) is the primary technique used for this purpose, mostly via optimization. The memory-based CF techniques are very effective in spite of their simple designs. However, they are likely to deliver worse performance than the model-based CF approaches when the data provided is highly sparse [50].

Matrix factorization is basically a mathematical process to derive two matrices from a given matrix where the multiplication of these two matrices relates to the original matrix. Singular value decomposition (SVD) [11] is one of the well known matrix factorization methods used in various CF studies. The main issue with this kind of pure mathematical approaches is the requirement of having a full matrix. Thus, a pre-processing step is needed to complete a sparse matrix first. The other family of MF algorithms define the factorization process from an optimization perspective. The goal is to find two matrices that gradually approximates to a given matrix. While earlier methods like SVD deliver a perfect factorization, the second group of methods usually provide near-perfect or near-optimal factorization. Thus, they are also referred to as *"matrix approximation"*.

23.2.2.2 Matrix Approximation and Factorization Methods

The matrix approximation methods have been used on two different types of matrices. The first matrix type involves both negative and positive real values. The basic strategy to address matrix completion is minimizing the sum of squared error between the original matrix and the predicted matrix. Gradient descent is used to perform this optimization task by updating two factorized-matrices in an alternating manner. Besides this basic implementation, Maximum-Margin Matrix Factorization (MMMF) [48] emulates SVM by considering multiple classification tasks and accommodating the hinge loss. Semidefinite programming is used to realize the whole optimization process. Due to the scalability issues of semidefinite programming, a gradient-based optimization technique was proposed to speed up MMMF in [38]. Unlike these methods, probabilistic MF [43] optimizes the posterior distribution on the factorized-matrices. In order to deliver better factorization performance, *tensor factorization* [56] has been studied to add content-based user or item specific data as additional dimensions to the traditional two-dimensional matrices. The second matrix type only allows non-negative values. The idea is to further analyse factorization results by applying the non-negative MF (NMF) algorithms [65]. The analysis part is related to the hidden/latent factors characterizing users and items, revealed by MF. In the non-negative case, these factors, which are also non-negative, are directly interpretable due to their additive nature. Although the factors from the first matrix type are still useful, there is no a systematic way to specify what each factor refers to. In terms of solution strategies, we evaluate a number of memory-based and model-based CF algorithms on two different matrix forms. Besides that, we suggest a way to combine the strengths of the tested methods when they are used on one of the matrices to deliver overall better performance.

23.3 Problem Statement

In this section we will define the problem we are dealing with, in which we have multiple customer classes. However, to understand the concept we rely on first explaining the single-cluster model which can then be generalized to a multiple-cluster one so we start with this one first.

23.3.1 Single-Cluster Model

Let $\mathbf{N} = \{1, 2, \ldots, N\}$ be the set of available services. Let $K < N$ be the desired bundle size. The objective of our bundle design problem is to maximize the expected profit for a typical bundle customer:

$$\max \sum_{i \in \mathbf{N}} r_i(\mathbf{x}) x_i, \tag{23.1}$$

with the QoS constraint that the overall bundle attractiveness must be at least c:

$$\sum_{i \in \mathbf{N}} a_i(\mathbf{x}) x_i \geq c. \tag{23.2}$$

Finally, to ensure that the bundle size is correct, we have:

$$\sum_{i \in \mathbf{N}} x_i = K. \tag{23.3}$$

In the above formulation, x_i is a binary decision variable: set to 1 if service i is included and 0 otherwise. $\mathbf{x} = \{x_1, \ldots, x_N\}$ is the vector of all decision variables $x_i, i \in \mathbf{N}$. The function $r_i(\mathbf{x})$ gives the expected profit earned by the operator when the bundle is $\{i | x_i = 1, i \in \mathbf{N}\}$. Following earlier definitions, this expected profit function can be computed as:

$$r_i(\mathbf{x}) = P(S_i = 0 | S_1 = x_1, \ldots, S_N = x_N)\ u_i, \tag{23.4}$$

where $P(S_i = 0 | S_1 = x_1, \ldots, S_N = x_N)$ is simply the probability that a typical bundle customer would choose NOT to utilize service i, and u_i is the fee the operator needs to pay service provider i if the customer uses the service.

The attractiveness of service i can be similarly defined as:

$$a_i(\mathbf{x}) = P(S_i = 1 | S_1 = x_1, \ldots, S_N = x_N)\ v_i, \tag{23.5}$$

where the first component of Eq. (23.5) is the probability that a customer would choose service i, and v_i is the associated *value* it would bring to the customer. Note that for both cases, $P(\cdot)$ needs to be computed from the historical data.

Both Eqs. (23.4) and (23.5) are nonlinear; to linearize these two sets of equations, we can enumerate all possible bundles of size K as set $\mathbf{Y}_K$, and define new decision variables y_j to take on the value of 1 if bundle $Y^j \in \mathbf{Y}_K$ is chosen, and 0 otherwise. Let Y_i^j be 1 if service i is included in bundle j; Eqs. (23.4) and (23.5) can then be rewritten as:

$$r_i(Y^j) = \begin{cases} 0, & \text{if } Y_i^j = 0 \\ P(S_i = 0 | S_1 = Y_1^j, \ldots, S_N = Y_N^j)\ u_i, & \text{o/w}, \end{cases} \tag{23.6}$$

$$a_i(Y^j) = \begin{cases} 0, & \text{if } Y_i^j = 0 \\ P(S_i = 1 | S_1 = Y_1^j, \ldots, S_N = Y_N^j)\ v_i, & \text{o/w}. \end{cases} \tag{23.7}$$

With (23.6) and (23.7), the bundle design problem (23.1)–(23.3) can then be linearized as:

$$\max \sum_{Y^j \in \mathbf{Y}_K} y_j \sum_{i \in \mathbf{N}} r_i(Y^j). \tag{23.8}$$

s.t.

$$\sum_{Y^j \in \mathbf{Y}_K} y_j \sum_{i \in \mathbf{N}} a_i(Y^j) \geq c, \tag{23.9}$$

$$\sum_{Y^j \in \mathbf{Y}_K} y_j = 1. \tag{23.10}$$

The final constraint ensures that exactly one bundle is selected.

The above optimization model looks deceivingly simple, yet it is non-trivial to solve, as the set $\mathbf{Y}_K$ is combinatorial, and the number of decision variable y_j will grow exponentially in K.

23.3.2 *Multi-Cluster Model*

As stated earlier, one major contribution we make in this chapter is the incorporation of multiple customer classes. To accommodate this, we only need to modify Eqs. (23.6) and (23.7). Instead of having only a single set of joint probability distribution on service selection ($P(\cdot)$), we will now have multiple sets of probability distributions.

23.4 A Data-Driven Approach

The primary contribution of our chapter is the design and implementation of a data-driven approach for an operator managing a collection of services. More specifically, recommending bundles of services using consumer redemption data should maximize expected profits, while obeying QoS requirements (characterized by a lower bound on attractiveness or customer valuation). The mathematical formulation of the proposed method is already presented in Sect. 23.3.

In this section, we introduce and compare approaches for extracting multiple user classes from a real-world dataset. As the size of the mathematical formulation grows exponentially, we use an efficient and effective greedy heuristic for solving the bundle design problem. To provide a concrete context to the methodological discussion, we develop our approaches based on a real-world dataset (details are presented in Sect. 23.5) collected from a leisure park operator who leases out its land parcels to multiple attraction operators (i.e. service providers).

The proposed approach involves two important stages: (1) customer segmentation: based on historical data, infer multiple classes of customers who have inherently different preferences over provided services, and (2) segment-specific bundle recommendation: based on segment-specific parameters (in the form of probability function $P(.)$), solve individual bundle design problem using a greedy heuristic (to be explained). For the second stage, we propose two variants. The first variant recommends *static* bundles, referring that customers have to commit to the content of the purchased bundle. The second variant allows bundles to be *dynamic*, which means that a customer can request part of the bundle to be replaced by new recommendation. The greedy heuristic is based on an earlier work [33] on similar problem. For the dynamic variant, a memory-based CF method, i.e. nearest-neighbour and a model-based CF algorithm (CofiRank) are applied to two matrices consisting binary visit information and time information. Figure 23.2 visualizes the complete workflow of the proposed strategy.

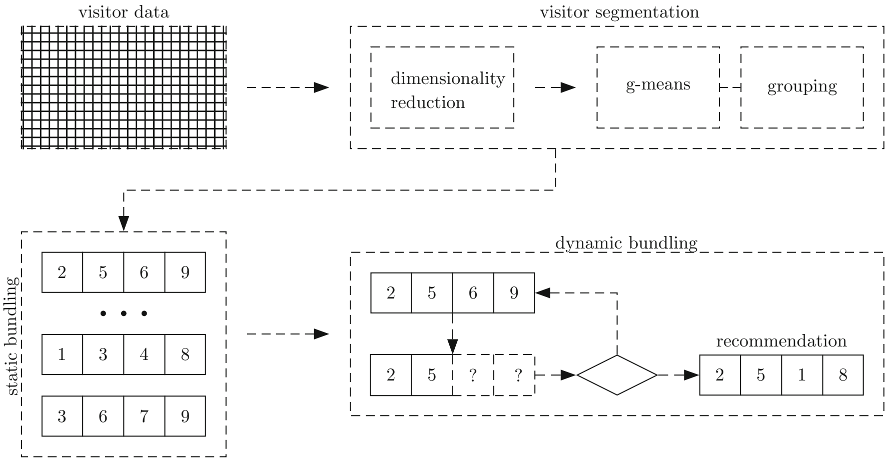


Fig. 23.2 Workflow of the proposed data-driven approach with three steps including visitor segmentation: to determine different types of customers, static bundling: to offer attractive and profitable attraction bundles and dynamic bundling: to suggest on-the-fly changes on the bundles

23.4.1 Visitor (Customer) Segmentation

Visitor segmentation is a critical aspect of a bundle design process. For instance, consider a tourism agent who would like to sell travel packages for visiting multiple cities. The first step is how to determine these packages. In principle, each package should be designed such that they can meet different customers' needs. There can be a group of customers who are interested in nature while another group of customers are interested in history. Designing a single package to satisfy these two groups of customers is likely to be an unsuccessful strategy considering the strict differentiation between customers' preferences. Instead, providing a travel package with visits relating to nature and a travel package involving visits of historical sites is expected to more satisfactory for these customers.

In order to detect such variations on customers, some prior information regarding consumers is required. A questionnaire performed on a large group of people shows that customers' explicit preference information, historical order data and online browsing history are some examples that can be used to specify customer groups or can be used to segment customers. In this study, we use visit data of earlier visitors in the form of visitor $\times$ attraction matrices such that each row refers to a visitor and each column represents an attraction. This data indicates which attractions are visited by each visitor with day/time information. Although directly using such data would be useful, it might be hard to process depending on its size. Besides that, it is likely that data of a large number of visit information has some noise.

Singular Value Decomposition (SVD) [11] is applied to address both data size and noise issues. SVD is a widely used MF [26] technique particularly in the CF field both to approximate a given matrix to its smaller-rank version and to provide representative latent (hidden) factors. Matrix rank here refers to the column rank. SVD is capable of decreasing the number of columns of a corresponding matrix allowing to perform various matrix manipulations easier due to having relatively smaller-sized data. Thus, SVD is also stated as a dimensionality reduction approach. The other capability of SVD, being able to provide latent factors, is helpful to eliminate noise in the data. In particular, visitors can be characterized more efficiently by using these latent factors rather than directly using their attraction visit information.

As shown in Eq. (23.11), applying SVD to a matrix $\mathcal{M}$ results in three matrices, i.e. U, V and Σ. U and V matrices are defined as left and right singular vectors while the diagonal Σ matrix provides sorted singular values. These U and V matrices represent rows and columns of $\mathcal{M}$, respectively. The Σ matrix informs about the importance of resulting latent factors that decreases for each subsequent dimension, e.g. the first latent factor is more significant than the second one.

$$\mathcal{M} = U \Sigma V^T \tag{23.11}$$

SVD is practical to achieve an efficient similarity analysis between the elements of a matrix. However, the primary reason using it here is to take advantage of dimensionality reduction for the sake of better and faster clustering for segmentation. After applying SVD to a customer's visit data, the produced U matrix is used as the matrix representing customers. The clustering on U is performed by the Gaussian-means (G-means) algorithm [23]. It is a clustering algorithm using k-means where k is automatically determined. G-means incrementally applies k-means for larger k values if a resulting cluster doesn't belong to Gaussian distribution. The process starts with a single cluster, $k = 1$ or $k > 1$ if there is prior knowledge, and k is iteratively increased until all the clusters are from Gaussian distribution. The distribution check is operated by applying the Anderson-Darling statistic [4]. If this test determines non-Gaussian clusters, the corresponding clusters are divided into 2 using principal component analysis and k-means is re-applied. The overall customer segmentation process is explained in Algorithm 1.

Algorithm 1: Two-step clustering strategy for customer segmentation, where $\mathcal{M}_{v,a}$ is a customer-attraction matrix

```
2  Data extraction: $\phi \rightarrow \mathcal{M}_{v,a}$ ;
4  Dimensionality reduction: SVD($\mathcal{M}_{v,a}, r$) $\rightarrow U_{v,r}\,\Sigma_{r,r}\,V^T_{r,a}$ where $r <= min(v, a)$ ;
6  Customer-based clustering: $C$ = G-Means($U_{v,r}$) ;
8  Calculate transition frequency matrices: $T_{c_i,aa}$ for each $c_i$ from $C = \{c_1, \ldots, c_n\}$ ;
10 Measure cluster similarities based on these $T$ matrices: $sim(T_{c_i,aa}, T_{c_j,aa})$ ;
12 Combine the clusters with a certain similarity level: $sim(T_{c_i,aa}, T_{c_j,aa}) \geq 0.5$ ;
```

The same MRF approach studied in [33] is pursued, now for each customer segment. Thus, it is expected that each segment differs in terms of their visit transition frequencies. For this purpose, the clusters determined by G-means are processed to generate transition matrices that are converted to vectors afterwards, where the frequency values are normalized as percentages. These vectors are compared to a similarity metric for detecting similar clusters. The similarity check is realized either using the Cosine similarity (Eq. 23.13) or Pearson Correlation Coefficient (Eq. 23.13). The Cosine similarity compares two vectors based on the angle between them while the Pearson Correlation coefficient considers average values in addition to Cosine. For the first metric, the similarity values change between 0 and 1 where 1 refers to the most similar cases. For the second metric, the similarity level varies between -1 and 1 where 1 refers to the strong positive correlation and -1 indicates strong negative correlation. In the equations, v_i refers to the customer transition vector, $a \in A$ is an attraction and $r_{v_i,a}$ is the transition frequency value for a customer-attraction pair.

$$\begin{aligned} Cos.Sim(v_1, v_2) &= \frac{\vec{v_1}.\vec{v_2}}{\|v_1\| \times \|v_2\|} \qquad (23.12) \\ &= \frac{\sum_{a \in A} r_{v_1,a} r_{v_2,a}}{\sqrt{\sum_{a \in A} r^2_{v_1,a}}\sqrt{\sum_{a \in A} r^2_{v_2,a}}} \end{aligned}$$

$$Corr.Sim(v_1, v_2) = \frac{\sum_{a \in A} (r_{v_1,a} - \overline{r_a})(r_{v_2,a} - \overline{r_a})}{\sqrt{\sum_{a \in A} (r_{v_1,a} - \overline{r_a})^2}\sqrt{\sum_{a \in A} (r_{v_2,a} - \overline{r_a})^2}} \tag{23.13}$$

23.4.2 Static Recommendation

After all the customers are segmented, a static bundle is recommended for each visitor segment. The motivation studying static recommendation is to deliver generally acceptable bundles. When a new customer arrives, s/he should be able to find a reasonably satisfactory static bundle to start visiting the corresponding leisure park. The per segment static bundles with varying sizes are generated by applying a simple yet effective greedy construction heuristic [33]. The greedy heuristic iteratively adds an attraction with at least c/K attractiveness level and the lowest cost to generate a bundle of a size K, i.e. involving exactly K attractions. The greedy heuristic initially adds a single attraction via Eq. (23.14) respecting Eq. (23.15). The remaining $K - 1$ attractions are selected using Eq. (23.16) while satisfying Eq. (23.17). It was shown that this heuristic is capable of delivering near-optimal solutions. The bundle tickets at this point are recommended as either fixed bundles or flexible bundles. The fixed ones belong to the customer segments which are too large considering all the segments. These segments are detected by using the interquartile range (IQR). IQR is calculated as $IQR = Q3 - Q1$, the difference between the third, $Q3$, and first quartiles, $Q1$. The segments larger than $Q3 + (1.5 \times IQR)$ are considered upper outliers.

$$Y = \arg\max_{Y_i \in X} \sum_{y_i \in Val(Y_i)} R_{y_j} P(y_i | X \backslash Y_i = 0) \tag{23.14}$$

$$\frac{\Gamma_{min}}{K} \le P(Y_i = 1 | X \backslash Y_i = 0) \tag{23.15}$$

$$\arg\max_{Y_i \in X \backslash Y} \sum_{Y_j \in \hat{Y}_i}^{y_j \in Val(Y_j)} R_{y_j} P(y_j | X \backslash \hat{Y}_i = 0) \tag{23.16}$$

$$\frac{(|Y| + 1)c}{K} \le \sum_{Y_j \in \hat{Y}_i} P(Y_j = 1 | X \backslash \hat{Y}_i = 0) \tag{23.17}$$

23.4.3 Dynamic Recommendation

The idea of dynamic recommendation is to make online suggestions by offering changes on an existing ticket. Figure 23.2 provides a simple example. In the given scenario, a visitor who bought a bundle ticket (2, 5, 6, 9) decides to make changes on the ticket. This visitor either already visited attractions 2 and 5 or planning to visit these two attractions. Dynamic bundling aims at suggesting other attractions in exchange of attractions 6 and 9 which are not preferred to visit by the visitor. In the given example, (6,9) is swapped with (1,8). It should be noted that in this operation there is no obligation regarding 1-to-1 exchange meaning that one or more than two attractions can also be offered to change with two attractions.

As shown in the given example, dynamic recommendation is the next step of the static recommendation on the bundles. However, it could be argued that the dynamic recommendation might not be a critical component since the bundles are already personalized with the prior operation of customer segmentation. Although the argument sounds right, it is not completely correct since the primary idea of dynamic recommendation is to increase the level of personalization on the bundles. The level of personalization could be increased due to the generalization effect of the customer segmentation. From the customer segmentation perspective, the optimal segmentation would be considering each customer as a separate segment while the worst segmentation would be considering all the customers in a single segment. The customer segmentation approach can provide a balance between the optimal segmentation and the worst segmentation. The optimal segmentation could have been, of course, used to resolve this issue yet it can obviously make the system impractical. For instance, if we consider a leisure park of just 10 attractions, the possible number of bundles is 1023 ($2^N - 1$ where $N = 10$) that is the optimal case which can easily confuse the customers. The customer segmentation handles this confusion at a large extent yet it is still a generalization due to the conflict between the personalization and practicality. The dynamic recommendation can give the opportunity of personalizing the bundles further.

Dynamic recommendation is useful not only for a high level personalization but also for addressing the issues occurring in real-time. To exemplify, a customer walks towards to an attraction but encounters with a long queue and decides that they do not want to wait, or a customer needs to leave the leisure park earlier than her/his planned schedule so to find an attraction which requires shorter time or a customer sees an attraction from outside and dislikes it. For such cases, a customer can easily change her/his decision on-the-fly with dynamic recommendation.

For addressing dynamic bundling, the idea of collaborative filtering (CF) is incorporated. CF is popular in making predictions or recommendations of items that are expected to be interesting for a particular customer. When partial customers' preference information on items is available, a CF method can efficiently predict a customer's preferences on the items s/he haven't seen or tried yet. For this purpose, a number of memory-based and model-based CF methods are utilized. The k-nearest neighbour (kNN) based prediction method with varying similarity metrics is employed as a memory-based CF algorithm. A MF algorithm, i.e. CofiRank, is applied as a model-based approach. The customer-attraction matrix either with binary visit information or time period data is used to test dynamic recommendation. The results from the first matrix provide dynamic ticket changes while the second matrix is useful to also suggest tickets with order-based information.

For kNN, cosine similarity is used to determine the most similar entries. The similarity measurement can be done either by comparing customer entries/rows or attraction entries/columns. Due to the data size we are dealing with, similarities are measured based on matrix columns. After calculating all the similarities, a prediction matrix is created using the weighted prediction measure [39] as shown in Eq. (23.18).

$$pred(v, a_i) = \overline{r_{a_i}} + \frac{\sum_{a_j \in N} Cos.sim(a_i, a_j) \times (r_{i,a} - \overline{r_{a_j}})}{\sum_{a_j \in N} Cos.sim(a_i, a_j)} \tag{23.18}$$

CofiRank [57] is inspired from MMMF [48]. It primarily aims at optimizing a rank-based loss function, i.e. Normalized Discounted Cumulative Gain (NDCG), Eq. (23.19).

NDCG@k is a metric that shows how well the top k items are predicted, besides that predicting an earlier item is more valuable for predicting an item that comes later, in the form of ranking. In other words, NDCG imitates the motivation behind search engines about finding the most relevant sites about a subject, first. The given equation provides a value between 0 and 1 where 0 refers to the case where all the predictions are wrong while 1 shows the case of a perfectly correct prediction. DCG@k is the actual measure indicating the quality of the prediction. It is divided by IDCG@k which is the perfect prediction case to normalize. DCG@k is calculated as shown in Eq. (23.20). $r_{i,j}$ is the score or quality of item j where i indicates the prediction order. $^1/_{log(i+2)}$ is the discount factor. If the initial items are correctly predicted, their contributions to the DCG will be higher. Thus, the objective is to maximize DCG@k so NDCG@k. Besides NDCG, the squared regression loss (Eq. 23.21) and ordinal loss (Eq. 23.22) functions are provided. The regression loss is the generally used loss function considering the differences between the predicted values, p, and the actual values, r. The last function focuses more on predicting the right order of items regarding their scores or qualities. CofiRank targets to minimize one of these loss functions ($L(UV, \mathcal{M})$) together with a trace norm based regularization element, as illustrated in Eq. (23.23). The optimization process accommodates a method manipulating U and V in an alternating manner for approximating to a given sparse or incomplete matrix. Due to the superior performance of the regression loss, only its results are reported.

$$NDCG@k = \frac{DCG@k}{IDCG@k} \tag{23.19}$$

$$DCG@k = \sum_{i=1}^{k} \frac{2^{r_{i,j}} - 1}{log(i+2)} \tag{23.20}$$

$$l(p, r) = \frac{1}{2}(p - r)^2 \tag{23.21}$$

Table 23.2 Tested collaborative filtering methods

Method	Details	Target matrix	Output
kNN-B	Column-based NN with Cosine similarity	Binary matrix	Dynamic bundling
kNN-T	Column-based NN with Cosine similarity	Time matrix	Dynamic bundling with ordering
kNN-BT	Column-based NN with Cosine similarity	Binary matrix + time matrix	Dynamic bundling with ordering
CofiRank-B	Regression loss	Binary matrix	Dynamic bundling
CofiRank-T	Regression loss	Time matrix	Dynamic bundling with ordering
CofiRank-BT	Regression loss	Binary matrix + time matrix	Dynamic bundling with ordering
kNN-B+CofiRank-T	Combined	Binary matrix + time matrix	Dynamic bundling with ordering

$$l(p,r) = \frac{1}{M} \sum_{r_i < r_j} C(r_i, r_j) max(0, 1 + p_i - p_j) \tag{23.22}$$

$$\min_{U,V} L(UV, \mathcal{M}) + \frac{\lambda}{2}(trVV^T + trUU^T) \tag{23.23}$$

Table 23.2 details all the experiments carried out to address dynamic bundling. Besides the experiments using the aforementioned CF algorithms for completing sparse binary and time matrices, a combined approach is additionally suggested. kNN-BT, CofiRank-BT and kNN-B+CofiRank-T fall into this category. The idea here is to apply a CF algorithm separately to binary and time matrices, then validating the time matrix by the predictions from binary matrix. Each element of a time matrix changes between 0 to a relatively large value representing the daily end visit time. A 0 means unvisited or “not-be-included” and the rest of the values show that corresponding attractions are included. Considering that space of possible values to be assigned, it is very likely to set values other than zero to an attraction which is not actually visited. In the case of binary matrix, a possible value is either 0 or 1 so the chance of false prediction is very limited in comparison to the case where time matrix is used. In order to alleviate this issue, the unvisited attractions detected from the binary matrix are used to remove the non-zero elements for those attractions in the predicted time matrix.

For comparison purposes, Association Rule Mining (ARM) [63] (a.k.a. Frequent Itemset Mining[2]) is used. ARM is a popular approach particularly in market basket analysis. The idea is to determine the items which are expected to be bought by a customer when s/he already bought a set of items. In other words, ARM looks for rules such that $A \rightarrow B$ where A is the set of existing items and B is the items expected to be included or interesting ($A \cup B = \varnothing$). ARM operates based on frequencies like MRFs but in a simpler way. Support and confidence are the two main measures to evaluate the quality of ARM rules. Support of a rule $A \rightarrow B$ is calculated as shown in Eq. (23.24) where $n_{A \cup B}$ shows the number of transactions when items A and B are bought together and N refers to the total number of transactions. Confidence is calculated using support as presented in Eq. (23.25). While support is a basic frequency calculation of a rule, confidence is a conditional probability, $P(B|A)$, indicating how strong a rule is. For dynamic bundling, attractions (to be) visited are the ones included in A while B are the ones to be offered as dynamic choices ignoring the attractions not to be visited. However, it should be noted that using ARM can be costly for large datasets. Suppose that there are m items. The total number of itemsets can be derived is $2^m - 1$. Exhaustive search requires $N \times (2^m - 1)$ checks to evaluate and detect all the rules. Increasing m values makes this process computationally expensive. Apriori [3] was introduced to efficiently tackle this issue. Apriori is able to quickly derive all the rules where each rule has a support level which is equal or higher than a predetermined value. The speed up provided by Apriori comes from this support level bound. Whenever it detects a rule that has a lower support than this bound, there is no need to check other rules containing the items from this particular rule. Thus, many infeasible solutions (the ones with low support) can be eliminated without even checking them and delivers a faster way of rule detection. For dynamic bundling, preferences on to-be visited attractions and not-to-be visited attractions are provided as prior information to ARM. ARM returns the item set or bundle that includes these prior preferences with the highest confidence.

$$support(A \rightarrow B) = \frac{n_{A \cup B}}{N} \tag{23.24}$$

$$confidence(A \rightarrow B) = \frac{support(A \cup B)}{support(A)} \tag{23.25}$$

As the final comparison algorithm, the aforementioned Greedy construction heuristic is used. In this case, dynamic bundling is thought as static bundling with prior information. This prior information consists of the attractions to be initially included and the attractions to be ignored from visitors' dynamic bundling requests.

23.5 Computational Analysis

We work with a large leisure park operator in Asia which manages 17 attraction providers. The attendance (or redemption) records of 22,287 visitors under an all-you-can-visit model (i.e. visitors would pay for a bundle that consists of ALL 17 attractions) were collected. Our focus in this section is to show how we can improve the bottom line of the operator through better bundle design.

Each attendance record is a tuple containing $\langle Timestamp, CardID, AttrID\rangle$, where $AttrID$ is the attraction ID, $CardID$ is the card id representing each visitor and $Timestamp$ shows when a particular attraction is visited. Sample attendance records are illustrated in Table 23.3.

Table 23.3 Sample attendance records of two visitors

Timestamp	CardID	AttrID
9/1/2012 09:33:11	308702168	4
9/1/2012 09:35:17	308599949	11
9/1/2012 10:12:34	308702168	6
9/1/2012 10:25:19	308599949	15

The data is then used to generate two matrices as exemplified in Table 23.4. The first is a binary matrix indicating whether an attraction is visited by each visitor. The second matrix (called the time matrix) keeps track of which attraction is visited at what time by which visitor. For the latter, time is sliced into 15-min intervals, where each time interval is denoted by an integer. In the given example matrices, visitor v_1 is shown as visited attractions a_2, a_3 and a_4 during time periods 1, 4 and 6, respectively.

For the dynamic bundling experiments, both binary and time matrices are used. In particular, one binary matrix and one time matrix are extracted from each visitor segment that is determined earlier. Initially, these matrices are processed to generate test instances in the form of k-fold cross validation. The cross-validation is done for the visitors/matrix rows. The visitors are first divided into k partitions. For instance, using the matrices from Table 23.4, twofold cross validation can be realized by dividing the visitors into two groups as (v_1, v_2, v_3) and (v_4, v_5, v_6).

Table 23.4 Redemption data representation example with six visitors and six attractions in two matrix forms: 0–1 matrix and time matrix from left to right (v: visitor, a: attraction)

	a_1	a_2	a_3	a_4	a_5	a_6
v_1	0	1	1	1	0	0
v_2	1	0	1	0	0	1
v_3	1	0	1	0	0	1
v_4	0	0	0	1	1	0
v_5	0	1	1	0	1	1
v_6	1	1	1	0	0	1

	a_1	a_2	a_3	a_4	a_5	a_6
v_1	0	1	4	6	0	0
v_2	6	0	3	0	0	4
v_3	11	0	7	0	0	5
v_4	0	0	0	4	9	0
v_5	0	10	12	0	18	20
v_6	3	9	5	0	0	13

For each partition, x entries are provided while the rest are removed in each row. The resulting test matrix has the full matrix entries for $k - 1$ partitions and only x entries for the visitors in the other partition. In total, k different test matrices are created by considering each partition with partial data. Again if we look at

Table 23.4 with $k = 2$ and $x = 3$, we keep the visit information of the first visitor group while keeping only $x = 3$ entries of each visitor for the second group. Table 23.5 shows the resulting matrices after performing the explained process. The same process is done for the second visitor group. As a result, two versions of binary and time matrices are generated for twofold. The goal here is to apply a CF method to complete these two matrices. In the matrices to be completed, available x values for each visitor refer to their partial preferences of the attractions visited and attractions not being visited. The missing values, denoted as "–", are the ones to be predicted to determine visitors' preferences on the related attractions. The attractions with the prediction of being visited are offered to swap with the attractions excluded as dynamic bundling. In a dynamic bundling scenario, the values different than zero refer to the attractions which are already visited and entries with zeros refer to the attractions requested to be swapped.

In the experiments, k is set to 5 (fivefold cross validation) which is small enough to test on the small-sized visitor segments and x is set as 3 to evaluate the dynamic bundling performance when the data is sparse enough.

For the clarity of the computational analysis, the notations used are listed in Table 23.6.

Table 23.5 Redemption data representation example for dynamic bundling with six visitors and six attractions in two matrix forms: 0–1 matrix and time matrix from left to right (v: visitor, a: attraction)

	a_1	a_2	a_3	a_4	a_5	a_6
v_1	0	1	1	1	0	0
v_2	1	0	1	0	0	1
v_3	1	0	1	0	0	1
v_4	–	0	–	1	1	–
v_5	0	–	1	–	1	–
v_6	1	1	–	–	0	–

	a_1	a_2	a_3	a_4	a_5	a_6
v_1	0	1	4	6	0	0
v_2	6	0	3	0	0	4
v_3	11	0	7	0	0	5
v_4	–	0	–	4	9	–
v_5	0	–	12	–	18	–
v_6	3	9	–	–	0	–

23.5.1 *Complementarity Analysis*

One of the most commonly studied aspects of bundles is the relation between bundled items. The level of relation between items can affect both the attractiveness and the price of a bundle. This relation can be evaluated from the perspectives of *substitutability* and *complementarity*. Substitutability refers to the products or services which have similar nature, e.g. two different roller coasters in the same leisure park. Complementarity is used for the goods with different characteristics such as a roller coaster and a 3D movie theatre, to be tried in complement to the overall experience. In principle, the bundles of complementing products are expected to be more attractive than the ones with the substitutable products, to the customers. However, it is not always clear to say whether two products are substitutable, complementary or totally independent so to talk about the bundles' attractiveness, as discussed in Sect. 23.2.1. For instance, a customer who likes roller coasters a lot would be more interested in a bundle of allowing to experience two roller coasters than the bundle of a roller coaster and a 3D movie theatre.

In [60], the degree of complementarity concept evaluating these relations is introduced from a pricing perspective. We consider the same concept using frequency ratios of service bundles ($f_r(b_i)$) derived from the given historical redemption data. A frequency ratio simply indicates the popularity of a bundle. For instance, a bundle of two products, i.e. $b = \{\text{p1,p2}\}$, appears together in the redemption data for 30% of the visitors, means that $f_r(b) = 0.3$. As given in Table 23.6, the degree of complementarity ($\mathcal{D}$) is calculated by using $f_r(b)$ of a given bundle divided by average $f_r(.)$ of all the subsets of b excluding itself. $\mathcal{D}$ changes between 0 and 1 where 1 shows the case when a bundle occurs as much as the average of its subsets. Thus, the values close to 1 indicate that the bundles with a high degree of complementarity while 0 indicates that some of the items decrease the degree of complementarity. Going back the two product bundle examples, $b = \{p_1, p_2\}$, consider that product p_1 appears in the 80% of the visits, i.e. $f_r(\{p_1\}) = 0.8$ and

Table 23.6 Notations used onwards

Notation	Definition
$\mathbf{N} = \{a_1, a_2, \ldots, a_N\}$	Set of available services or attractions to be visited
$\mathbf{B} = \{b_1, b_2, \ldots, b_{2^N-1}\}$	Set of all possible bundles, involving services from **N**, i.e. powerset of **N** excluding $\emptyset$: $\mathcal{P}(\mathbf{N})\backslash\emptyset$
$\mathbf{B}' \subseteq \mathbf{B}$	Set of unique bundles consumed before
$M = \lvert B'\rvert \leq 2^N - 1$	Number of unique bundles consumed before
$K = \lvert b_i\rvert \leq N$	Size of a bundle or number of services included in a bundle
V_s	Set of segments
$V_{s,i}$	Set of visitors in segment i
$f(b_i)$	Bundle frequency, i.e. number of times a set of services, composing b_i, appears together in $\mathbf{B}'$
$f(b_i)$ where $\lvert b_i\rvert = 1$	Single service consumption frequency, i.e. number of times a service is consumed before
$f_r(b_i) = f(b_i)/M$	Bundle frequency ratio
$f_r(b_i) = f(b_i)/N$ where $\lvert b_i\rvert = 1$	Single service consumption frequency ratio
$f_r(b_i) \times 100$ where $\lvert b_i\rvert = 1$	Single service consumption frequency in percentage, i.e. % visit frequency
c_{a_j}	Expected cost of service a_j, to the park operator
q_{a_j}	Attractiveness of service a_j
$\mathcal{C}(b_i) = \sum_j^{\lvert b_i\rvert} c_{a_j}$ where $a_j \in b_i$	Expected cost of bundle b_i, to the park operator
$\mathcal{Q}(b_i) = \sum_j^{\lvert b_i\rvert} q_{a_j}$ where $a_j \in b_i$	Attractiveness of bundle b_i
$\mathcal{P}_l(b_i) = \{b_{i1}, b_{i2}, \ldots, b_{iz}\}\backslash\{b_i\}$ where $f(b_{ij}) \geq l$	Subsets of services from b_i excluding b_i, with $f_r(b_i) \geq l$
$\mathcal{D}_l(b_i) = f_r(b_i)/(\sum_j^{\lvert\mathcal{P}_l(b_i)\rvert} b_{ij}/\lvert\mathcal{P}_l(b_i)\rvert)$ where $b_{ij} \in \mathcal{P}_l(b_i)$	Degree of complementarity

product p_2 is chosen for 70% of the time, i.e. $f_r(\{p_2\}) = 0.7$. In this case. the degree of complementarity of the bundle b is $\mathcal{D} = f_r(b)/((f_r(\{p_1\}) + f_r(\{p_2\}))/2) = 0.3/((0.8 + 0.7)/2) = 0.4$. Although both products are relatively popular when considered independently, the resulting bundle is unable to provide a similar level popularity as only appearing in 40% of the redemption transactions. It should also be noted that the highest possible value of $\mathcal{D}(b_i)$ is the product subset with the lowest frequency ratio, $f_r(.)$, as shown in Eq. (23.26). Thus, including a highly unpopular product set in a bundle substantially degrades the degree of complementarity of a bundle additionally with very popular products.

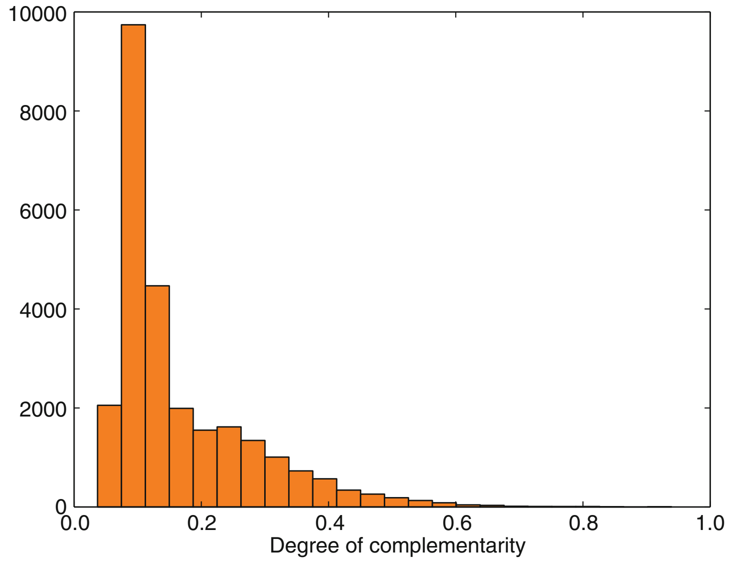
$$max(\mathcal{D}_l(b_i)) \leq min(f_r(\{b_{ij}\})) \text{ for } \forall\, b_{ij} \in \mathcal{P}_l(b_i) \tag{23.26}$$

Figure 23.3 indicates $\mathcal{D}_{0.01}$ of each bundle with respect to its subsets using the target redemption data. Without considering bundle size, the figure shows that majority of the attraction sets or bundles have a degree of complementarity around 0.1. However, there are still bundles with high degree of complementarity, e.g. 0.93, even though they are very rare. When we look at the bundles of the same size, the smaller sized bundles show a higher degree of complementarity. The underlying reason is about the number of subsets of a bundle, i.e. adding more attractions to a bundle increases the number of attraction subsets. Thus, $\mathcal{D}$ has tendency to become smaller yet more stable. Besides that, adding an attraction set with very low visit occurrence automatically decreases the visit frequency of the whole bundle, as just discussed.

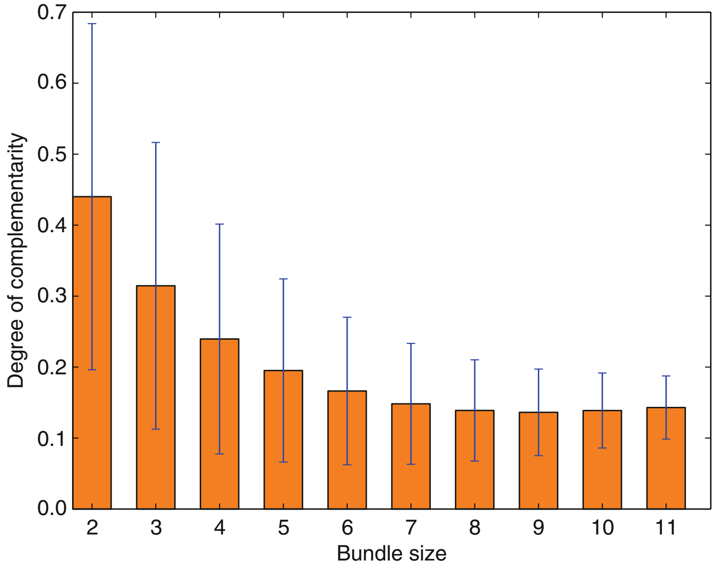
In addition, having many attractions in a bundle causes a more complex complementarity structure, requiring evaluating the relations between all the subsets of a bundle. That's why complementarity has been studied and analysed mostly for the bundles of two products only [5].

23.5.2 Visitor (Customer) Segmentation

The time matrix is extracted from the redemption data (Algorithm 1, line 1) for segmenting visitors. Each matrix row represents a visitor's attendance record. SVD is applied to the time matrix first for dimensionality reduction (Algorithm 1, line 2). One of the resulting matrices, i.e. $U_{v,r}$, is used in the next clustering step since it represents visitors. While v shows the total number of visitors, r can be set to a value between 1 and $min(v, a)$, where a refers to the number of attractions. $min(v, a)$ here is the mathematical dimension limit for SVD. In our case, r can be 17 ($min(22287, 17)$) at most. Since SVD provides a singular matrix with sorted values, the initial features are more critical to approximate than the subsequent ones. In particular, the top five singular values returned by SVD are 8256, 2794, 2234, 2148, 2091. Due to the large difference of the first singular value to the rest, single dimension already provides a very good approximation to the time matrix alone. The clusters derived by G-means also showed that for $r > 2$, there is no significant



(a)



(b)

Fig. 23.3 Degree of complementarity of the bundles in the given historical redemption data (only attraction sets with at least 1% of occurrence ($f_r(b_i) \geq 0.01$) are considered). (**a**) All bundles. (**b**) Per bundle size

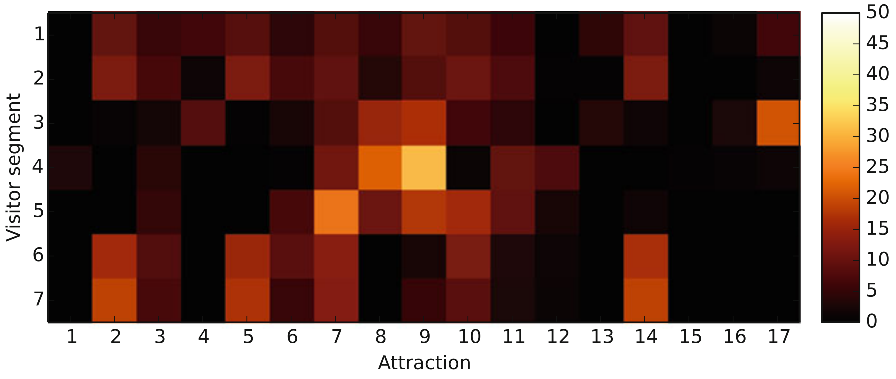
Table 23.7 Eight features representing visitors

Feature	Definition
f0	Ticket type
f1	Number of visited attractions
f2	Visit day
f3	First visit start time in hours
f4	Day type: weekday or weekend
f5	Total time spent except for the last visited attraction
f6	Average spent time except for the last visited attraction
f7	Standard deviation of the time spent, ignoring the last visited attraction

change on the clusters, thus r is set to 2. In the next step (Algorithm 1, line 3), G-means revealed 207 clusters derived from the U matrix with $r = 2$. The 207 clusters are further analysed to detect the similar clusters in terms of their normalized visit frequencies using two similarity metrics, namely Cosine and Pearson (Algorithm 1, line 4–5). After combining similar clusters, the number of clusters is decreased to 7 and 8 for Cosine and Pearson (Algorithm 1, line 6).

Figure 23.4 visualizes differences between clusters/visitor segments regarding percentage visit frequencies of each attraction in each segment. In other words, this figure gives information like 30% of the time attraction a_i is visited while attraction a_j is visited only 5% of the time. In the redemption data, the reported visit frequencies indicate that there is no clear similarity between visitor segments meaning that visitors are efficiently segmented based on this most basic visit frequencies. This is valid for both the Cosine and Pearson similarity metrics. When the visitor segments derived using these two metrics are compared, per attraction visit behaviours of the customers look similar. This shows that there is relatively high overlap between segments of both metrics. If we just consider the most frequently visited attractions in each segment, attractions 2, 7, 9, 14 and 17 are the most popular ones across different segments for both similarity metrics. If we look at the attractions which are either not visited at all or visited least frequently, attractions 1, 4, 8, 13, 15, 16, 17 and attractions 1, 2, 4, 5, 13, 15, 17 come out for Cosine and Pearson, respectively.

In order to further determine the elements affecting the formation of these clusters, eight basic visitor features are defined, as given in Table 23.7. The first feature represents the ticket type bought by visitors and the second feature shows the number of visits performed by each visitor. Unlike these two features, the remaining ones are all date/time-related. Random forests are applied to analyse the effects of these features on these clusters by generating a classification model between the features and clusters (as classes). Figure 23.5 details the Gini importance of each feature while building the corresponding classification models. The results indicate that the standard deviation of spent time on attractions and total spent time during a visit are the most critical features in determining the clusters obtained. Average spent time per attraction also contributed as another relatively useful feature.



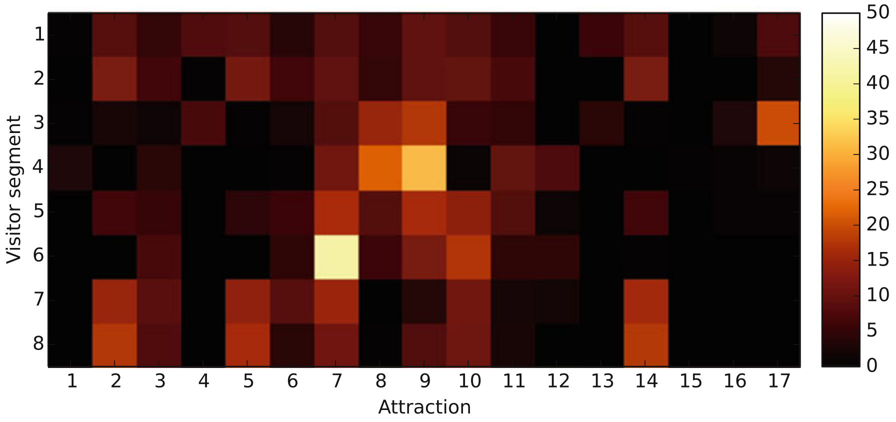
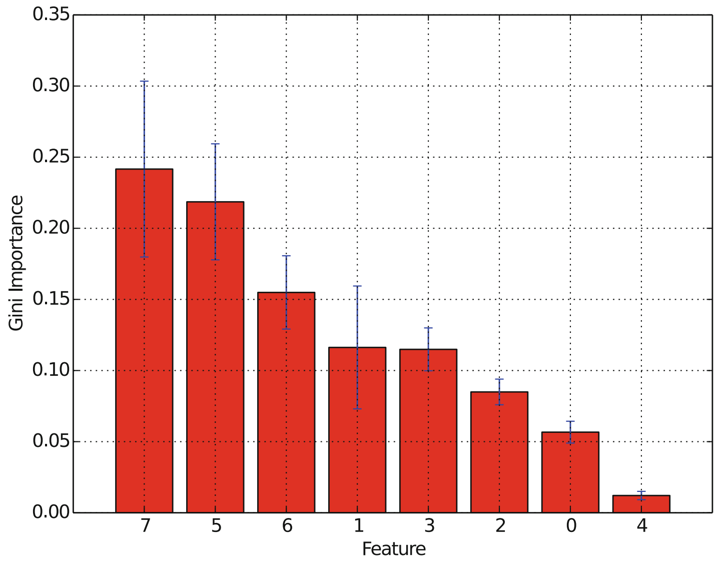


Fig. 23.4 Percentage visit frequencies of each attraction in each visitor segment for Cosine (top) and Pearson (bottom)

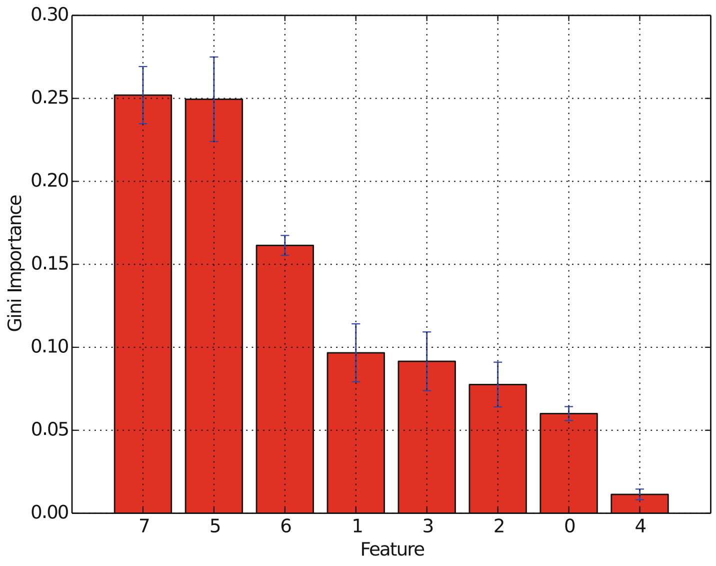
Following these major features, the number of visited attractions, the visit start time for the first attraction and weekly visit day respectively. Among the least important two features, the ticket type has very limited effect since the tickets provided do not really differentiate the behaviour of the visitors. The worst performing feature, in terms of separating the clusters, is that of whether a particular visit is performed on a weekday or weekend. Considering that the number of resulting clusters and this feature is binary, its effect on clusters is negligible as expected.

23.5.3 Static Recommendation

Figure 23.6 illustrates the bundle tickets recommended for different bundle sizes. When Cosine is used as the similarity metric to combine clusters determined by G-means, attractions 6 and 12 are the ones not included in any of the bundles.

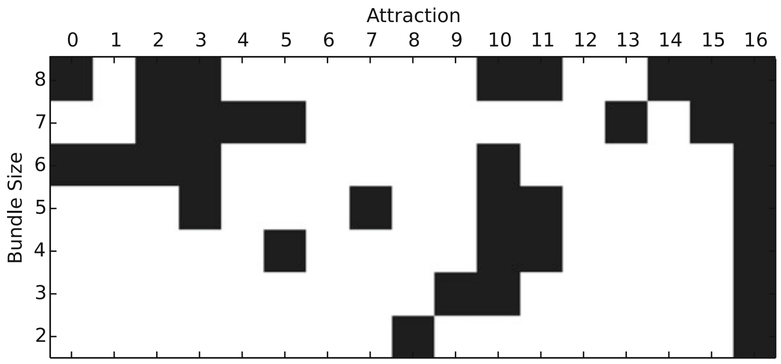


(a)

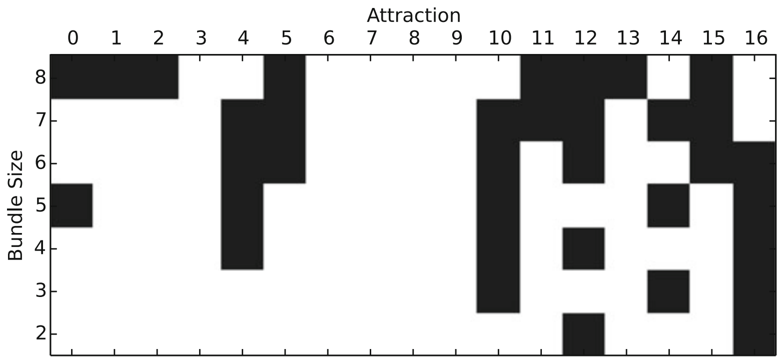


(b)

Fig. 23.5 Feature/variable importance determined by random forests using eight features based on clusters determined by the two-stage clustering (features are sorted w.r.t. their importance levels). (**a**) Cosine—seven segments. (**b**) Pearson—eight segments



(a)



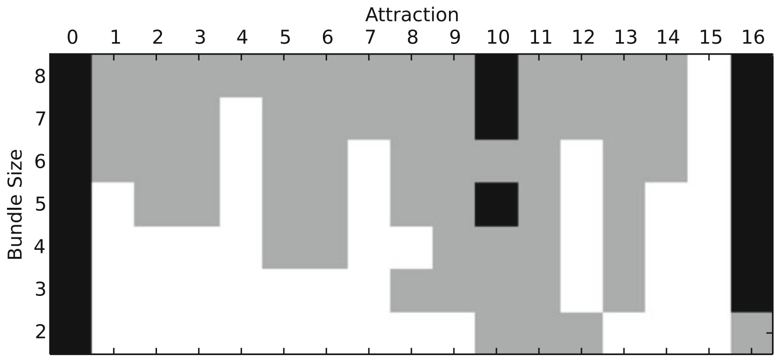
(b)

Fig. 23.6 Outlier bundles delivered by the Cosine and Pearson similarity metrics (black refers to fixed attractions, white shows that attractions are not recommended in bundles). (**a**) Cosine. (**b**) Pearson

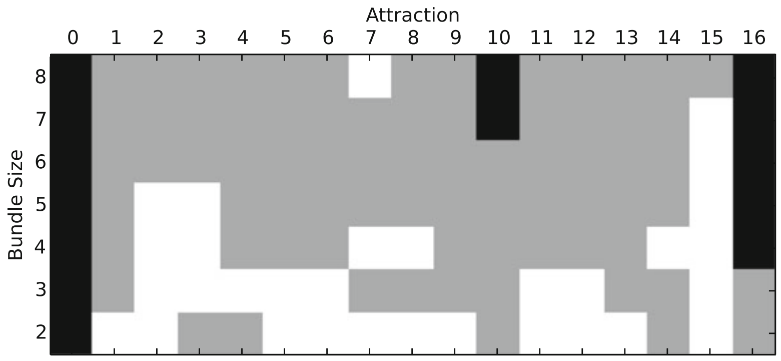
Attraction 16 is included in each bundle. For the Pearson Correlation, attractions 3, 6, 7, 8 and 9 are not considered in the suggested bundles. Although there is no attraction that is always available in the bundles in this case, attraction 10 and attraction 16 are the most commonly bundled attractions.

Considering that the segment sizes are relatively small, flexible bundle tickets are proposed for the remaining segments. Figure 23.7 details these bundles. For both similarity metrics, 0 is the common attraction for all the reported bundle sizes. Attraction 16 is also bundled in the majority of the bundles. Attraction 10 is the last fixed bundled attraction, but only for relatively large-sized bundles. When Cosine is used in segmentation, many attractions are excluded from the bundles, especially for the bundles with a few attractions. The Pearson similarity provides more flexible choices for different bundle sizes.

In order to perform an additional analysis on the quality of the visitor segments or to determine whether visitor segmentation is required for our data, we evaluated



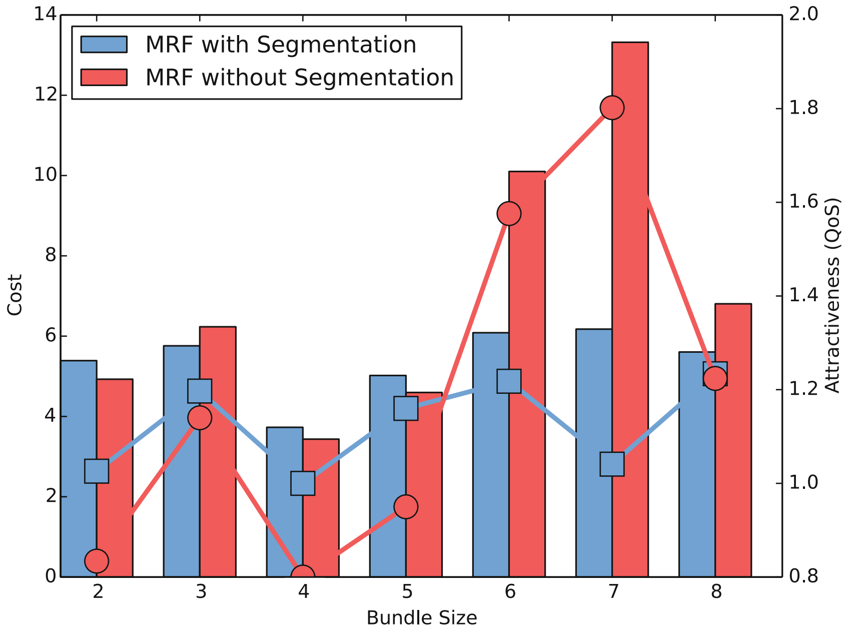
(a)



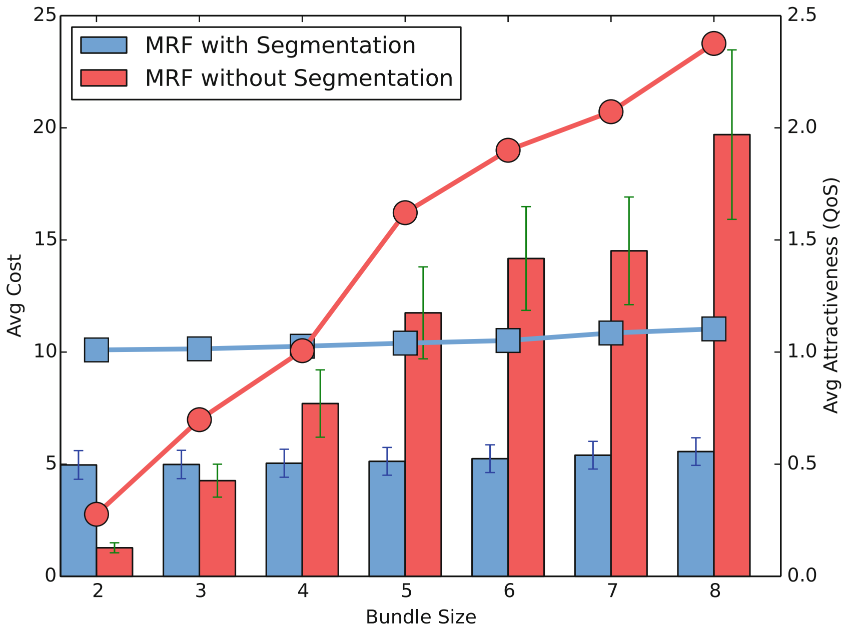
(b)

Fig. 23.7 Non-outlier flexible bundles delivered by the Cosine and Pearson similarity metrics (black refers to fixed attractions, grey shows that attractions that can be added to the bundles, white shows that attractions are not recommended in bundles). (**a**) Cosine. (**b**) Pearson

both the expected cost and attractiveness (QoS) of each static bundle generated for each segment. For evaluation, we generated a separate MRF for each visitor segment and one MRF using whole redemption data without segmentation. Using the resulting MRFs, we re-evaluated each static bundle. Figures 23.8 and 23.9 illustrate the results for the static bundles generated after visitor segmentation with Cosine and Pearson, respectively. In both figures, one chart focuses on only the largest segments which have significantly more visitors than the remaining segments. For these charts, expected bundle costs and attractiveness are directly reported. For the smaller-sized visitor segments, the costs and attractiveness values are illustrated in the form of weighted average. Equations (23.27) and (23.28) show how average cost and average attractiveness are calculated. When the largest single segment in Cosine is considered, similar cost evaluation is seen for both MRF of the whole data and segment specific MRFs for the bundles sized until $K = 5$. For the larger sized bundles ($K > 5$), in particular the ones with 6 and 7 attractions,

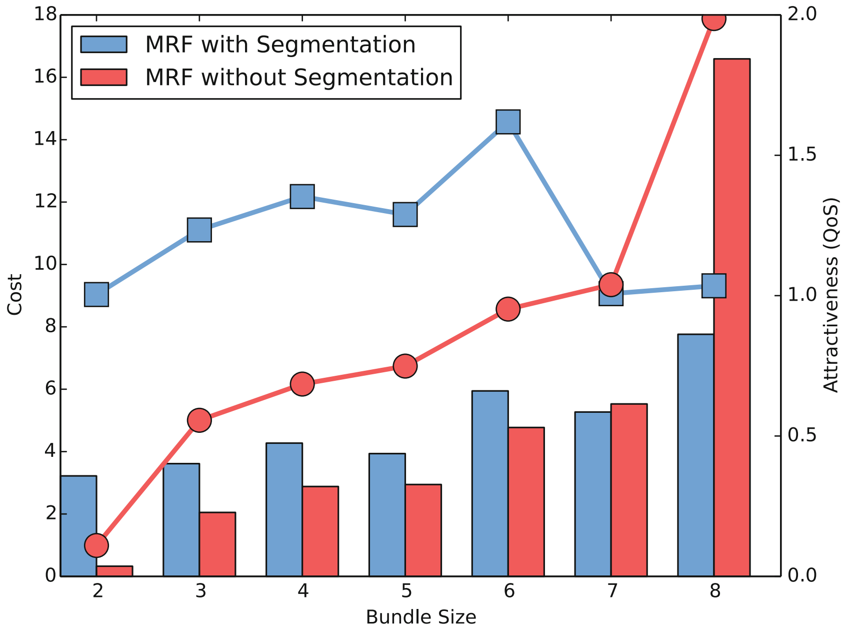


(a)

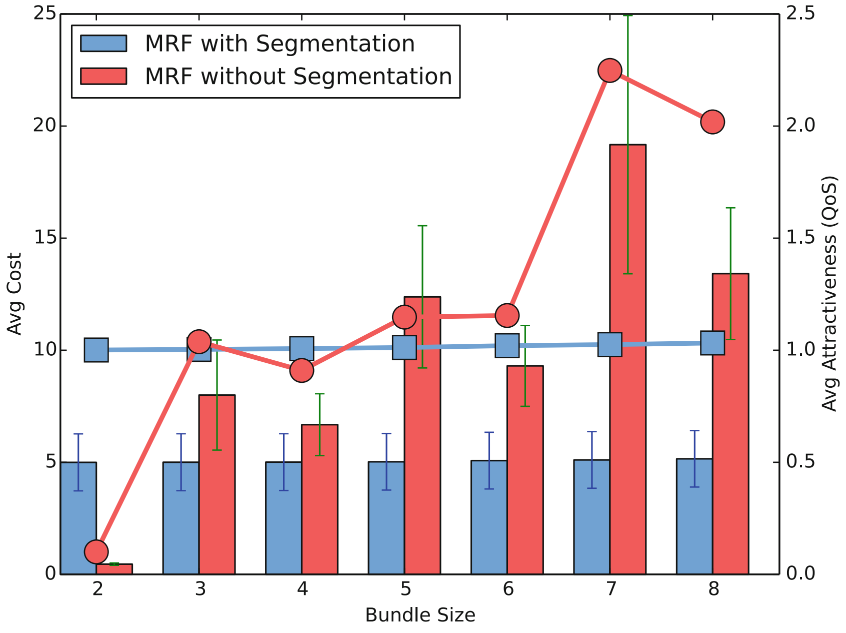


(b)

Fig. 23.8 The effect of Cosine-based visitor segmentation in terms of bundles' expected costs and attractiveness (QoS constraint set to 1.0; bar plots show cost, line plots indicate attractiveness). (**a**) Outlier/largest segment. (**b**) Remaining segments



(a)



(b)

Fig. 23.9 The effect of Pearson-based visitor segmentation in terms of bundles' expected costs and attractiveness (QoS constraint set to 1.0); bar plots show cost, line plots indicate attractiveness). (**a**) Outlier/largest segment. (**b**) Remaining segments

full redemption data based MRF expects significantly more costly bundles while segment specific MRFs tell that they are actually expected to incur less cost to the park operator. The bundles' attractiveness levels are smaller for the full data based MRF compared to the segment specific MRFs except the bundles with size 6 and 7 which are the costly ones. In the Pearson-based largest segment, the changes on cost and attractiveness are smoother. The expected bundle costs consistently increase in relation to the bundles' sizes for the MRF generated without segmentation. Until a bundle size of 7, segment specific MRF based expected costs and attractiveness levels are higher. For a bundle size of 8, the full data based MRF evaluates the corresponding bundle as significantly costly and very attractive for the visitors. A similar trend can be seen for the remaining segments of both Cosine and Pearson. In these cases, cost and attractiveness of segment based MRFs are very consistent.

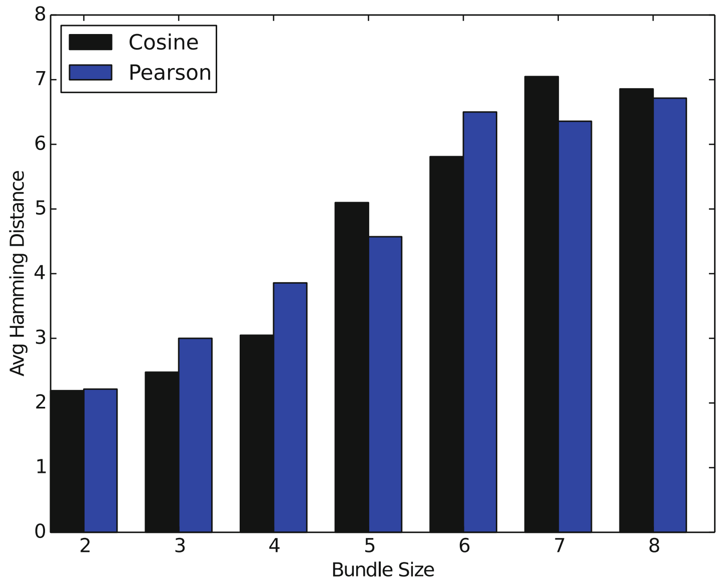
$$\sum_{i}^{k} V_{s,i} \times \mathcal{C}(b_i)/|V_s| \tag{23.27}$$

$$\sum_{i}^{k} V_{s,i} \times \mathcal{Q}(b_i)/|V_s| \tag{23.28}$$

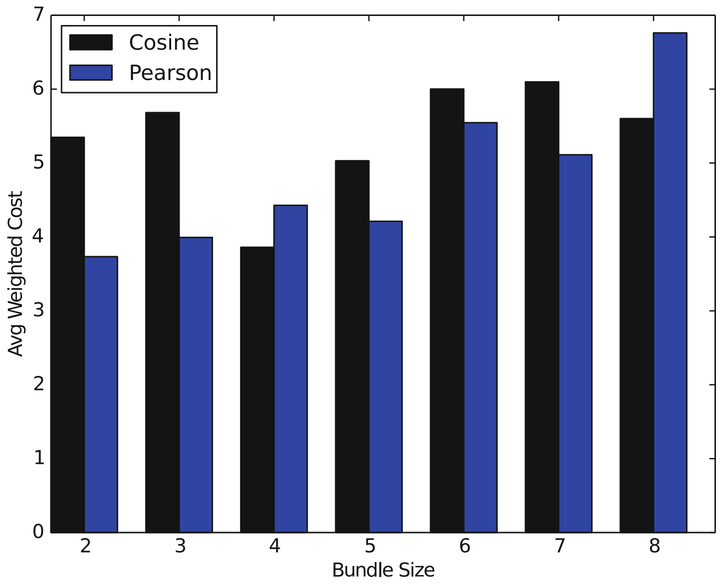
In brief, these figures indicate that generating a single MRF using whole redemption data can be misleading due to not taking visitors with distinct attraction preferences into account. The differences on the expected costs and attractiveness can be explained by two reasons. If expected cost and attractiveness are higher when full data based single MRF is used, it means that the corresponding bundles occur more frequently in the complete data while these bundles are rarely preferred in some of the visitor segments. For the opposite case, there can be a small group of visitors with similar visit behaviour. Since the number of these visitors is small, they are considered as not-interesting and the single full data based MRF is unable to appreciate their specific attractiveness to these small set of visitors. Thus, the expected costs and attractiveness are measured as lower by the single MRF.

Figure 23.10 shows how diverse the static bundles are and the average weighted cost incurs when either Cosine or Pearson is used as a similarity metric besides the bundle size. Both diversity and cost are measured across all the static bundles generated for all the visitors segments. Diversity is particularly useful to check how similar the bundles proposed for each visitor segment. High diversity indicates that different visitors have distinct preferences while low diversity means that visitor segmentation is unnecessary for bundling (indicating a homogeneous customer base). Diversity is measured using the mean hamming distance, $\mathcal{H}(B)$, where B refers all the bundles. As shown in Eq. (23.29), it is calculated as the average of the pairwise distances.

$$\mathcal{H}(B) = \frac{2}{N(N-1)} \sum_{i_1 \neq i_2}^{N} \sum_{j=1}^{l} |r_{i_1,j} - r_{i_2,j}| \tag{23.29}$$



(a)



(b)

Fig. 23.10 Diversity and cost of the bundles with varying sizes generated by the greedy heuristic for the visitor segments. (**a**) Diversity. (**b**) Avg weighted cost

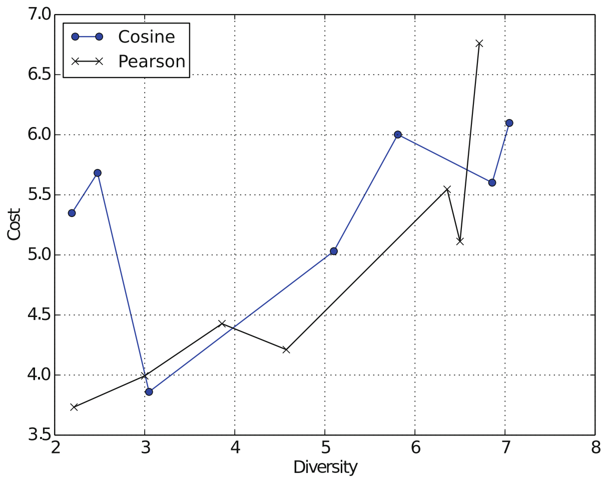


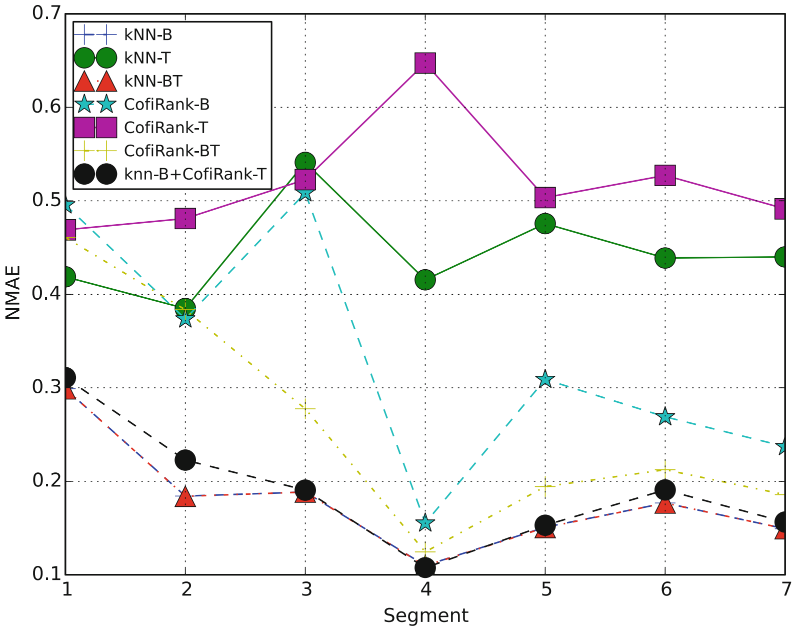
Fig. 23.11 Diversity vs Cost relation for the Cosine and Pearson similarity metrics

In terms of diversity, the best similarity metric changes for different bundle sizes. For instance, if the bundle size is set to 4, Pearson provides higher diversity but Cosine is better when bundle size is set to 7. Besides that, the diversity values in general and their increase in relation to bundle size indicate that visitor segmentation *is* in fact required and the proposed approach is able to detect the differentiate visitors segments. The cost results indicate that Cosine delivers more costly bundles for 5 out of 7 bundle size options. As detailed in Fig. 23.11, if the relation between Diversity and Cost are explicitly analysed, Pearson usually offers bundles with lower costs for similar diversity levels than Cosine.

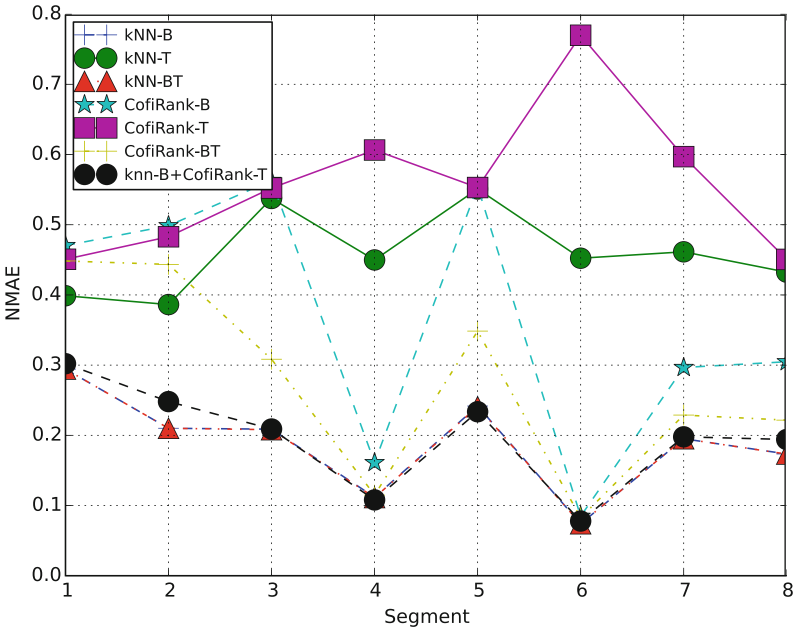
23.5.4 Dynamic Recommendation

The performance of different CF recommendation approaches is analysed on the binary and time matrices. The first prediction task is to determine which attractions should be included in a bundle in the form of dynamic bundling. The other prediction task is about accurately detecting visiting order of the bundled attractions.

Figure 23.12 reports Normalized Mean Absolute Error (NMAE) (Eq. 23.30) for predicting the right dynamic bundle suggestions in terms of the attractions included in the bundles. MAE indicates the prediction error as the average of the absolute difference between the predicted and actual values. NMAE is the normalized version of MAE. For example, NMAE = 0.2 means that 20% of the predictions are wrong, so lower NMAEs are better. The results indicate that kNN is able to deliver more accurate bundle predictions than CofiRank when the Binary matrix is used even though kNN is faster and simpler.



(a)



(b)

Fig. 23.12 Normalized Mean Absolute Error (NMAE) for evaluating the quality of the dynamic bundle suggestions (the graphs belong Cosine and Pearson respectively). (**a**) Cosine. (**b**) Pearson

$$MAE = \frac{\sum_{i,j} |p_{ij} - r_{ij}|}{n}$$

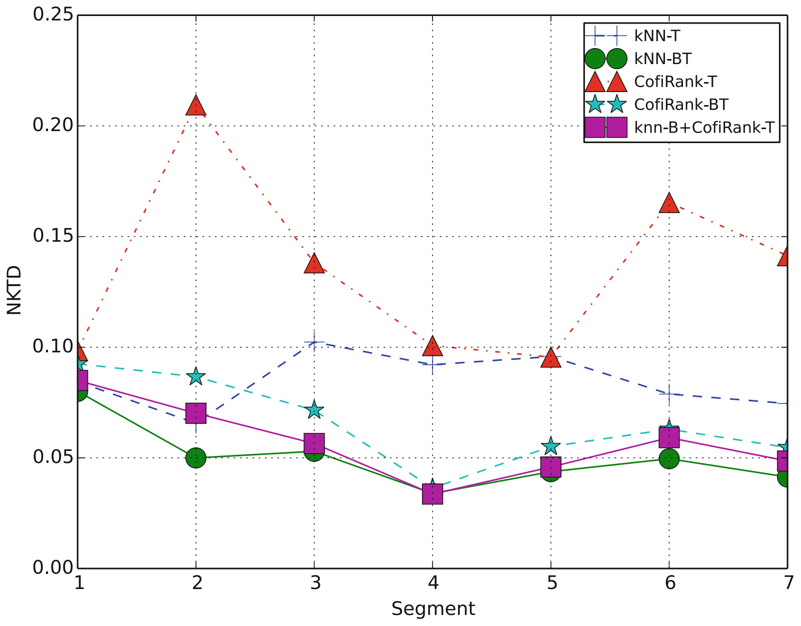
$$NMAE = \frac{MAE}{r_{max} - r_{min}} \tag{23.30}$$

The kNN performance changes between ∼10% and ∼30% for different visitor segments both for Cosine and Pearson Correlation, which is very successful from a collaborative filtering perspective. The underlying reason behind the superior performance of kNN is the given incomplete matrix. Considering that we have a large amount of visitor redemption data, the matrix incompleteness level/sparsity is very low in comparison to the existing target collaborative filtering data. Thus, memory-based approaches like kNN are expected to perform well. When the incompleteness level is very low or there is not enough redemption data, model-based approaches like CofiRank are likely to perform better. If the time matrix is used to predict which attractions should be included in a bundle for each visitor, the results are significantly worse compared to the binary matrix case. As mentioned earlier, the reason is that the time matrix has two types of information including attractions available in each bundle and the visiting time of each attraction with higher range of values than just 2 as in a binary matrix. Applying a CF algorithm directly to a time matrix assigns time to many of the unvisited attractions. Thus, these attractions are also assumed to be visited in the predictions which result in high NMAEs.

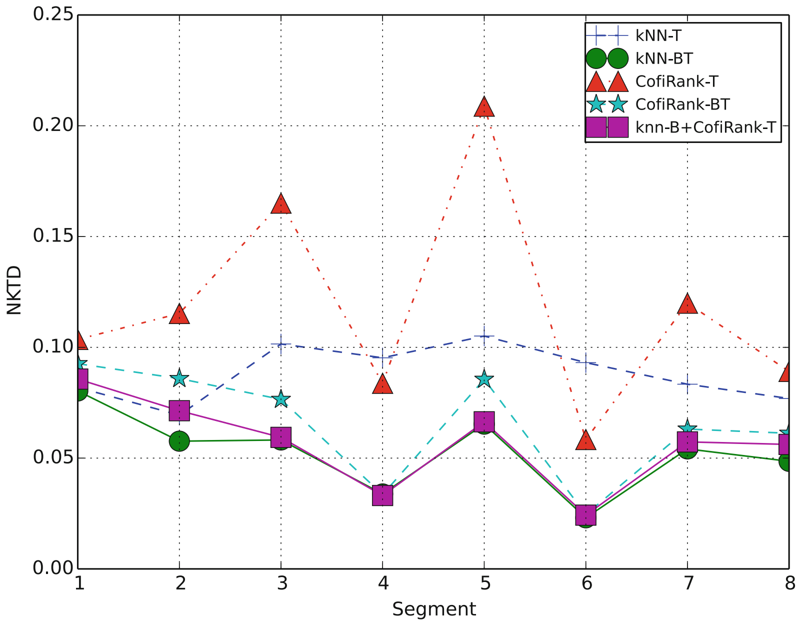
In order to resolve this issue of using the time matrix for prediction, the predictions on the binary matrix are utilized to correct the results on the time matrix. As a consequence, a combined approach, i.e. kNN-BT, which applies kNN to the binary matrix first to correct the predictions of kNN on the time matrix, delivered the best performance. The Wilcoxon rank sum test within a 99% confidence level indicated that kNN-BT is statistically better than all tested approaches except kNN-T and kNN-B+CofiRank-T.

Besides recommending dynamic bundles efficiently, predicting a good visiting order for bundles would help the visitors to enjoy the theme park more or to make better choices to decide on which attraction to visit next. Figure 23.13 evaluates the performance of the tested CF methods for the ordinal predictions in terms of the normalized Kendall's tau distance (NKTD) $\in [0, 1]$ (Eq. 23.32).

KTD basically counts the number of times when the pairwise visiting order of attractions is incorrect. For the ordinal predictions, kNN-BT comes as the statistically best method together with kNN-B+CofiRank-T according to the Wilcoxon rank sum test within a 99% confidence level. kNN-BT delivers results with NKTD varying between 0.04 and 0.07. This indicates that CF is successful in determining good visit orders or trip plans while suggesting dynamic bundle recommendations. CF is able to address a small-sized optimization problem without needing an optimization algorithm, solely as a recommender system.



(a)



(b)

Fig. 23.13 Normalized Kendall's tau distance (NKTD) for assessing the recommended visiting order of the dynamic bundle suggestions (the graphs belong Cosine and Pearson respectively). (**a**) Cosine. (**b**) Pearson

$$KTD = \sum_{i}^{v} \sum_{j}^{a} \sum_{k}^{a} z_{i,j,k} \text{ where} \tag{23.31}$$

$$z_{i,j,k} = \begin{cases} 1, & \mathcal{M}(i,j) < \mathcal{M}(i,k) \text{ and } \mathcal{P}(i,j) > \mathcal{P}(i,k) \\ 0, & \text{otw.} \end{cases}$$

$$NKTD = \frac{2KTD}{v(a^2 - a)} \tag{23.32}$$

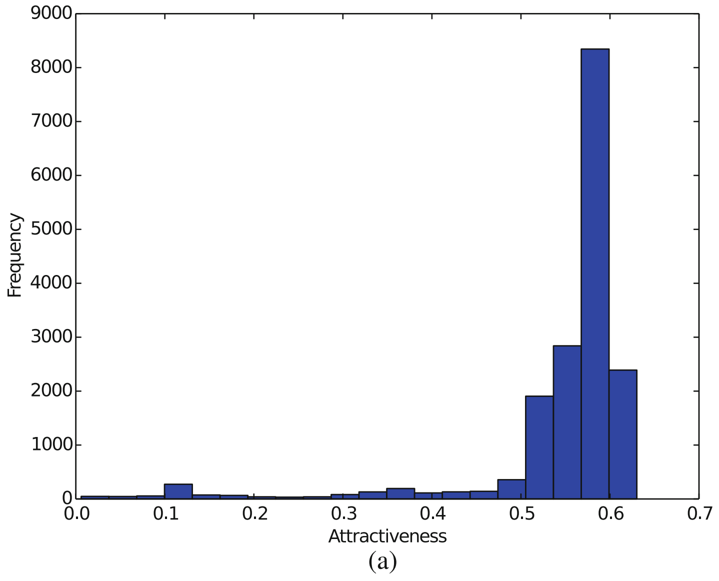
Figure 23.14 shows the average attractiveness of each attraction included in dynamic bundles for the first and largest visitor segments of both Cosine and Pearson. Attractiveness is measured again using MRFs.

The majority of the bundles' attractiveness levels are around 0.5–0.6. Although MRFs are not suitable for detecting attractive bundles for relatively small group of visitors, it is still able to indicate that the tested CF-based dynamic bundling methods are able to deliver attractive bundles. This is important since the idea of CF is all about providing attractive suggestions. Thus, this analysis also supports the earlier results about the success of using CF for dynamic bundling.

The best performing CF-based dynamic bundling method, i.e. kNN-BT, is compared to Apriori based ARM approach. Figure 23.15 presents the NMAEs both for Cosine and Pearson. kNN-BT clearly outperforms ARM based dynamic bundling and its performance is statistically better within a 99% confidence level for the Wilcoxon rank sum test.

23.5.4.1 Dynamic Recommendation as Static Recommendation

The reported static bundles are found by applying the Greedy heuristic. In order to make a direct comparison between static bundling and dynamic bundling, the same heuristic is used to offer dynamic bundles. Thus, dynamic bundling is thought as static bundling with prior preferences. However, the Greedy heuristic failed to generate comparable bundles. The one reason is that it adds attractions to the additional attractiveness of c/K. If there is no attraction that meets this criterion, it is unable to add new attractions to the bundle. Considering that the dynamic bundles offered by CF are highly attractive, this criterion fails to generate comparable bundles at most cases. Besides that, CF is able to offer large sized bundles mostly at size of 13 attractions. Since the total number of attractions 17, there is not much of attraction choices to vary. The aforementioned per attraction attractiveness constraint, c/K, becomes very challenging to meet. Thus, in order to suggest good dynamic bundles via static bundling, the corresponding static bundling algorithm should be more flexible, e.g. having a backtracking method.



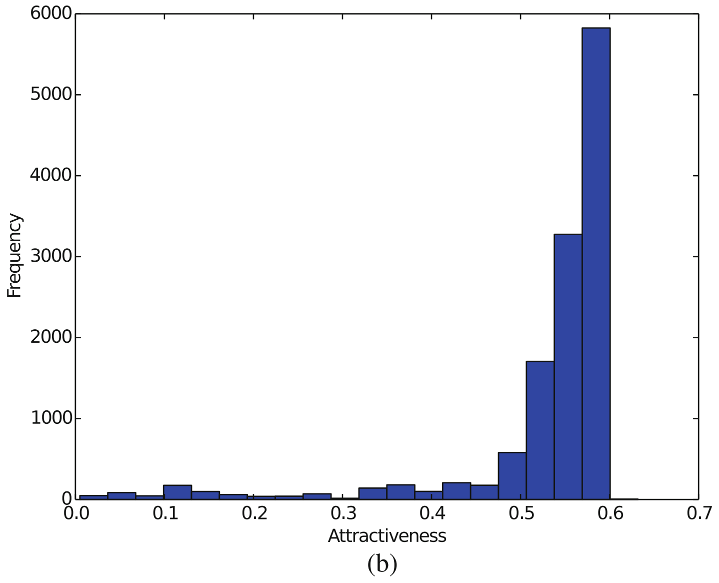
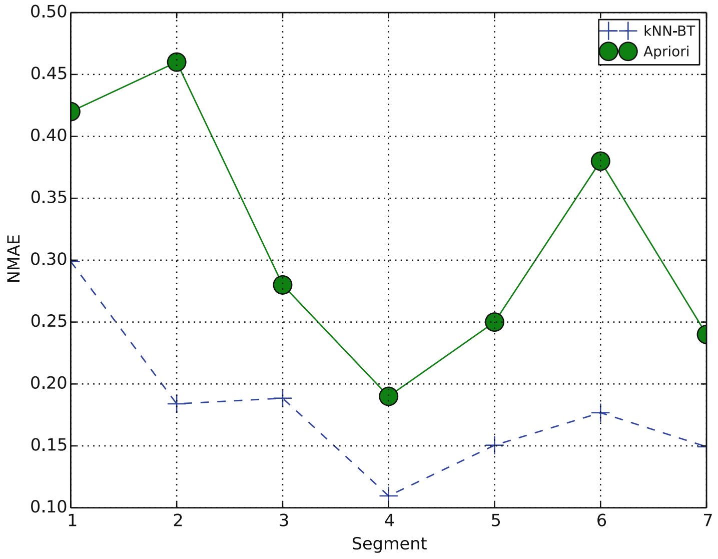
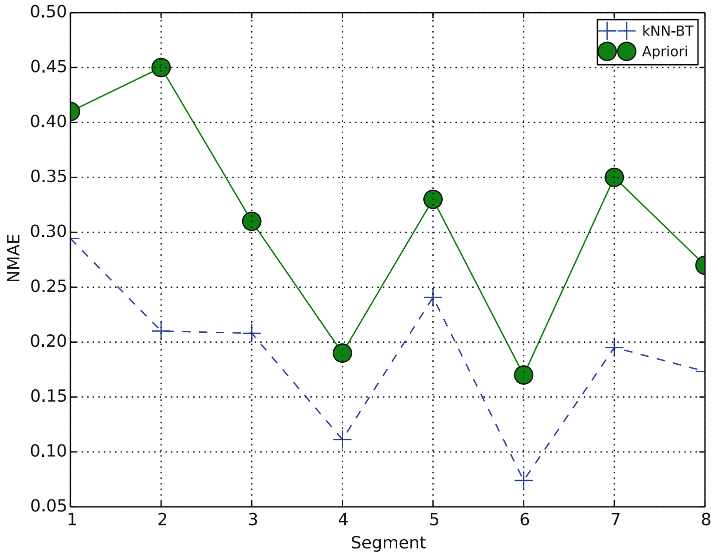


Fig. 23.14 Average attractiveness of the dynamic bundles offered w.r.t. MRF's attractiveness measure. (**a**) Cosine—Segment #1. (**b**) Pearson—Segment #1



(a)



(b)

Fig. 23.15 Normalized Mean Absolute Error (NMAE) for evaluating the quality of the dynamic bundle suggestions of the best tested CF method and ARM based recommendation (the graphs are from Cosine and Pearson results respectively). (**a**) Cosine. (**b**) Pearson

23.6 Conclusion

This chapter introduces a data-driven approach to recommend personalized bundles for leisure parks using historical visitor trajectory data. Our idea is to first perform visitor segmentation using clustering. We then deliver personalized recommendations on the fly to make changes on existing ticket bundle. This is akin to the general idea in marketing of targeted upselling and cross-selling on the fly based on the specific visitor consumption pattern, backed by insights derived from a wealth of data collected from past visitor sale and consumption records. What set our work apart from many others is that we consider bundling and provide recommendation to a visitor *without* the need to take specific visitor information. While this idea has been studied under collaborative filtering that uses any preference or rating data to derive recommendations, it is mostly in the online world. We have adapted it for the physical world in a specific leisure park setting, which we believe to be first of its kind.

Our experimental results showed the correlation between bundle sizes, their diversity levels and costs for the static bundles. The low error rates achieved by the tested CF methods revealed the appropriateness of considering the bundling problem as a recommendation problem.

For future research, the performance of hybrid recommendation systems combining content-based and collaborative filtering should be investigated. The temporal dynamics based matrix factorization approaches will be additionally applied to offer dynamic bundles considering long-term visitor behavioural changes.

Acknowledgements The authors would like to thank Shih-Fen Cheng, Pradeep Varankantham, and a number of un-named team members who contributed in various ways to this project. This research is supported by the Singapore National Research Foundation under its International Research Centre @ Singapore Funding Initiative and administered by the IDM Programme Office, Media Development Authority (MDA).

References

1. Manoj K Agarwal and Subimal Chatterjee. Complexity, uniqueness, and similarity in between-bundle choice. *Journal of Product & Brand Management*, 12(6):358–376, 2003.
2. Rakesh Agrawal, Tomasz Imieliński, and Arun Swami. Mining association rules between sets of items in large databases. In *ACM SIGMOD Record*, volume 22, pages 207–216. ACM, 1993.
3. Rakesh Agrawal, Ramakrishnan Srikant, et al. Fast algorithms for mining association rules. In *Proc. 20th int. conf. very large data bases, VLDB*, volume 1215, pages 487–499, 1994.
4. T. W. Anderson and D.A. Darling. Asymptotic theory of certain "goodness of fit" criteria based on stochastic processes. *The Annals of Mathematical Statistics*, 23(2):193–212, 1952.
5. Mark Armstrong. A more general theory of commodity bundling. *Journal of Economic Theory*, 148(2):448–472, 2013.
6. Alessandro Avenali, Anna DÁnnunzio, and Pierfrancesco Reverberi. Bundling, competition and quality investment: a welfare analysis. *Review of Industrial Organization*, 43(3):221–241, 2013.

7. Yannis Bakos and Erik Brynjolfsson. Bundling information goods: Pricing, profits, and efficiency. *Management Science*, 45(12):1613–1630, 1999.
8. Yannis Bakos and Erik Brynjolfsson. Bundling and competition on the internet. *Marketing Science*, 19(1):63–82, 2000.
9. Moran Beladev, Bracha Shapira, and Lior Rokach. Recommender systems for product bundling. *Knowledge-Based Systems*, 111:193–206, 2016.
10. James Bennett and Stan Lanning. The netflix prize. In *Proceedings of KDD cup and workshop*, volume 2007, page 35, 2007.
11. D. Billsus and M.J. Pazzani. Learning collaborative information filters. In *Proceedings of the 15th International Conference on Machine Learning (ICML'98)*, pages 46–54, 1998.
12. Enrico Calandro and Chenai Chair. Policy and regulatory challenges posed by emerging pricing strategies. *Information Technologies & International Development*, 12(2), 2016.
13. Kathryn A Carroll, Anya Savikhin Samek, Lydia Zepeda, et al. Product bundling as a behavioral nudge: Investigating consumer fruit and vegetable selection using dual-self theory. In *2016 Annual Meeting, July 31-August 2, 2016, Boston, Massachusetts*, number 236130. Agricultural and Applied Economics Association, 2016.
14. Suchan Chae. Bundling subscription tv channels: A case of natural bundling. *International Journal of Industrial Organization*, 10(2):213–230, 1992.
15. Yongmin Chen. Equilibrium product bundling. *Journal of Business*, 70(1):85–103, 1997.
16. Yongmin Chen and Michael H Riordan. Profitability of product bundling. *International Economic Review*, 54(1):35–57, 2013.
17. Bing Tian Dai and Hady W Lauw. Modeling preferences with availability constraints. In *2013 IEEE 13th International Conference on Data Mining*, pages 101–110. IEEE, 2013.
18. Gary D Eppen, Ward A Hanson, and R Kipp Martin. *Bundling-new products, new markets, low risk*. Purdue University, Krannert Graduate School of Management, 1991.
19. David S Evans and Karen L Webster. Designing the right product offerings. *MIT Sloan Management Review*, 49(1):44, 2007.
20. Esther Gal-Or. Evaluating the profitability of product bundling in the context of negotiations. *The Journal of Business*, 77(4):639–674, 2004.
21. Joshua S Gans and Stephen P King. Paying for loyalty: Product bundling in oligopoly. *The Journal of Industrial Economics*, 54(1):43–62, 2006.
22. Xianjun Geng, Maxwell B Stinchcombe, and Andrew B Whinston. Bundling information goods of decreasing value. *Management science*, 51(4):662–667, 2005.
23. G. Hamerly and C. Elkan. Learning the k in k-means. In *Proceedings of the 17th Annual Conference on Neural Information Processing Systems (NIPS'03)*, pages 281–288, 2003.
24. Judy Harris and Edward A Blair. Consumer preference for product bundles: The role of reduced search costs. *Journal of the Academy of Marketing Science*, 34(4):506–513, 2006.
25. Michael D Johnson, Andreas Herrmann, and Hans H Bauer. The effects of price bundling on consumer evaluations of product offerings. *International Journal of Research in Marketing*, 16(2):129–142, 1999.
26. Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, 2009.
27. V. Krishnan and Karl T. Ulrich. Product development decisions: A review of the literature. *Management Science*, 47(1):1–21, 2001.
28. Arthur Lewbel. Bundling of substitutes or complements. *International Journal of Industrial Organization*, 3(1):101–107, 1985.
29. Greg Linden, Brent Smith, and Jeremy York. Amazon. com recommendations: Item-to-item collaborative filtering. *Internet Computing, IEEE*, 7(1):76–80, 2003.
30. Qi Liu, Enhong Chen, Hui Xiong, Yong Ge, Zhongmou Li, and Xiang Wu. A cocktail approach for travel package recommendation. *IEEE Transactions on Knowledge and Data Engineering*, 26(2):278–293, 2014.
31. Silvano Martello, David Pisinger, and Paolo Toth. New trends in exact algorithms for the 0–1 knapsack problem. *European Journal of Operational Research*, 123(2):325–332, 2000.

32. R Preston McAfee, John McMillan, and Michael D Whinston. Multiproduct monopoly, commodity bundling, and correlation of values. *The Quarterly Journal of Economics*, 104:pages 371–383, 1989.
33. T.H. Nguyen, P.R. Varakantham, S-F. Cheng, and H.C. Lau. Mining resource bundles by balancing profitability. LARC-TR-01-14, Singapore Management University, 2014.
34. Dusit Niyato, Dinh Thai Hoang, Nguyen Cong Luong, Ping Wang, Dong In Kim, and Zhu Han. Smart data pricing models for the internet of things: a bundling strategy approach. *IEEE Network*, 30(2):18–25, 2016.
35. Brooks Pierce and Harold Winter. Pure vs. mixed commodity bundling. *Review of Industrial Organization*, 11(6):811–821, 1996.
36. Srinivasan Raghunathan and Sumit Sarkar. Competitive bundling in information markets: A seller-side analysis. *MIS Quarterly*, 40(1):111–131, 2016.
37. Elli Rapti, Anthony Karageorgos, and Georgios Ntalos. Adaptive constraint and rule-based product bundling in enterprise networks. In *IEEE 23rd International WETICE Conference*, pages 15–20. IEEE, 2014.
38. Jasson D.M. Rennie and Nathan Srebro. Fast maximum margin matrix factorization for collaborative prediction. In *Proceedings of the 22nd International Conference on Machine learning*, pages 713–719. ACM, 2005.
39. P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl. Grouplens: an open architecture for collaborative filtering of netnews. In *Proceedings of the 1994 ACM conference on Computer supported cooperative work*, pages 175–186. ACM, 1994.
40. Francesco Ricci, Lior Rokach, and Bracha Shapira. *Introduction to recommender systems handbook*. Springer, 2011.
41. Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th International Conference on World Wide Web*, pages 285–295. ACM, 2001.
42. Andrew I Schein, Alexandrin Popescul, Lyle H Ungar, and David M Pennock. Methods and metrics for cold-start recommendations. In *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 253–260. ACM, 2002.
43. Hanhuai Shan and Arindam Banerjee. Generalized probabilistic matrix factorizations for collaborative filtering. In *Proceedings of the IEEE 10th International Conference on Data Mining (ICDM'10)*, pages 1025–1030. IEEE, 2010.
44. Mehdi Sheikhzadeh and Ehsan Elahi. Product bundling: Impacts of product heterogeneity and risk considerations. *International Journal of Production Economics*, 144(1):209–222, 2013.
45. Sandro Shelegia. Multiproduct pricing in oligopoly. *International Journal of Industrial Organization*, 30(2):231– 242, 2012.
46. Shibin Sheng, Andrew M Parker, and Kent Nakamoto. The effects of price discount and product complementarity on consumer evaluations of bundle components. *Journal of Marketing Theory and Practice*, 15(1):53–64, 2007.
47. Allan D Shocker, Barry L Bayus, and Namwoon Kim. Product complements and substitutes in the real world: The relevance of "other products". *Journal of Marketing*, 68(1):28–40, 2004.
48. Nathan Srebro, Jason D.M. Rennie, and Tommi Jaakkola. Maximum-margin matrix factorization. In *NIPS*, volume 17, pages 1329–1336, 2004.
49. Stefan Stremersch and Gerard J Tellis. Strategic bundling of products and prices: A new synthesis for marketing. *Journal of Marketing*, 66(1):55–72, 2002.
50. Xiaoyuan Su and Taghi M Khoshgoftaar. A survey of collaborative filtering techniques. *Advances in Artificial Intelligence*, 2009:4, 2009.
51. Chang Tan, Qi Liu, Enhong Chen, Hui Xiong, and Xiang Wu. Object-oriented travel package recommendation. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 5(3):43, 2014.
52. M Mithat Üner, Faruk Güven, and S Tamer Cavusgil. Bundling of telecom offerings: An empirical investigation in the turkish market. *Telecommunications Policy*, 39(1):53–64, 2015.

53. Robin Van Meteren and Maarten Van Someren. Using content-based filtering for recommendation. In *Proceedings of the Machine Learning in the New Information Age: MLnet/ECML2000 Workshop*, 2000.
54. R Venkatesh and Wagner Kamakura. Optimal bundling and pricing under a monopoly: Contrasting complements and substitutes from independently valued products*. *The Journal of Business*, 76(2):211–231, 2003.
55. R Venkatesh and Vijay Mahajan. The design and pricing of bundles: a review of normative guidelines and practical approaches. *Handbook of Pricing Research in Marketing*, page 232, 2009.
56. Yuanhong Wang, Yang Liu, and Xiaohui Yu. Collaborative filtering with aspect-based opinion mining: A tensor factorization approach. In *Proceedings of the IEEE 12th International Conference on Data Mining (ICDM'12)*, pages 1152–1157. IEEE, 2012.
57. Markus Weimer, Alexandros Karatzoglou, Quoc Viet Le, and Alex Smola. CofiRank: Maximum margin matrix factorization for collaborative ranking. In *Proceedings of the 21th Annual Conference on Neural Information Processing Systems (NIPS'07)*, 2007.
58. Wann-Yih Wu, Badri Munir Sukoco, Chia-Ying Li, and Shu Hui Chen. An integrated multiobjective decision-making process for supplier selection with bundling problem. *Expert Systems with Applications*, 36(2):2327–2337, 2009.
59. Min Xie, Laks VS Lakshmanan, and Peter T Wood. Breaking out of the box of recommendations: from items to packages. In *Proceedings of the fourth ACM conference on Recommender systems*, pages 151–158. ACM, 2010.
60. Ruiliang Yan and Subir Bandyopadhyay. The profit benefits of bundle pricing of complementary products. *Journal of Retailing and Consumer Services*, 18(4):355–361, 2011.
61. Zhiwen Yu, Yun Feng, Huang Xu, and Xingshe Zhou. Recommending travel packages based on mobile crowdsourced data. *IEEE Communications Magazine*, 52(8):56–62, 2014.
62. Zhiwen Yu, Huang Xu, Zhe Yang, and Bin Guo. Personalized travel package with multipoint-of-interest recommendation based on crowdsourced user footprints. *IEEE Transactions on Human-Machine Systems*, 46(1):151–158, 2016.
63. Chengqi Zhang and Shichao Zhang. *Association rule mining: models and algorithms*. Springer-Verlag, 2002.
64. Mingyue Zhang and Jesse Bockstedt. Complements and substitutes in product recommendations: The differential effects on consumers' willingness-to-pay. In *Proceedings of the Joint Workshop on Interfaces and Human Decision Making for Recommender Systems co-located with ACM Conference on Recommender Systems (RecSys 2016), Boston, MA, USA, September 16, 2016.*, pages 36–43, 2016.
65. Sheng Zhang, Weihong Wang, James Ford, and Fillia Makedon. Learning from incomplete ratings using non-negative matrix factorization. In *SDM*, pages 549–553. SIAM, 2006.
66. Tao Zhu, Patrick Harrington, Junjun Li, and Lei Tang. Bundle recommendation in ecommerce. In *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval*, pages 657–666. ACM, 2014.

Chapter 24
A Fuzzy Evaluation of Tourism Sustainability

Joseph Andria, Giacomo di Tollo, and Raffaele Pesenti

Abstract For many years the sustainability assessment of tourist destinations has been based on the carrying capacity, which is a measure that takes into account the preservation of a geographical area (by measuring the number of tourists, the visitor flow and the environmental thresholds) along with its tourist fruition (by assessing the quality of the experience perceived by visitors). Unfortunately, its definition lacks clarity, and its dependence upon qualitative variables makes it unable to provide a unique criterion for its assessment.

In this paper we propose a fuzzy approach that takes into account the inherent uncertainty and vagueness of the involved variables to assess a destination's sustainability, by introducing a fuzzy indicator. Our approach has been applied to the Italian region of Sicily, showing a high versatility that makes its use possible jointly with (or alternatively to) other existing methodologies developed by European and international institutions.

Keywords Sustainable tourism · Social impact · Sustainable tourism management · Tourism · Tourist establishments density

J. Andria
Dipartimento di Scienze Economiche, Aziendali e Statistiche, University of Palermo, Palermo, Italy
e-mail: joseph.andria@unipa.it

G. di Tollo (✉)
Dipartimento di Economia, Universitá Ca' Foscari, Venezia, Italy
e-mail: giacomo.ditollo@unive.it

R. Pesenti
Dipartimento di Management, Universitá Ca' Foscari, Venezia, Italy
e-mail: pesenti@unive.it

P. Moscato, N. J. de Vries (eds.), *Business and Consumer Analytics: New Ideas*,
https://doi.org/10.1007/978-3-030-06222-4_24

24.1 Introduction

Environmental-intensive consumption and exploitation due to modern industrial practices are triggering more and more awareness about sustainability issues. There are many definitions of sustainability but the simplest and the most effective one is *the capacity to endure* [8]. From an economic perspective this leads to seek a balance between short and long-term approaches by trying to ensure short-term revenues along with a long-term growth strategy.

Nowadays, tourism is one of the most prominent industries which is able to generate an important contribution to a country's economy: it has been reported that in 2015 the global travel and tourism industry created approximately 11% of the global employment, and that travel and tourism represented approximately 10% of the global GDP (Gross Domestic Product). Furthermore, the average international tourist receipt is over US$700 per person, and travellers spent over $1.4 trillion [28].

The increased recognition of tourism as a strategic pillar for economic growth and development has boosted competitiveness among tourist destinations. This imposes the challenge of adopting sustainable tourism practices so that an optimal strategy has to define the right trade-off between usage and maintenance purposes: a further massive growth of the tourism sector, as predicted for the next years, would be a great prosperity opportunity but, at the same time, a potential threat to the environment and local communities if not well managed.

An analysis of the state of the art reveals that many definitions of tourism sustainability have been produced, and, consequently, there are no widely accepted measures in order to assess sustainability.

The World Tourism Organization (WTO) states that sustainable tourism can be defined as *"tourism that takes full account of its current and future economic, social and environmental impacts, addressing the needs of visitors, the industry, the environment and host communities"* [29] . Often the overused and little-understood term "sustainability" is replaced by "responsibility". Responsible tourism involves threes interconnected aspects: *environmental, economic, and socio-cultural.* More in detail, *the environmental* aspects focus on minimizing climate change, waste and conserving ecosystems and biodiversity, and encouraging a wise use of resources such as energy and water. The *economic* aspect concerns the opportunity to create stable sources of income for local entrepreneurs, to improve working conditions and access to the industry, to provide employment opportunities for the community. The *socio-cultural* aspect aims at providing more enjoyable experiences for tourists through more meaningful connections with local people and a greater understanding of local cultural and social issues, encouraging respect between tourists and hosts and providing access for physically challenged people. Following these issues, tourism sustainability does not appear to be anything measurable, but rather a principle to adhere to. In this view, some researchers and practitioners have even questioned its real utility. Many methodological approaches have been developed and proposed to provide a comprehensive approach for assessing tourism sustainability but, unfortunately, the complexity of the problem implies comprising several dimensions/indicators for which reliable data are generally lacking.

As reported by related works [12, 22], "an indicator is a variable which supplies information on other variables which are difficult to access and can be used as benchmarker to take a decision" or "alternative measures enable us to gain an understanding of a complex system so that effective management decisions can be taken that lead towards initial objectives". Introducing a composite fuzzy indicator makes us able to define an approximate algorithmic characterization of a complex phenomena such as sustainability: the sustainability assessment problem is a complex task, since it takes into account attributes (variables) whose nature and value are difficult to define. Hence we want to use a fuzzy-set based method to convert qualitative information into numeric aggregates, without costly and time-consuming parameter tuning which could hinder real-life applications. Besides, a fuzzy indicator would provide either an informative function by supplying simplified information about an *unmeasurable* criteria or a decision aid function to help to achieve the initial objectives.

In this work we propose a fuzzy approach for evaluating the level of sustainability of a tourist destination that takes into account the guidelines provided by WTO and European Union (EU) published works, and that is robust, since it allows the user to use different types of indicators to attribute the corresponding level of sustainability to a given area.

We compare our results with those derived both from the World Tourism Organization Environment Committee and the European Commission developed indicators for "*supporting industry decision-makers in the face of growing concern globally and locally about environmental quality*" and for "*addressing the links between the tourism industry and the environment, the impact of the industry on the environment, and the effects of social and natural environmental factors on the prosperity of the industry*" [10, 31]. A basic drawback of the former approaches lies in the fact that they account mainly for individual indicators. Moreover, the configuration of the proposed composite indices is arbitrary and the weightings are set in a subjective way. In our analysis weights are fixed according to the existing international benchmarks as reported in [10] or in the online Eurostat statistics database [9].

This chapter is organized as follows: Sect. 24.2 briefly reviews some of the literature on sustainable tourism. Next, we introduce our model in Sect. 24.3 and we detail the variables used for the computation of the sustainability index in Sect. 24.4. The experimental analysis is outlined in Sect. 24.5. In Sect. 24.6 we give some more insights on the advantages of implementing a flexible methodological framework to assess tourism sustainability and Sect. 24.7 concludes.

24.2 Sustainable Tourism: What Has Been Done So Far?

Proposing a suitable compromise between economic growth and tourism sustainability has been a crucial issue for many years, due to social, economic and cultural factors which may impact upon sustainable tourism [20].

Economic growth is traditionally attributed to an increasing production of goods/services, and the relation between their offer and environmental health is depicted by a concave curve showing an inverse relationship between services offering and environmental health. This can be formulated by the *Environmental Kuznets Curve* [13, 16] which postulates an inverse U-shaped relationship between per capita income and pollution. Tourism can cause pollution on both supply (e.g. construction activity) and demand (e.g. litter, rubbish, congestion) side. Furthermore, tourism exploits environmentally scarce resources whose availability might be threatened as a consequence of an uncontrolled development.

An accepted definition that reconciles environment protection with tourist fruition is based on the *carrying capacity* concept [5], which relates to the theoretical maximum number of individuals that a location can support. More in detail, the carrying capacity is *the maximum number of people that may visit a tourist destination at the same time, without causing destruction of the physical, economic, socio-cultural environment and an unacceptable decrease in the quality of visitors' satisfaction* [30]. The carrying capacity principle is used as a measure of the multifaceted tourism impacts on a destination and it represents a useful tool for spatial planning in a sustainable development perspective.

Although the definition appears intuitive, its application in tourism management [1, 2] is not straightforward: for example, when thinking of a mountain area, if it is simple to quantify the number of individuals that can walk down a single trail, it is much harder to assess the number of trails that might be opened in the same area without causing irreversible damage to flora and fauna. It is even harder to quantify how many people achieve a well-being condition or the optimal level of an area's global tourist offer to maximize consumer benefits with minimum environmental impact. Besides, it should be remarked that every tourist destination has its own features, thus the same indicator cannot be applied to different destinations in the same way.

Hence, the carrying capacity cannot be regarded as neither an objective nor as a neutral-based measure, due to the following reasons:

- the difficulty of managing variables which are not homogeneous;
- the intrinsic subjective nature of many variables;
- the lack of a universally accepted criterion for measuring and evaluating the environmental impact.

Earlier studies focussed only on economic or human issues [7, 11], while other aspects have been taken into account by Miller [21]: environment (physical and human), employment, financial leakages and customer's aspects (e.g. satisfaction levels).

A conceptual framework for assessing tourism sustainability is defined by Ko [15] that takes into account eight different dimensions: political, economic, socio-cultural, production related aspects, environmental impact, ecosystem quality, biodiversity and environmental policies. However, this approach shows two main limitations: first, the methodology proposed is adapted to the specific conditions of each tourist destination, making direct comparisons not possible; second, it relies upon hypothetical data which is generally unavailable.

Sustainable tourism assessment can be tackled by means of activity analysis by introducing the concept of the production frontier of a tourist destination as well as the concept of sustainable tourism efficiency [6]. This methodology takes into account only the economic and environmental aspects. All the above-mentioned works show that, despite many methodological advances in the development of tourism sustainability assessment procedures, there is still a lack of comprehensive indicators for supporting sustainable development strategies. Hence, a more suitable approach is needed as an operational and effective tool for tourism planning [4].

Fuzzy set theory [32] allows us to model and control complex dynamical processes. According to Zadeh's incompatibility principle, *as the complexity of a system increases, our ability to make precise and significant statements about its behaviour diminishes until a threshold is reached beyond which precision and significance (or relevance) become almost mutually exclusive characteristics* [32].

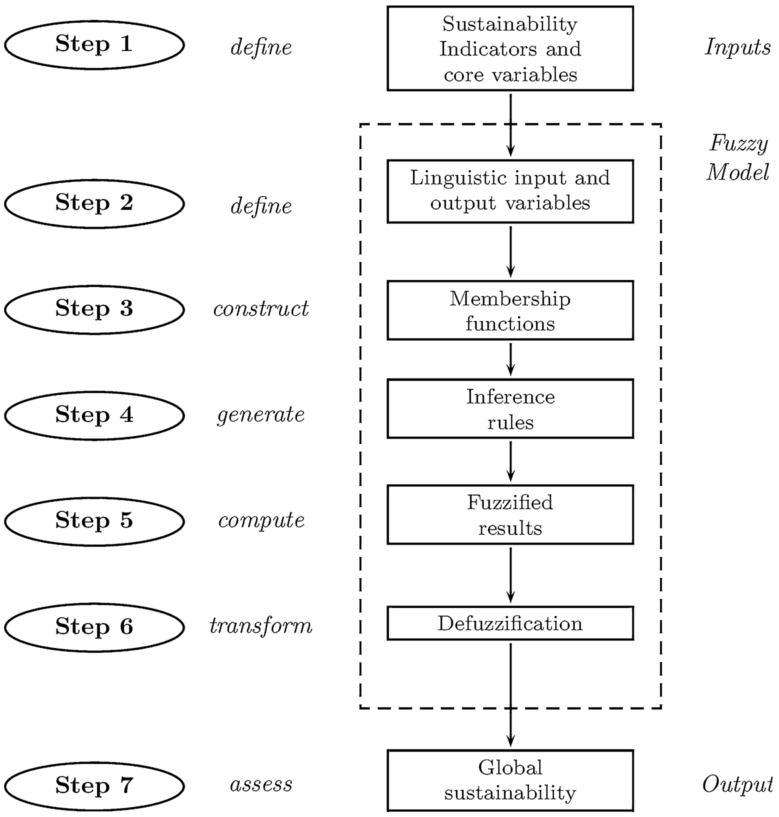


Fig. 24.1 The four subprocesses of our fuzzy inference model for sustainability assessment: fuzzification, inference, composition and defuzzification

24.3 A Fuzzy Model for Tourism Sustainability Assessment

In this section we are outlining our fuzzy model for the tourism sustainability assessment problem. The model is depicted in Fig. 24.1 as a combination of four components, sequentially connected to each other via ordinary input-output operations:

1. **Fuzzification**, given by the definition of input/output fuzzy variables and of the membership functions (steps 2 and 3);
2. **Inference**, given by the implementation of fuzzy control rules (step 4);
3. **Composition**, given by the aggregation of fuzzy relations (step 5);
4. **Defuzzification**, given by the conversion of the fuzzy output into a crisp value (step 6).

Sustainability indicators and core variables are defined in Sect. 24.3.1; the linguistic variables and membership functions are outlined in Sect. 24.3.2; fuzzy inference rules are defined in Sect. 24.3.3 and the defuzzification procedure is explained in Sect. 24.3.4.

24.3.1 Sustainability Indicators and Core Variables

Assessing the level of sustainability of a tourist destination is mainly based on dataset collection: structured data are used to define specific indicators, and these indicators need to be simple, robust, sensible to changes of environmental and economic patterns, and built on reliable and periodically updatable data.

Many indicators could be used for assessing the impact of tourism on environmental health and socioeconomic conditions, and in our model a high number of indicators would imply an exponentially increasing number of basic fuzzy rules. In order to avoid numerical problems and for better modelling purposes, the indicators must be normalized, and in what follows, the description of the normalization procedure is given.

Let us define sustainability as a function of the environmental integrity in a global systemic perspective. In what follows

- $T(V_j)$ represents the acceptable threshold for V_j based on sustainability performance benchmarks
- $\min(V_j)$ represents the minimum value for V_j
- $\max(V_j)$ represents the maximum value for V_j.

We define the normalized value $N(\upsilon_{i,j})$ of the ith individual observation υ relative to the V_j indicator as:

$$N(\upsilon_{i,j}) = \begin{cases} \frac{\upsilon_{i,j} - \min(V_j)}{T(V_j) - \min(V_j)}, & \text{per } \upsilon_{i,j} < T(V_j) \\ 1, & \text{per } \upsilon_{i,j} \geq T(V_j) \end{cases} \tag{24.1}$$

if $T(V_j)$ is a maximum,

$$N(\upsilon_{i,j}) = \begin{cases} 1, & \text{per } \upsilon_{i,j} \leq T(V_j) \\ \frac{\max(V_j) - \upsilon_{i,j}}{\max(V_j) - T(V_j)}, & \text{per } \upsilon_{i,j} > T(V_j) \end{cases} \tag{24.2}$$

if $T(V_j)$ is a minimum,

$$N(\upsilon_{i,j}) = \begin{cases} \frac{\upsilon_{i,j} - \min(V_j)}{\min T(V_j) - \min(V_j)}, & \text{per } \upsilon_{i,j} < \min T(V_j) \\ 1, & \text{per } \upsilon_{i,j} \in [\min T(V_j), \max T(V_j)] \\ \frac{\max(V_j) - \upsilon_{i,j}}{\max(V_j) - \max T(V_j)}, & \text{per } \upsilon_{i,j} > \max T(V_j) \end{cases} \tag{24.3}$$

if $T(V_j) \in [\min T(V_j), \max T(V_j)]$.

The use of a normalized domain is necessary for defining a linguistic term. By normalization the same linguistic term can be used in different linguistic variables.

24.3.2 Linguistic Variables and Membership Functions

High precision and high complexity are mutually exclusive concepts in the sense that there is an inverse relation between the complexity of a system and the precision with which it can be analysed. By applying linguistic variables it is possible to approximate systems which are too complex to be analytically characterized such as multidimensional real-life issues. Linguistic variables are variables whose values are not numbers but words or sentences in a natural or artificial language rather than a numerical one so that less specific (linguistic) characterization can apply [33].

A term-set is the totality of values of a linguistic variable which in principle could have an infinite number of elements. In our work the term-set associated to the **output** variable *Global Sustainability* (GS) is defined as GS = {Very Low, Low, Acceptable, High, Very High}, while the linguistic term set (LT) associated to each of the component variables (inputs) is LT = {Low, Medium, High}. Please notice that *Global Sustainability* is a composite categorical ordinal variable whose components are also categorical variables. The different components correspond to the different sustainability dimensions.

In terms of a generic variable V_i, a linguistic value such as *Low* represents a *restriction* on the values of the variable itself. By a compatibility function, each of these values is associated to a number in the interval [0,1] which represents its compatibility with the fuzzy restriction. The degree of compatibility, or grade of membership, is generally defined by $\mu_A(u)$ where A is a subset of the universe U and u is a generic element of U.

The most commonly used membership functions are:

- *Singleton* which takes value 1 only at a particular point of the universe of discourse U and 0 otherwise;
- *Triangular* which is a three point function defined by minimum, maximum and modal values;
- *Trapezoidal* which is specified by four parameters;
- *Bell* which is defined as:

$$\mu_A(u) = \begin{cases} 0, & \text{if } u < a \\ e^{-\frac{(u-b)^2}{2\sigma^2}}, & \text{if } a \le u \le c \\ 0, & \text{if } u > c. \end{cases} \tag{24.4}$$

The membership functions chosen are triangular and trapezoidal for the output (Fig. 24.2) and strictly trapezoidal for the inputs. Their use is motivated by their ease of application and because they provide us a representation of sustainability which is largely accepted by practitioners. As an example, the membership function for the output variable *Global Sustainability* is described as below:

$$\mu_{Very\_Low}(u_i) = \begin{cases} 1, & \text{if } 0 \le u_i \le 0.1 \\ \frac{0.3-u_i}{0.3-0.1}, & \text{if } 0.1 \le u_i \le 0.3 \end{cases} \tag{24.5}$$

$$\mu_{Low}(u_i) = \begin{cases} \frac{u_i-0.1}{0.3-0.1}, & \text{if } 0.1 \le u_i \le 0.3 \\ \frac{0.5-u_i}{0.5-0.3}, & \text{if } 0.3 \le u_i \le 0.5 \end{cases} \tag{24.6}$$

$$\mu_{Acceptable}(u_i) = \begin{cases} \frac{u_i-0.3}{0.5-0.1}, & \text{if } 0.3 \le u_i \le 0.5 \\ \frac{0.7-u_i}{0.7-0.5}, & \text{if } 0.5 \le u_i \le 0.7 \end{cases} \tag{24.7}$$

$$\mu_{High}(u_i) = \begin{cases} \frac{u_i-0.5}{0.7-0.5}, & \text{if } 0.5 \le u_i \le 0.7 \\ \frac{0.9-u_i}{0.9-0.7}, & \text{if } 0.7 \le u_i \le 0.9 \end{cases} \tag{24.8}$$

$$\mu_{Very\_High}(u_i) = \begin{cases} \frac{u_i-0.7}{0.9-0.7}, & \text{if } 0.7 \le u_i \le 0.9 \\ 1, & \text{if } 0.9 \le u_i \le 1. \end{cases} \tag{24.9}$$

24.3.3 Fuzzy Inference Rules

The basic rule of inference is a traditional two valued logical reasoning and it is called *Modus Ponens* (MP) [14, 27]: in the inference scheme "if X is A then Y is

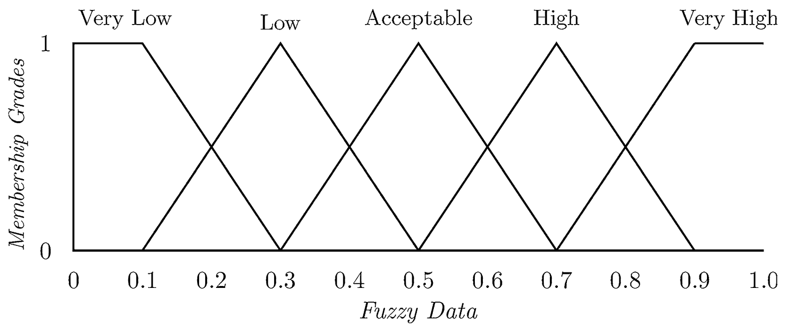


Fig. 24.2 Membership function for sustainability

B", the proposition "X is A" has to be observed to consider the proposition "Y is B". In fuzzy logic, a proposition "X is A'", that in natural language can be considered as similar (close) to the premise "X is A" can be observed to provide a conclusion "Y is B'" close to another conclusion "Y is B". This fuzzy inference scheme uses the so-called *Generalized Modus Ponens* (GMP) rule. A real world example of the application of this principle would be: "*This is a very traffic congested city*" (A') —"*If a city is traffic congested, then it is unsustainable*" (if A then B) —"*This is a very unsustainable city*" (B').

In general, a fuzzy inference system is a knowledge-based or rule-based system consisting of fuzzy *if-then* rules, i.e. statements in which some attributes are characterized by continuous membership function or possibility distributions. A fuzzy rule base, as a vehicle of knowledge representation, is therefore of great value when dealing with a complex concept such as sustainability. This structure of rules leads to a flexible, readable and user-friendly model representation, and it also allows an easy and effective computation without triggering lack of accuracy or precision.

Next step of the inference process is combining membership functions with the control rules to derive the fuzzy output. A relation $R(u, v)$ of two symbolic variables names can be considered a fuzzy set with two dimensional membership function:

$$\mu_R(u, v) = f(\mu_A(u), \mu_B(v))$$

where f is a fuzzy implication function. Among the different implication functions proposed in literature [17, 18, 32], some of the most widely applied are:

Min (Mamdani): $$\mu_R(u, v) = \min\{\mu_A(u), \mu_B(v)\} \tag{24.10}$$

Product (Larsen): $$\mu_R(u, v) = \mu_A(u) \times \mu_B(v) \tag{24.11}$$

Max-Min (Zadeh): $$\mu_R(u, v) = \max\{\min\{\mu_A(u), \mu_B(v)\}, 1 - \mu_A(u)\} \tag{24.12}$$

$$\text{Standard implication:} \mu_R(u, v) = \begin{cases} 1, & \text{se } \mu_A(u) \leq \mu_B(v) \\ 0, & \text{se } \mu_A(u) > \mu_B(v). \end{cases} \quad (24.13)$$

In our study we use the *max-min* compositional rule of inference [19]: given a premise A' and a rule like "if A then B", $R_{A \to B}$, the conclusion $B' = A' \circ R_{A \to B}$ is given by:

$$\mu_{B'}(v) = \max_u \min \{\mu_{A'}(u), \mu_R(u, v)\}. \quad (24.14)$$

The fuzzy inference process consists of two stages: the implication and aggregation processes. By the implication process, as we said above, a fuzzy set (*consequent*), represented by a membership function, is obtained for each rule and for given controller inputs (*antecedent*), while aggregation is the process by which all the consequents are combined into a single fuzzy set.

24.3.4 Defuzzification

The combination of inputs, output membership functions and fuzzy rules characterizes the fuzzy inference scheme, in which the fuzzy conclusion is represented by a fuzzy linguistic variable which needs to be converted into a single number (crisp variable). This is made by *defuzzification*, through which the combined fuzzy set resulting from the aggregation process is transformed into a crisp output. Several defuzzification techniques have been proposed in the literature, and in our approach we have adopted the *Centroid Method* (also referred to as *centre of area* or *centre of gravity*), which is one of the most commonly used defuzzification methods in Mamdani type systems [19, 23, 26]. In Mamdani inference, the consequent of each rule is a fuzzy set, and the consequent fuzzy set of each rule is reshaped by a single matching number. Defuzzification is then required to aggregate all the reshaped fuzzy sets into a single fuzzy set.

A centroid of the fuzzy controller output variable S and linguistic variable A is computed as:

$$\mathrm{S} = \frac{\sum_k S_k \mu_A(S_k)}{\sum_k \mu_A(S_k)}, \quad (24.15)$$

having defined with S_k the k-th element of the output fuzzy sets, and with $\mu_A(S_k)$ its corresponding membership degree. Please note that there exist *fuzzy inference methods* that do not need defuzzification. Experiments performed with a specimen of these models [25] lead to comparable results, and they are not being reported in what follows.

24.4 Tourism Sustainability in Sicily and Model Variables

The fuzzy model introduced in Sect. 24.2 is applied to assess the tourism sustainability level of provinces belonging to Sicily, which is a region featuring tourist attractiveness, thanks to its natural, cultural and historical heritage. To this end, we have used data provided by the *Statistical and Economic Analysis Service* and the *Tourism Observatory* of the Sicilian Regional Authority [24] that shows touristic features on a provincial scale. Touristic attractions are well spread over provinces: Siracusa is one of the great cities of the Greek diaspora, featuring its old town (named *Ortigia*) and its fifth-century theatre; in Agrigento one may find Greek temples, as well as in Selinunte and Segesta, both in the province of Trapani. Taormina lays in the province of Messina, and Etna, which is the highest active volcano in Europe, belongs to the province of Catania. Piazza Armerina, with its Villa Imperiale del Casale (one of the best preserved and best known Roman villas) is located near Enna, and eventually the regional capital, Palermo, is in the UNESCO heritage list. Other important tourist attractions and destinations are the Aeolian, Egadi and Pelagie archipelagos consisting of attractive and extremely varied islands and which are geographically located, respectively, in the province of Messina, Trapani and Agrigento (Fig. 24.3).

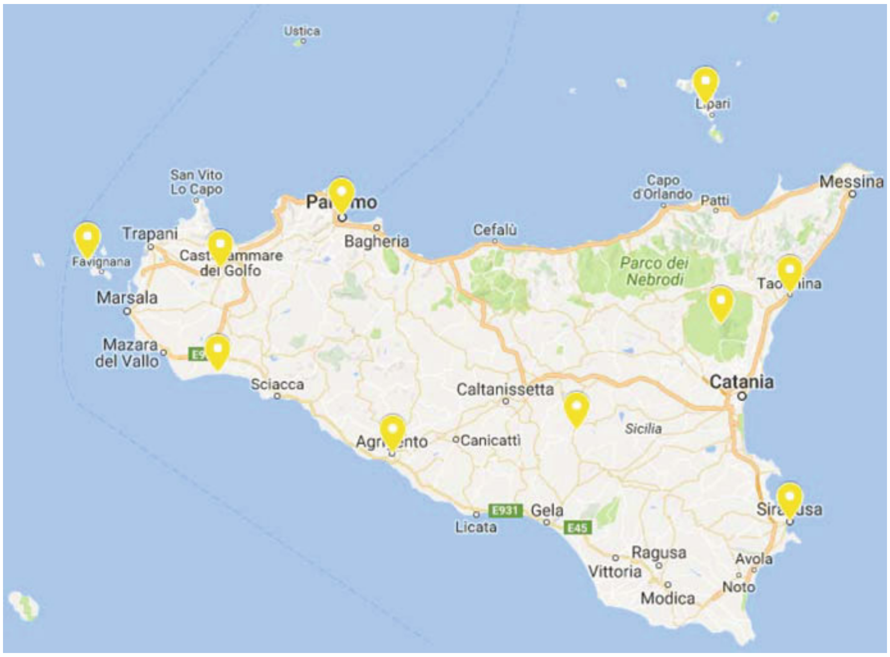


Fig. 24.3 Map of the most attractive places to visit in Sicily

The province showing the highest tourist vocation is Messina (roughly one million visitors per year and more than 10,000 daily presences); Palermo follows with a comparable flow of arrivals but with a lower number of overnight stays (i.e. 1 day less than Messina). The province featuring the highest average length of stay is Ragusa, followed by Messina, Trapani and Siracusa. The tourist density ratio (TDR: tourists to land) and the tourist intensity ratio (TIR: tourists to residents) indicate that the province of Messina is the one with the most significant tourism impact as it shows high levels of tourist flows with respect to both land area and resident population.

In our fuzzy model the information and representation of environmental condition is provided by a set of indicators which are calculated on the basis of the available information: numerical and qualitative values are the physical domain for the linguistic variables of the model whose strength is to reduce the possibility of wrong decisions caused by imprecise information and to minimize the impact of subjectivity in evaluation of composite indices.

Below is given a detailed description of the indicators used for the purpose:

V_1: receptive tourism establishments per km^2. It represents the concentration of receptive tourist structures in an area and therefore it provides an estimate of the buildings impact on environment;

V_2: social impact. It is computed as the number of tourists divided by the number of residents and is referred to as the *penetration ratio*;

V_3: number of tourists per square kilometre. It is a tourist density index and it measures the anthropic pressure degree over an area;

V_4: seasonality. It is assessed by the number of arrivals in the peak months.

The main statistics of the dataset used to compute the selected indicators are shown in Table 24.1 for all provinces of Sicily.

Table 24.1 Descriptive statistics of indicators used in our model for sustainability assessment

	Mean				Std			
	V_1	V_2	V_3	V_4	V_1	V_2	V_3	V_4
Agrigento	5.590	0.449	332.41	0.270	0.68	0.026	19.49	0.028
Caltanissetta	0.95	2.130	60.48	0.199	0.11	0.070	1.96	0.038
Catania	6.15	0.596	501.91	0.209	0.17	0.016	14.17	0.003
Enna	0.79	1.502	48.28	0.176	0.20	0.330	14.68	0.013
Messina	12.57	0.163	1240.75	0.313	0.85	0.007	58.20	0.015
Palermo	6.89	0.369	670.86	0.225	0.51	0.008	16.33	0.012
Ragusa	7.85	0.349	532.61	0.331	0.70	0.010	15.36	0.012
Siracusa	6.40	0.341	554.94	0.282	0.63	0.033	54.10	0.020
Trapani	8.35	0.305	579.27	0.354	1.20	0.047	93.86	0.020

Rows refer to all provinces belonging to Sicily

The following four linguistic input variables have been therefore defined:

l_1, tourist establishments density,

l_2, social impact,

l_3, anthropic pressure,
l_4, seasonality.

For each generic variable l_i, $i = 1, 2, 3, 4$, have been associated to the linguistic terms: y_{i1} = low (L), y_{i2} = medium (M), y_{i3} = high (H). It is worth to remark, once more, that the thresholds come from studies commissioned by the European Union as part of the project on "Indicators for the Sustainable Management of Destinations". The corresponding membership functions are given in Figs. 24.4, 24.5, 24.6, 24.7.

For the sake of clarity, if we look, for example, at the left side plot of Fig. 24.4 it can be observed that for every normalized values less than or equal to 0.2, corresponding to an original value of 2.69 beds per square kilometre, the membership function value is equal to one. This means that in terms of our first indicator, observed values less than or equal to the former threshold imply a state of "low" impact on sustainability with a high confidence level. The same state applies also for all values between 0.2 and 0.25 with a decreasing confidence level as the input increases. The maximum membership value for the "medium" and "high" states (central and right side) is assigned, respectively, to the intervals between 0.2 and 0.6 and between 0.6 and 1.

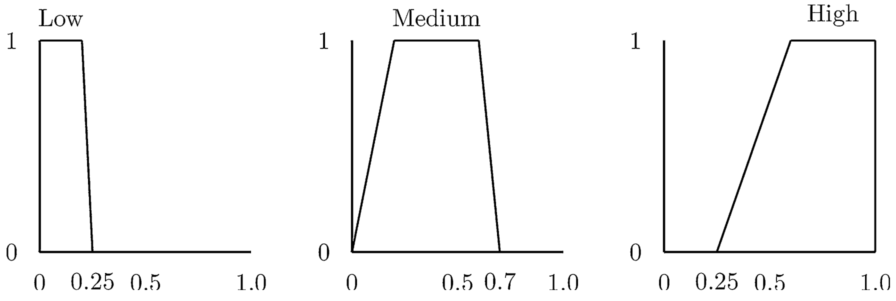


Fig. 24.4 Membership function for *tourist establishments density*

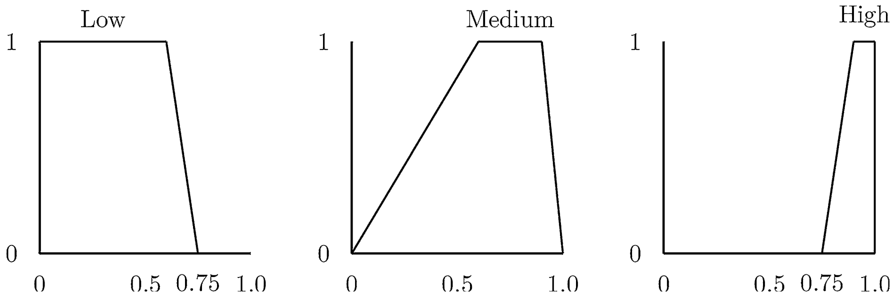


Fig. 24.5 Membership function for *social impact*

A surface representation of the above-mentioned rules and membership functions is given in Fig. 24.8.

As we used four linguistic variables with three terms each, the fuzzy rule base, i.e. the definition of the rules that correlate the input variables to the output ones, is given by $3^4 = 81$ rules (Table 24.2). *For the inference process we have used the* Min *operator to make a conclusion of each rule and the* Max *operator for the aggregation of the partial conclusions.*

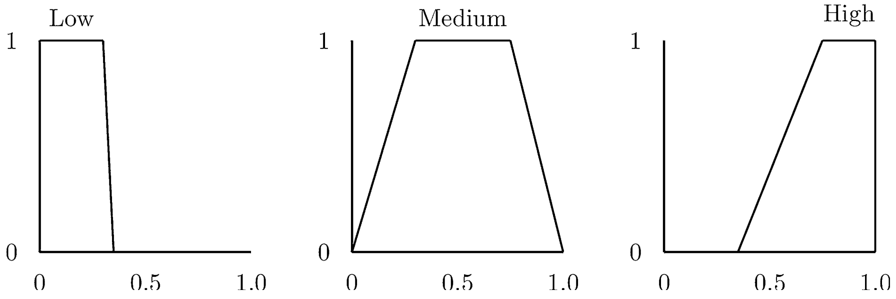


Fig. 24.6 Membership function for *anthropic pressure*

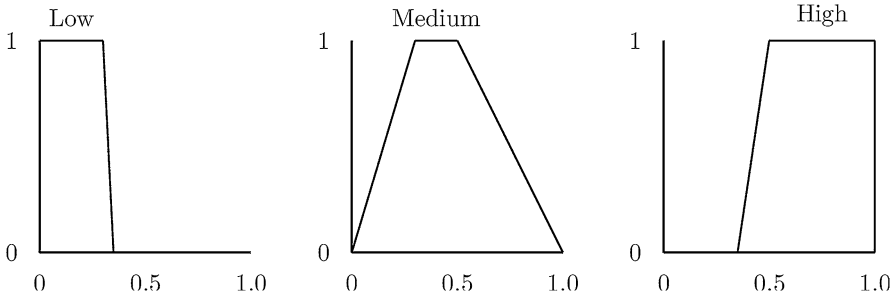


Fig. 24.7 Membership function for *seasonality*

24.5 Experimental Analysis

The goal of our analysis is to compute our Fuzzy Sustainability Index (FS) and to compare it to the World Tourism Index [31] and to the European Tourism Index [10] for measuring the sustainability level of a destination. All indices have been computed with MATLAB on a Laptop equipped with an AMD Kabini E1-2100 1 GHz Dual Core, with 4 GB RAM and running Ubuntu. Computation times have been below 1 s for each indicator and territorial area (province).

As for our FS Index, from the WTO and EU indicators we have selected the ones which link to the three pillars related to tourist pressure and sustainability of a destination, i.e. environmental, economic and social.

The main shortcoming of both EU and WTO indicators is that they are composed of several metrics that has to be monitored individually. In our work, each indicator is determined by the combination of three components: the first takes into account the economic impact from tourism in the local economy and with local businesses, the second takes into account the impact of tourism on local communities and their cultural heritage, the third takes into account the environmental stress also as a result of the *effect of use*. These three components are converted into numeric values by using metrics and ratios introduced by EU and WTO. More in detail, for the EU index the economic component is measured by the Average Length of Stay (ratio between the total number of overnight stays and the number of arrivals) and the Occupancy Rate; the social component is given by the number of tourists per inhabitant; and the environment component is given by Solid Waste Volume. Although there are some other indicators on environmental related issues such as those on transport, solid waste management, sewage treatment, bathing water quality and energy usage, they are relatively harder to measure due to lack of information. Indeed, assuming a positive relationship between tourist volume and environmental decay, we can consider our tourist density index as another indirect measure of tourism environmental impact. For the WTO index the economic component is given by the Total Number of Tourist Presences; the social component is given by the Tourism Intensity Rate (visitors per host population) and the environment component is given by the Tourist Density Rate (tourists per square kilometre) and the Tourist Accommodation Establishments Density Rate.

In order to obtain an aggregate measure, we need to weight the three indicators: this leads to different aggregated values, depending on the chosen weights, and can be seen in Fig. 24.9, in which we have plotted in histograms the aggregate

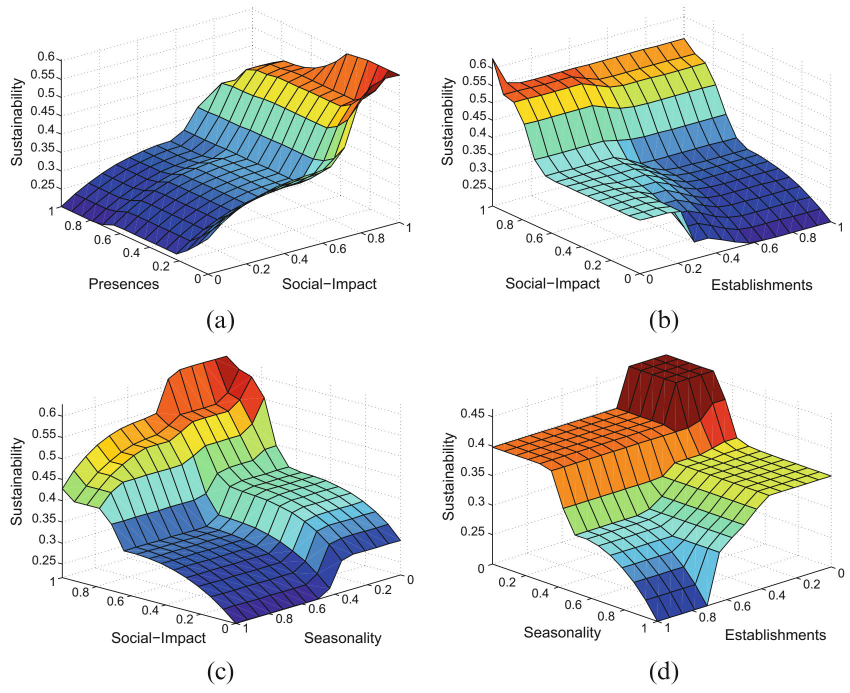
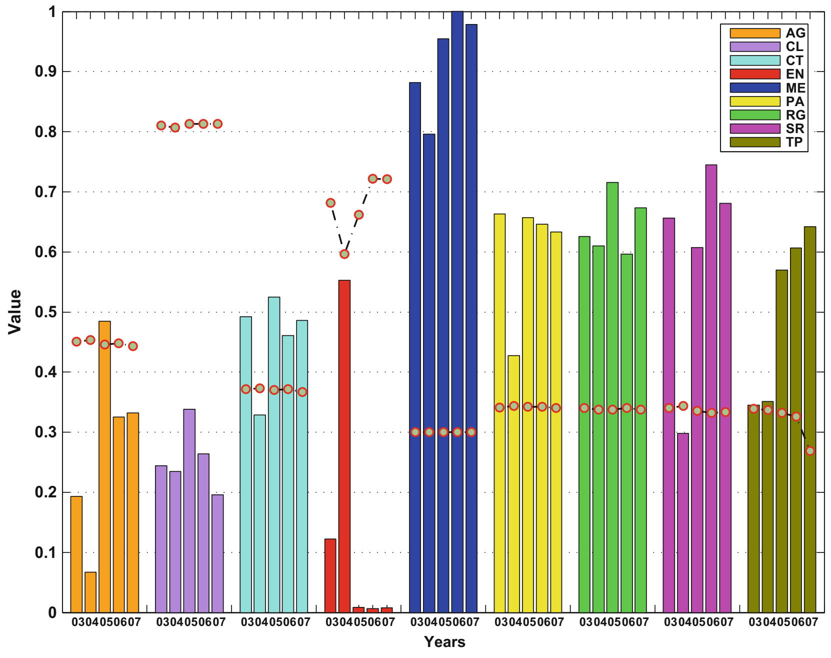
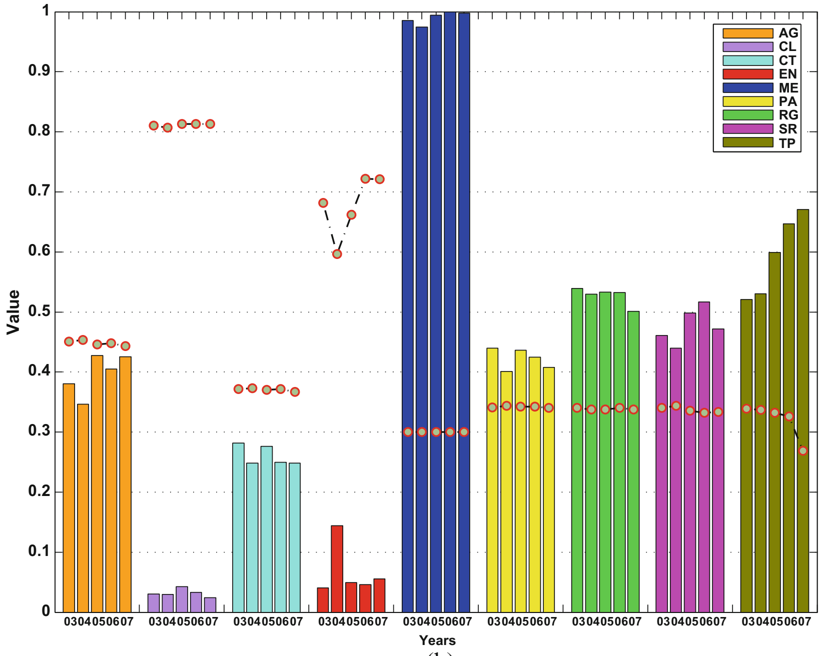


Fig. 24.8 Rules surface representation: sustainability as a function of (**a**) presences and social impact, (**b**) social impact and establishments, (**c**) social impact and seasonality, (**d**) seasonality and establishments



(a)



(b)

Fig. 24.9 FS (circles) and EU values (histograms) obtained by setting different weights combinations among the economic, environmental and social metrics. Weights values are reported below plots. (**a**) Economic = 0.8; Social = 0.1; Environmental = 0.1. (**b**) Economic = 0.1; Social = 0.8; Environmental = 0.1

measures computed for EU, for each province (labelled *AG*, *CI*, *CT*, etc.) over the period of observation (years 2003, 04, 05, etc., labelled on the x-axis for every histogram group) corresponding to different weights values, along with our Fuzzy Index (shown in connected circles).
We remark that different weights lead to different values, hence it is difficult to decide a priori the weights that may correspond to a desired compromise among the three criteria. In this context, the introduction of the fuzzy indicator is of great help, for it translates qualitative information into numeric aggregates, without costly and time-consuming parameter tuning.

We may also analyse the impact of the three components on the fuzzy measure. In particular, we want roughly to assess how much a given sub-dimension accounts for the resulting Fuzzy Sustainability Index.

This can be done with a correlation analysis, reported in Table 24.3, in which the overall Pearson correlation coefficient ρ between the FS and, respectively, WTO and EU indices are reported. As for EU and WTO weights have been used for their computation, which are indicated in the triplet $[i\text{–}j\text{–}k]$, in which i represents the

Table 24.2 Fuzzy rule base: IF-THEN rules stored in the fuzzy system

IF $\mathbf{V_1}$	AND $\mathbf{V_2}$	AND $\mathbf{V_3}$	AND $\mathbf{V_4}$	THEN **GS**
High	High	High	High	Low
High	Medium	High	High	Low
High	Low	High	High	Very low
High	High	Medium	High	Acceptable
High	Medium	Medium	High	Low
High	Low	Medium	High	Very low
High	High	Low	High	Acceptable
High	Medium	Low	High	Acceptable
High	Low	Low	High	Low
High	High	High	Medium	Acceptable
High	Medium	High	Medium	Low
High	Low	High	Medium	Low
High	High	Medium	Medium	High
High	Low	Low	Medium	Low
High	High	High	Low	Acceptable
High	Medium	High	Low	Acceptable
High	Low	High	Low	Low
High	High	Medium	Low	High
High	Medium	Medium	Low	Acceptable
High	Low	Medium	Low	Low
High	High	Low	Low	High
High	Medium	Low	Low	High
High	Low	Low	Low	Acceptable
...	...	...	...	...
Low	Medium	Low	Low	Very high
Low	Low	Low	Low	High

weight associated to the economic component, j represents the weight associated to the social component and k represents the weight associated to the environmental component. For the sake of completeness the correlation between EU and WTO is reported. We recall that EU and WTO are stress indices, hence their correlation with FS, which is a sustainability index, is supposed to be negative. Indeed, the higher the environmental stress, the higher the EU and WTO indices' values, whereas the healthier the environment, the higher the FS index value.

Table 24.3 Pearson correlation analysis among EU, WTO and fuzzy indices

Weights	$\rho(WTO, FS)$	$\rho(EU, FS)$	$\rho(WTO, EU)$
[1–0–0]	−0.65	−0.64	0.66
[0–1–0]	−0.77	−0.77	0.95
[0–0–1]	−0.85	−0.80	0.87
[0.8–0.1–0.1]	−0.71	−0.71	0.77
[0.1–0.8–0.1]	−0.79	−0.79	0.97
[0.1–0.1–0.8]	−0.87	−0.81	0.93
[0.5–0.5–0.0]	−0.74	−0.78	0.87
[0.5–0.0–0.5]	−0.84	−0.77	0.91
[0.0–0.5–0.5]	−0.85	−0.80	0.98
[0.6–0.3–0.1]	−0.75	−0.76	0.85
[0.6–0.1–0.3]	−0.79	−0.76	0.87
[0.3–0.6–0.1]	−0.78	−0.79	0.94
[0.3–0.1–0.6]	−0.87	−0.80	0.95
[0.1–0.3–0.6]	−0.86	−0.81	0.97
[0.1–0.6–0.3]	−0.83	−0.80	0.98

In Table 24.3 we can see that FS convey information in some way similar to the WTO and EU indexes given the existing correlations.

By looking the first three rows we may also say that the relative importance of the component corresponding to the unitary weight may be assessed by looking at its correlation with FS, hence we may argue that the qualitative information contained in FS influenced mostly by the environmental aspect, while the economic component appears to be the less relevant to compute the index, and the most inversely related to it (see Fig. 24.9a, province *EN* (Enna)). This latter observation gives quantitative evidence to the environmental sustainability threat represented by an area's economic development.

We also remark that the similar profile for both WTO and EU does not hold for Caltanissetta ed Enna. This is because EU, contrarily to WTO, takes into account the two variables “average length of stay” and the “occupancy rate” which reach levels comparable to those of the other provinces.

Please notice that the fuzzy indicator shows stable values over time for all provinces. This does not hold for Enna, and this is due to the low touristic vocation of the area, for which the incremental impact of visitors has a greater impact on sustainability with respect to more touristic destinations as Messina and Trapani, whose impact on the territory is far bigger but not subject to great changes over time. This is an important point: it shows that the qualitative information used to compute FS is able to provide additive information in areas showing weak touristic presence, in which small variations of touristic flows may have a big impact on the area’s vocation. This phenomenon can be hardly found out via the other two indicators, because of their sensitivity to weights values.

24.6 Discussion

In this paper, we have seen that the term tourism sustainability means refers to interacting activities which simultaneously operate in three main fields: the environmental, the social/cultural and the economic ones. The principles of sustainability can apply either to any type of tourism mass or specialty, city, beach, or wilderness or to any business involved such as transport, lodging, tours, agencies. Small lodges and large hotels can be irresponsible and unsustainable in the same way and in some cases the former ones even more than the latter ones. This is why there is a growing need for internationally recognized indicators for sustainability that could be effectively locally adapted and applicable.

An internationally accepted sustainable tourism certification is of great value as, first, it could effectively distinguish sustainable tourism businesses from others; second, it would produce benefits for certified businesses, consumers, governments, the environment and local communities. In terms of government issues, a broadly accepted certification form of sustainable tourism accreditation would strengthen the market position, raises industry standards in health, safety, environment and social stability, lowers the regulatory costs of environmental protection. By requiring economic benefits to communities, it could also reduce poverty, especially in rural area [3].

However, the great number of different types of certification has generated a big confusion and raised the question on the reliability of these procedures.

Starting from these motivations, we propose a model-based approach using a fuzzy logic framework and provide a simple tool to handle the complexities surrounding tourism sustainability.

24.7 Conclusion

Tourism needs environmental quality, but it may produce adverse effects on the environment. Hence, one should tackle the problem of the environment preservation by adopting a suitable and rational approach to tourism development and management. Controlling and quantifying the level of environmental health and sustainability of a given territory is not a simple task: on the one hand people and governments pay more and more attention on environment-related issues; on the other hand balancing environmental limits with tourism development is a tough matter to deal with, due to the complexity of all the interconnected and dynamic variables involved.

Moreover, all the strategies and procedures undertaken to protect environmental quality are mainly conceived for prescriptive purposes rather than for management approaches. Hence, a tourism sustainability assessment model would be of great value as a management tool, providing a deeper knowledge of the potential impact that could be caused by political and regulatory actions. An assessment model helps decision-makers to evaluate the impact of new regulations on the tourism industry and to select those ones based on an optimal control framework.

Classical methods have limitations due to the imprecision and uncertainty inherent in defining and assessing individual indicators. Moreover, since each tourist destination has distinctive features, applying the same measurement standards and limit values may cause a large inaccuracy estimation.

On the bases of the aforementioned considerations, we have introduced a fuzzy logic approach to assess, monitor and manage the sustainability of a tourism destination. The method has been applied, on a provincial scale, to Sicily (Italy), because of the internationally recognized tourist vocation of the region. The proposed methodology offers useful insights: it allows to handle the complexity of the diverse sustainability issues involved; it helps to manage both quantitative and qualitative data and, eventually, it is easy to use and to interpret.

A direction for future research would be merging our fuzzy model with a neural network controller in order to combine the advantages of a fuzzy approach with the learning capabilities of neural networks which could be useful for fuzzy rules refinement purposes.

Acknowledgement This work has been partially supported by the Italian Government Prin Project #2015XAPRKF.

References

1. J. Andria and G. di Tollo. Clustering local tourism systems by threshold acceptance. In *Applications of Evolutionary Computation - 18th European Conference, EvoApplications 2015, Copenhagen, Denmark, April 8–10, 2015, Proceedings*, pages 629–640, 2015.
2. J. Andria, G. di Tollo, and R. Pesenti. Detection of local tourism systems by threshold accepting. *Computational Management Science*, 12:559–575, 2015.

3. Amos Bien. *A simple user's guide to certification for sustainable tourism and ecotourism*. Center for Ecotourism and Sustainable Development, 2008.
4. R. Butler. The concept of a tourist area cycle of evolution: Implications for management of resources. *Canadian Geographer / Le Géographe canadien*, 24(1):5–12, 1980.
5. H. Coccossis, A. Mexa, A. Collovini, A. Parpairis, M. Konstandoglou, J. van der Straaten, J. van der Borg, and I. Trumbic. Defining, measuring and evaluating carrying capacity in European tourism destinations. Technical report, Laboratory of Environmental Planning. The University of Aegean, 2001.
6. M.F. Cracolici, M. Cuffaro, and P. Nijkamp. Tourism sustainability and economic efficiency - a statistical analysis of Italian provinces. Serie Research Memoranda 0023, VU University Amsterdam, Faculty of Economics, Business Administration and Econometrics, 2009.
7. Larry Dwyer, Peter Forsyth, and Ray Spurr. Evaluating tourism's economic effects: new and old approaches. *Tourism Management*, 25(3):307–317, 2004.
8. Jody Emel, Paul G. Lewis, and James O. Wheeler. Is capitalism sustainable? Political economy and the politics of ecology. Martin O'Connor, editor; Privatopia: Homeowner associations and the rise of residential private government. Evan McKenzie; ground truth: The social implications of geographic information systems. John Pickles, editor. *Urban Geography*, 18(1):90–93, 1997.
9. European Statistical System (ESS). Eurostat statistics database. http://ec.europa.eu/eurostat/data/database.
10. European Commission. Study on the feasibility of a European tourism indicator system for sustainable management at destination level. http://ec.europa.eu/growth/sectors/tourism/offer/sustainable/indicators/, 2013.
11. Douglas C. Frechtling. The tourism satellite account: foundations, progress and issues. *Tourism Management*, 20(1):163–170, 1999.
12. R. Gras, M. Benoit, and JP. Deffontaines. *Le Fait Technique en Agronomie, Activité Agricole, Cocepts et Méthodes dÉ́tude*. L'Hamarttan, Paris France, 1989.
13. G. M. Grossman and A. B. Krueger. Environmental impacts of a north American free trade agreement. Working Paper 3914, National Bureau of Economic Research, November 1991.
14. A. Grzegorczyk. An outline of mathematical logic. fundamental results and notions explained with all details. Synthese Library. Vol. 70. Dordrecht-Holland – Boston-U.S.A., 1974.
15. Tae Gyou Ko. Development of a tourism sustainability assessment procedure: a conceptual approach. *Tourism Management*, 26(3):431–445, 2005.
16. S. Kuznets. Economic growth and income inequality. *The American Economic Review*, 45(1):1–28, 1955.
17. P. Martin Larsen. Industrial applications of fuzzy logic control. *International Journal of Man-Machine Studies*, 12(1):3–10, 1980.
18. E. H. Mamdani. Application of fuzzy logic to approximate reasoning using linguistic synthesis. *IEEE Transactions on Computers*, C-26(12):1182–1191, Dec 1977.
19. E.H. Mamdani and S. Assilian. An experiment in linguistic synthesis with a fuzzy logic controller. *International Journal of Man-Machine Studies*, 7(1):1–13, 1975.
20. V. Middleton and R. Hawkins. *Sustainable tourism: A marketing perspective*. Oxford: Butterworth-Heinemann, 1998.
21. G Miller. The development of indicators for sustainable tourism: Results of a Delphi survey of tourism researchers. *Tourism Management*, 22(4):351–362, 2001.
22. G. Mitchell, A. May, and A. McDonald. PICABUE: a methodological framework for the development of indicators of sustainable development. *International Journal of Sustainable Development & World Ecology*, 2(2):104–123, 1995.
23. Leszek Rutkowski. *Computational Intelligence: Methods and Techniques*. Springer Publishing Company, Incorporated, 1st edition, 2008.
24. Regione Sicilia. *Osservatorio Turistico Regione Siciliana*, 2009 (accessed April 29, 2016). https://osservatorioturistico.regione.sicilia.it/public/default.
25. M. Sugeno. *Industrial applications of fuzzy control*. Elsevier Science Pub. Co., 1985.

26. M. Sugeno. An introductory survey of fuzzy control. *Information Sciences*, 36(1):59–83, 1985.
27. P. Suppes. *Introduction to logic. New York*. D. Van Nostrand Company, New York, 1957.
28. The World Tourism Organization (UNWTO). UNWTO annual report 2015. http://cf.cdn.unwto.org/sites/all/files/pdf/annual_report_2015_lr.pdf/.
29. World Tourism Organization (WTO). The concept of sustainable tourism. http://sdt.unwto.org/content/about-us-5/. (accessed on 3 October 2016).
30. World Tourism Organization (WTO). Saturation of tourist destinations: Report of the secretary general. Technical report, World Tourism Organization, 1981.
31. World Tourism Organization (WTO). Indicators of sustainable development for tourism destinations: A guidebook. Technical report, World Tourism Organization, 2004.
32. L.A. Zadeh. Outline of a new approach to the analysis of complex systems and decision processes. *IEEE Transactions on Systems, Man, and Cybernetics*, 3(1):28–44, 1973.
33. L.A. Zadeh. The concept of a linguistic variable and its application to approximate reasoning. *Information Sciences*, 8(3):199–249, 1975.

Chapter 25
New Ideas in Ranking for Personalized Fashion Recommender Systems

Heri Ramampiaro, Helge Langseth, Thomas Almenningen, Herman Schistad, Martin Havig, and Hai Thanh Nguyen

Abstract Fashion is an area that is in constant growth. The proliferation of social media and the Web, in general, has made e-shopping, thus corresponding recommender systems, increasingly important. Fashion recommender systems is a related area that we focus on in this chapter. More specifically, we present how recommender systems are used in online fashion stores to enhance the user experience and increase sales. In addition, we look at challenges the fashion domain specifically faces. We exemplify solution strategies by considering the SoBazaar system, including showing how we built a recommendation approach for the system and discussing results from our experiments. The results from these experiments demonstrate the effectiveness and viability of our method.

Keywords Deterministic fashion recommender · Goal-oriented recommendation · Recommendation system · Stochastic fashion recommender

25.1 Introduction and Motivation

The proliferation of web-based applications and social media, such as movie and music streaming, e-commerce, and social networking, has given recommender systems an important role within both commercial and academic settings. Increasingly, people replace shopping at local stores with web-shops to buy products. This has given e-commerce companies like Amazon and eBay good growth opportunities,

H. Ramampiaro (✉) · H. Langseth · T. Almenningen · H. Schistad · M. Havig
Department of Computer and Information Science, Norwegian University of Science and Technology, Trondheim, Norway
e-mail: heri@idi.ntnu.no; helgel@idi.ntnu.no

H. T. Nguyen
Department of Computer and Information Science, Norwegian University of Science and Technology, Trondheim, Norway

Telenor Research, Trondheim, Norway
e-mail: HaiThanh.Nguyen@telenor.com

P. Moscato, N. J. de Vries (eds.), *Business and Consumer Analytics: New Ideas*,
https://doi.org/10.1007/978-3-030-06222-4_25

but also some new challenges. One important challenge is to make customers happy all the time. This has often been met by providing a huge variety of products. At the same time, the more products that are available, the more users will have difficulties in finding suitable items. Recommendation systems help users find and select items to overcome such a challenge.

Being an e-commerce domain, fashion e-commerce is a fast growing area in online shopping. The fashion domain has, however, several interesting properties which differ from most other recommender systems domains. It can be characterized as being specifically related to clothes, popularity, time, and cultural grouping. One of the main drivers of fashion is the need for *belonging*, and for individuals to share a common thought or opinion [44]. Hanf and Wersebe [16] argue that customers are rational with respect to price and quality. Moreover, the forming of a subculture happens through individuals seeking out other individuals with similar tastes in a variety of aspects [42]. Finally, brands greatly affect what the consumer purchases. A study done on the behaviour of consumers showed that knowing the brand of two almost identical products made the consumer crowd shift towards the more well-known brand [24].

With this in mind, we carried out a case study to gain a deeper understanding of fashion recommendation in general. We do a comprehensive review of published work related to recommender systems, while focusing on fashion recommendation. The main goal is to investigate how the characteristics of the fashion domain affect the choice of effective recommendation methods. In addition, we aim at giving deeper insights about how fashion is different from the traditional recommendation domain, and why traditional methods do not work. We carry out this research using users' interactions with a smartphone app as a starting point. The ultimate objective is to provide the users with separate personalized streams of items they would most likely buy. Thus, successful recommendation systems cannot rely on users explicitly rating items, but rather provide the ability to guide users through the rich history of interaction between the user and the system. This means that the system may have to rely solely on *implicit feedback*. Therefore, the user's preferences are to be automatically inferred from her/his behaviour [31, 32]. However, since the user's purchase history can be sparse and noisy, the approach must have a way to handle sparsity and low quality data.

In summary, the main contributions of this chapter are as follows. First, we present an analysis of the fashion domain focusing on recommender systems. As part of this we identify the properties and challenges of the fashion domain. Second, we do a comprehensive review of the literature and existing solutions for fashion recommender systems. Third, we present a case study based on our work development of an approach towards more personalized fashion recommendation.

Following this, the rest of this chapter is organized as follows. Section 25.2 provides an overview of recommender systems in general. Section 25.3 discusses existing recommender system approaches, methods, and systems for fashion. Section 25.4 discusses the aforementioned case study, including experiments with our own developed approach. Finally, Sect. 25.5 concludes the chapter and presents the remaining challenges that remain to be solved.

25.2 Recommender Systems: An Overview

Recommender systems as a research topic has been an important topic since the introduction of the first collaborative filtering system, Tapestry [14]. Recommender systems now play an important role in many of the most popular web-sites such as Amazon, YouTube, Netflix, TripAdvisor, Last.fm, and IMDb. In its most common formulation, the recommendation problem can be reduced to the problem of predicting the preference/rating of items that have not been seen by a user. Usually, this prediction is based on the estimation of ratings using one or more of the following assumptions:

- You are like your friends
- You are like people who do similar things that you do
- You like things that are similar to things you already like
- You are influenced by experts and the opinions of others.

Once we have estimated the ratings, we can recommend the items having the highest rating to the user. Recommending an item to a user can be compared to various decision-making processes, where we want to help a user decide, e.g., what items to buy, what music to listen to, or what online news or books to read. The overwhelmingly large amount of available items makes such aid crucial. Hence, the main goal with a recommender system is to reduce a user's burden in the process of finding items matching his/her *preferences*. Recommender systems are usually classified into the following categories, based on the available information about the users, user's preferences (e.g., ratings), and items [1].

- Content-based recommendations: The user will be recommended items with similar content to the ones the user preferred in the past;
- Collaborative recommendations: The user will be recommended items that people with similar tastes and preferences have liked in the past;
- Hybrid approaches: Combinations of collaborative and content-based methods.

25.2.1 Basic Methods for Recommender Systems

Perhaps the simplest and most obvious approaches to recommender systems are the *content-based* systems. In its most basic form, the key idea is to keep track of what a user likes, and looks for patterns in the descriptions that can be used for prediction, thereby building on the assumption that a user will continue to like items that are similar to those he/she already enjoyed. In this setup a text document may, for instance, be represented using its document or word embedding [23], while a TV-show may be defined by a feature-vector containing genre, actors, and director. In the TV-domain, a content-based system may tell that many users like political drama and enjoy thrillers starring a specific actor. The information like this can be used by a company like Netflix to produce specific series for its users. The series

House of Cards starring Kevin Spacey is an example of such a series.[1] A key benefit of this approach is that as soon as an item is represented in the system, it can be recommended to other users; the system does not need (or gain anything from) an item to be consumed by other users to be recommendable. This is very important in domains where the number of items is particularly large, or if new items are continuously being added to the system, for instance, for recommendation of online news articles (see, e.g., [27]). On the other hand, a user that has not seen and rated many items will be difficult to help. We call this the *cold-start* problem [36] (or, to signify it is the new user who is short-changed, the *cold-start-user* problem).

Alternatively to content-based filtering, *collaborative filtering* builds on the assumption that a user will enjoy the same items as others who, roughly speaking, enjoys the same things as her. At the centre of collaborative filtering methods is the *rating-matrix*; as sparse matrix containing information about the rating a user u gives an item i. Collaborative filtering methods are often further divided into *memory-based* and *model-based* filtering techniques. The memory-based techniques are based on keeping the rating-matrix in computer memory, and predicting the rating for user u on item j as weighted sum over all ratings given to item j by other users, and where the weights are defined by the *similarity* between u and the other users [25]. While memory-based methods naturally have a benefit from ease of learning and implementation, the major downsides are memory requirements and the computational cost at query-time. Model-based approaches attempt to alleviate these problems by compressing the data in a learned model. Different versions of matrix factorization have historically been most popular, with the actual implementation technique ranging from completely deterministic singular value decomposition [39] to more current probabilistic matrix factorization [3]. These approaches give a latent space representation of users and/or items, are fairly simple to learn, and are very fast at query-time. To facilitate the effect that a user has stronger beliefs in friends and people regarded as experts in the domain, model-based collaborative filtering systems have recently been extended by the notion of trust, enabling the recommendations for user u to be strongest influenced by her similar users, her friends and the domain experts, while putting less weight on other users. This can, for instance, be obtained by examining social networks [8].

There has also been investigations into so-called *hybrid recommendation* systems, where recommendations are based on a unification of collaborative and content-based information. For example, Pennock et al. [33] proposed a personality diagnosis method, which can be seen as combining memory-based and model-based approaches. Wang et al. [43] proposed a method for unifying the user-based and item-based collaborative filtering approaches within a memory-based context, Truyen et al. [41] combined content-based filtering and collaborative filtering in a conditional Markov random field model, and Gu et al. [15] considered methods for integrating content information based on a weighted non-negative matrix

[1]See https://blog.kissmetrics.com/how-netflix-uses-analytics.

factorization [6]. Due to its applicability, in practice most companies end up using some sort of hybrid approach.

25.2.2 Explicit vs. Implicit Feedback

Explicit feedback is present in many of the largest recommender systems today and hence, is extensively researched [1]. The user is commonly asked to rate item i on a Likert scale from 1 to k, ranging from strongly disagree to strongly agree with an item. Its advantages are, among others, the ability to get precise feedback from the user and capturing both positive and negative preferences. However, although having a high popularity, the method has multiple weaknesses. The most prominent weakness is the difficulty of collecting ratings: the method requires the user to spend time rating items and the amount of feedback is often scarce, creating sparse datasets. Further, what a user self-reports often has some disconnect with their real natural behaviour [2], and users might also be pressed to report different preferences due to peer pressure [4]. The fact that we are introducing a user overhead makes it difficult to have a complete view on the user preferences [21].

There are also practical reasons for not having explicit feedback in the form of ratings, dependent on how a website/application interacts and works with its customers. One of the most popular recommender platforms in the world, Netflix, often asks their users to rate items directly after consummation, which does not induce any extra *cost* for the user since it is included in the subscription, while in a web-store rating an item often requires the user to first buy it, consume it, and finally rate it.

Implicit feedback on the other hand utilizes, often already collected, analytics data and hence does not require any extra effort neither for the user nor the front-end-developers. This creates an unobtrusive experience for the user, and can also yield better recommender results since often the event log is less sparse than rating datasets and contains implicit knowledge which, by only using explicit information, would never be exposed in a traditional system. The only downside of this is the huge debate about “customer privacy” and data storage and what companies do with data, who they share it with, etc. However, this can be overcome by being transparent about what the company actually does and use the data for.

25.2.2.1 Differences Between Explicit and Implicit Ratings

It is important to understand that there are several fundamental differences between implicit and explicit ratings, in everything from *meaning* to *evaluation techniques*. Partly inspired by Hu et al. [17], we can identify the following key characteristics and differences between these two types of ratings.

Explicit Ratings

- Contains both positive and negative feedback. In other words, a user is able to explicitly tell the system that he/she does not like an item, as well as the opposite.
- Indicates preference, often on a Likert scale (or similar), where scores range from total dislike to high satisfaction with an item.
- There is medium level of noise in the data; the amount is dependent on the domain in question [2].
- Metrics in use for evaluating recommender systems with explicit ratings are commonly root mean square error (RMSE), mean absolute error (MAE), or other evaluation schemes where we consider how well we match a test-set.
- Both evaluation schemes and recommendation techniques using explicit feedback are heavily researched and used. Many of these are easy to implement, and multiple open-source tools and projects are at researchers and developers disposal [11].

Implicit Ratings

- Usually only contain positive feedback, since we base our models on user activity on an item representing something good. However, looking at metrics such as bounce rates, negative feedback can be produced with some effort.
- Indicates confidence, that is, a recurring event is more likely to reflect the user opinion, but not necessarily his/hers preference.
- There is a high degree of noise; this is one of the main challenges with implicit rating systems. Researchers and developers need to be sure to select the correct set of events and session-metrics to base implicit ratings on.
- As users do not provide numerical scores, a precision-recall evaluation scheme is often preferred to more preference-based metrics.
- Strengths include not requiring extra feedback for the user, having less sparsity and catching actual behaviour in the application.

Notice that several events such as purchasing an item may be considered as explicitly giving positive feedback, but often one sees all interactions treated as implicit feedback. The reasoning behind this lies in the fact that purchasing (in contrast to positively rating) an item happens before actual consumption. Hence the user may not like the item in question, it may be a gift for another person or a variety of other reasons. In the scenario of explicit ratings, the user has already consumed it and is explicitly answering the question *"To which degree did you like it?"*.

25.3 Literature Survey: Recommender Systems for Fashion

There are many challenges when making recommendations in the fashion domain. In this section we briefly discuss some of them.

25.3.1 Recommendation Challenges

(1) Recentness and Seasons Dependency Time is per definition a central feature in fashion, and consequently an important aspect to consider when analysing implicit feedback and making recommendations [30, 34]. There are many types of recentness. For example, an apparel may go *out of fashion*, i.e., experiencing a lowered popularity, or be replaced by a newer collection. Further, users' tastes may change, due to a range of external factors. In a recommender system we need to account for these factors by finding some way of penalizing old items combined with low popularity [30]. In addition, in contrast to other domains, we need to expire feedback given from the user after a fixed time interval as a result of the fact that consumer tastes normally change over time.

(2) Implicit Data The feedback from the user will mainly be implicit (see Sect. 25.2.2), and if the users had the possibility of rating an item there would be no way of knowing *which* features contributed to a positive rating, without collecting a large amount of data. A solution is to assume that an increased interest in an item is correlated with increased interaction with the item. In the fashion domain, users are generally price and brand-aware. Hence, a successful fashion recommender system should utilize this information, while trying to classify users by their preferences based on product features.

(3) Product Semantics Fashion recommender systems may need to aggregate product information or features from a variety of different web stores into a single product database. Such features include colour, type, target group, and style. However, the quality of this information may vary, and sometimes a lot of the data can be missing. This is a common issue in content-based recommendation, where we need to convert our product descriptions and semantics into *structured* data. While such a conversion can be done by various natural language processing techniques, there are other challenges such as ambiguity, sparsity, and language detection to be considered. An example challenge is finding a sensible way to use the product description to define whether a blouse is conservative or flashy, how trendy it is, or whether it is casual or formal [12].

(4) Novelty Some recommender systems produce highly accurate recommendations, with reasonable coverage, but which are ineffective for practical purposes. For instance, a system may recommend a highly popular item to all users that everyone has a pre-existing knowledge of, but some users choose not to access the product due to the fact that it is not novel to them. To address this, a system should have a way to recommend items that both score well, but that the user has never seen before. This property will improve the *novelty* of recommendations. Novelty can also be related to recentness, and is important to support in the fashion domain since we always wish to introduce *new* items and give them inspiration for new styles. Further, *serendipity* is increased when a system recommends items that positively surprise the users, and which the user would not have found given their usual shopping patterns. Overall, we would like to not only recommend based on brands and clothes the users frequently buy, but enhance their experience by including nontraditional choices as well.

(5) Dealing with Fluctuating Prices Price is an important factor when considering both fashion marketing and user behaviour. Users usually browse through multiple sites in order to find the best offer, and research is an important part of the shopping experience. Ideally, fixed low prices would help focusing on providing good recommendations based on other user-related factors such as preferences and product information. However, due to, e.g., competitions and product promotions, prices in the fashion domain often fluctuate. As a result, price is an important piece of information in a recommender system needs to provide. A way to address this is the presentation of “good deals” to the user and informing him/her about upcoming sales on items which they might find interesting. It can also help retailers target their sales campaigns towards items with high activity, but low conversion.

(6) Demographics, Subcultures, and Social-Classes Demographics and the social aspects within fashion are very important and can be utilized in a recommender system by looking at user properties such as location, friends, occupation, gender, and age. In addition, we could use social network information, such as Facebook likes, which implicitly expose the interests of the user. However, a major challenge recommender systems often faces is that not all users are willing to provide personal information due to privacy concerns. Therefore, a fashion recommender system has to provide a good method to compensate the possible lack of necessary demographic information, so that good recommendations can still be achieved.

25.3.2 Methods for Fashion Recommendations

In this subsection we present different methods that existing fashion-related systems have used to recommend products to their user. This will provide the reader with an overview of existing research approaches that have focused on fashion recommender systems.

25.3.2.1 Visual-Focused Approach

Fashion and the products to which it relates are highly dependent on visual characteristics. Like most products, without its visual appearance, a fashion product would not be very interesting to a customer [10]. To deal with this, Iwata et al. [18] suggest an approach that recommends fashion by assessing the importance of a product based on its images. Specifically, they propose a so-called fashion coordinates recommender system using photographs from fashion magazines. Their main idea is to build a topic model-based system that helps a user to choose the best coordinates for clothes. They train their system using full body images from fashion magazines. From this the authors modelled clothes using a multilingual topic model. They segment clothing images into regions of tops and bottoms, and use their visual features to train their system to decide which tops would match a give bottom and vice versa. This means that the system can recommend the tops (or bottoms) which fit the input bottoms (or tops) for the user. Their experiments showed that the proposed method can provide an accuracy that is higher than both a more naive approach and a random selection.

Jagadeesh et al. [19] propose a completely automated large-scale visual recommendation system for fashion. To achieve this, they efficiently exploit the availability of large quantities of online fashion images and their corresponding meta-data. Their approach is divided into four interrelated main stages. First, they build a collection of fashion images by crawling fashion photographs from the web. Second, they do a preliminary data analysis to find out how people combine their outfit colours. Third, they build a model to allow their system to predict or forecast what might be interesting to the users based on their past history. As part of this the authors propose two different approaches, a deterministic one called deterministic fashion recommender (DFR) and stochastic method called stochastic fashion recommender (SFR). These methods constitute the core of this fashion recommender system. With DFR, the main idea is to exploit the visual features to recommend which cloth pieces should a user wear together (e.g., a top and bottom), using deterministic learning methods. With SFR, on the other hand, the idea is to explore users' biases or subjectivity when recommending the clothes. This is done by applying statistical based methods such as using Gaussian mixture models. Fourth and finally they carry out statistical tests with independent users to validate the approaches. These tests help them to decide which methods provide the best recommendation. Overall, this method has great potentials in designing good recommender systems for fashion and that it indeed has industrial applicability in the context of mobile fashion shopping.

Miura et al. [29] propose a system called SNAPPER. SNAPPER can be viewed more as a retrieval system than recommender systems, per se. The authors call it a fashion coordinate image retrieval system. Nevertheless, we chose to present this system here to show the diversity of the approaches related to exploiting visual features in the fashion domain. The main idea is to have a system where the user can search for matching clothing parts such as tops, inner tops, and shoes based on a given part (e.g., a bottom). They suggest a system consisting of three main parts. The first part detects four different clothing areas, including top outer, top inner, bottoms, and shoes, from all fashion coordinate images in the database. The second part takes care of detecting and describing the image features from each area in each image. In the third part, the system retrieves a fashion coordinate image by comparing the features of a search object area with those of the query clothing image. Although being focused as a retrieval method, SNAPPER can still be used as a step towards recommending fashion.

25.3.2.2 Scenario and Goal-Oriented Recommendation

Shen et al. [37] propose a recommender system which produces recommendations based not only on the metadata of the products, but also on user-written input. Knowledge used to handle the user input is derived from so-called open mind common sense (OMCS). OMCS is an artificial intelligence project at the Massachusetts Institute of Technology (MIT) Media Lab, with a goal to build and utilize a large commonsense knowledge base from the contributions of many thousands of people across the Web [38]. The user uploads his or her clothes and adds brand (e.g., Nike), type (e.g., jeans), material, and a description about the item (e.g., "I put these on when I get home"). Then, the system makes recommendations based on the scenario in which the user needs help to find suiting clothes to wear. The typical use case of the system is when a user is unsure about what to wear under different circumstances but knows something about the scenario or occasion the clothes will be worn in, e.g., "I am going to the beach". The system uses information about similar users to recommend relevant outfits. The different describing fields about the items are given a six-tuple style value. The six tuples are: luxurious, formal, funky, elegant, trendy, and sporty, where each is given a value from 0 to 10 based on how much the describing field of an item matches the current tuple. The different describing fields are given a default value, which can be changed by the user whenever necessary. This is an interesting approach to fashion and clothes recommendations. However, the necessity of user scenario input and full description of the six-tuple for the different describing fields may make this approach unfeasible to handle a fast-increasing number of users and items. Hence, the approach is likely not very scalable.

Lin et al. [26] and Ying Zhao [45] propose two other similar systems, respectively. Both built fashion recommender systems that help the user decide what to wear in different situations. For example, the system by Ying Zhao [45] recommends two sets of top-to-toe clothing based on the current season, the occasion and the items the user has uploaded. To achieve this, the uploaded items must contain a set of descriptions, including the occasion to use the item. In addition, a user must add a user profile describing their interests and provide information about the different items in their wardrobe. Then, to recommend suitable outfits, the system matches the user profile with the available description of items.

Similar to the above approaches, Kobayashi et al. [22] present an approach called goal-oriented recommendation. They developed a system where the main goal is to help a user in a situation where he/she is looking for clothes for specific event but has no idea about what to wear. As above, the idea is to allow a system recommend cloth combinations based on a given occasion. This means that the user needs only to specify the occasion (the goal), e.g., "I am going to the entrance ceremony", and the system suggests what clothing items, such as jackets, shirts, and pants to wear. As this suggests, an important part of this approach is to understand the user-specified occasions and classify these correctly. It then relies on the user to choose the desired outfits by presenting them visually.

25.3.2.3 Fashion Recommendation Based on Photograph Integrated with Occasion

Liu et al. [28] combined the approaches by Iwata et al. [18], Ying Zhao [45], and Shen et al. [37]. They proposed an approach that recommends clothes based on both clothing photographs and the occasion in which they are to be worn. They do this by incorporating fashion rules like what can you wear to which occasion and what can you wear as a complete set to different occasions. The recommender learns the clothing recommendations through a latent support vector machine framework. They use this framework to match four potential functions: visual features vs. attribute, visual features vs. occasion, attributes vs. occasion, and attribute vs. attribute. Then they use these functions together in a combined scoring function for clothing recommendation. According to the results in [28], this method preformed better than the baselines, but was highly dependent on human intervention for higher detection accuracy. The main reason for this is that the learning process was done only from fashion photographs.

25.3.3 Systems and Applications for Fashion Recommendation

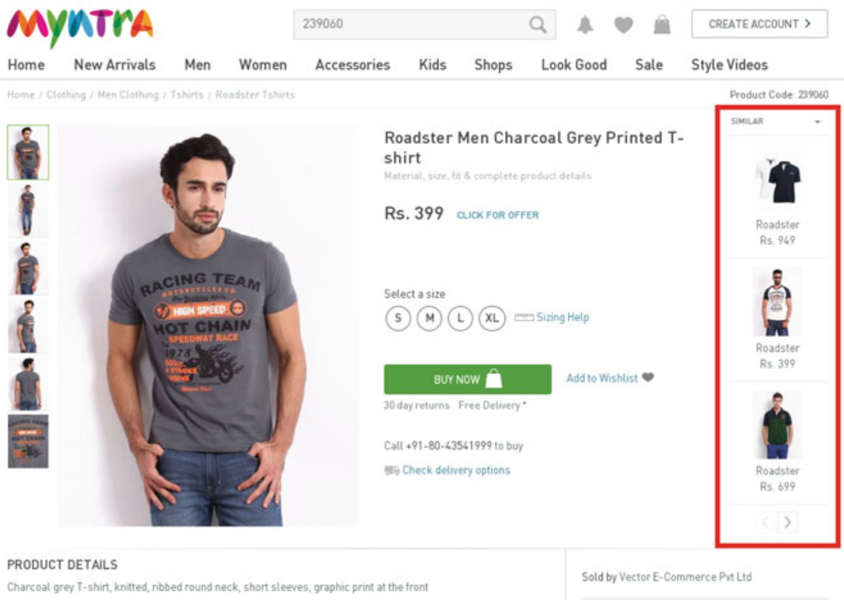
There are many different applications of fashion recommendation systems. To show how they are implemented and used in practice, in this section we discuss briefly a few online fashion stores that implement different approaches to enhance the user experience by using of recommender systems. Here, we have chosen a set that covers a breadth of features of recommendation methods.

25.3.3.1 Myntra

Myntra[2] is one of the India's largest e-commerce stores for fashion and aims at providing a hassle free shopping experience for the user. Their main goal is to bring the newest and most in-season fashion products available to the user on the web, using an online store. The brand base of Myntra consists of 500 leading brands from both inside and outside India.

The web page uses a set of recommendation approaches to inform the user of what they might like, and to increase the user's awareness of different kinds of items, such as similar items and most popular items. Figure 25.1 shows the user interface of the Myntra web-based system. In this figure, we can see in the red box how Myntra suggests items which are similar to the item the user is currently looking at.

[2]See http://www.myntra.com.



Charcoal grey T-shirt, knitted, ribbed round neck, short sleeves, graphic print at the front

Fig. 25.1 Example of Myntra's "similar item" approach

Myntra has several main advantages. First, it is both a web-based online store and a native smartphone app, with in-app purchase possibilities. Second, it provides a popular list for both brands and stores, allowing a user to know, e.g., which brands and stores that are currently popular. Third, as mentioned above, Myntra provides a list of items that are similar to the item a user is currently viewing. Fourth, it provides the user with the ability to add items in a "wish list", which could, in turn, potentially be used to improve the recommendation to users.

Despite these advantages, the main weakness of Myntra is that the recommendations are not personalized. This makes it hard to customize suggestions for, e.g., outfits to specific users. Moreover, a user cannot follow other users, which also restricts the ability for a user to be "inspired" by what others have bought or they are interested in. In conclusion, Myntra does not incorporate any social aspects. Hence, collaborative filtering-based approaches would not work very well.

25.3.3.2 ASOS

ASOS[3] is a global online fashion and beauty retailer selling over 65,000 branded and own-label products. According to their website, Asos has more than 29 million unique visitors per month and ship to more than 234 countries and territories.

[3]See http://www.asos.com.

Similar to Myntra, ASOS is a web-based online store, providing the user the ability to add products directly to a shopping chart. When accessing an item, Asos presents the user with a set of items which it recommends for the user. It also presents a set of clothes which might *complete the look*. These recommendations are products that Asos assumes to *go well* with the currently viewed or purchased item. The users can also like and save items for later in a form of a “wish list”. Other ASOS features are *most popular* and *outfits & looks*. *Outfits & looks* is a community where the user is given a set of options, including voting for items, shop looks, gather *style credits*, and follow other users. All of these extra features are a good source for ASOS to collect more information and data on consumer preferences.

Similar to Myntra, Asos has several main advantages. First, it is not only a web-based online store but also is provided as a smartphone app, with in app purchase possibilities. Next, it provides a popular list for both brands and stores, a list of items that are similar to the item a user i currently viewing, and a wish list. In contrast to Myntra, however, ASOS has a feature that allows a user to follow other users, making him/her aware of what others also like. Finally, Asos has the ability to suggest items that might *go well* with other items and that the recommendations can be tailored to each specific user.

25.3.3.3 Zoolando

Zoolando[4] is an European online fashion retailer, providing the user the ability to buy fashion products on the web via a smartphone app. According to their website Zoolando ships more than 1500 fashion brands to customers in 15 European countries. One of its most important features seems to exploit the fact that fashion shoppers love to see the looks of outfits they might be interested to buy. If “the look” is not available, then it offers the users some similar alternative. Further, Zoolando allows the users to see accessories and other articles that complement and complete “a look” by using a feature they call “Shop the Look” (see Fig. 25.2). This feature also allows them to try alternative looks with a single article.

Focusing on the recommender system aspects, Zoolando has the advantages of both Myntra and Asos discussed above. In addition, it has the aforementioned “Shop the Look” feature, which allows the user to customize a set of products to his/her taste.

25.3.3.4 Mallzee

In contrast to all the above-mentioned fashion retailers, Mallzee[5] is a smartphone app and not an online store. So instead of shopping online the users can use their mobile phone to access fashion outfits.

[4]See https://www.zalando.co.uk.

[5]See http://www.mallzee.com.

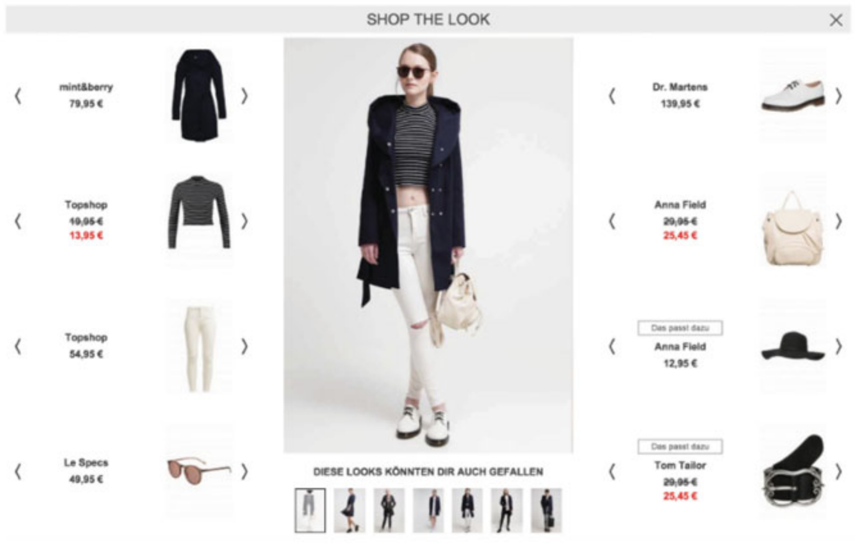


Fig. 25.2 Zoolando "Shop the Look" interface

Related to recommender systems, Mallzee has features that can help providing good recommendation to the users. It uses a *hot-or-not* approach, where the user can like or dislike products. This feedback is then used to recommend other products users might like. In addition, a user can get feedback from other users.

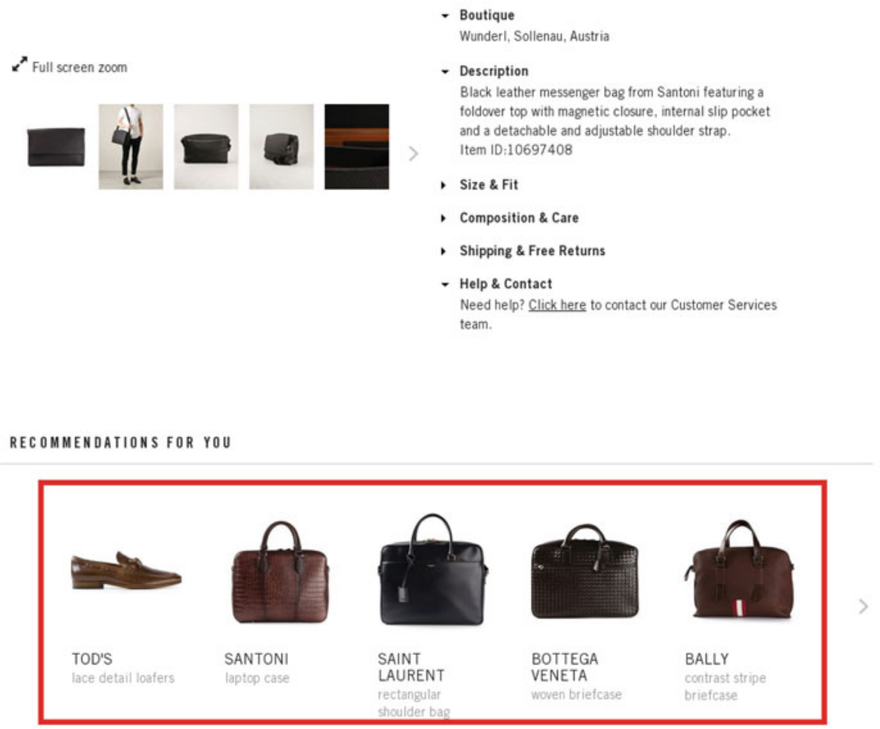
As this infers, apart from lacking a list of popular items and a list of similar items, Mallzee has all the advantages of all the applications discussed above. This comes in addition to the hot-or-not feature. The only weakness of Mallzee is that it does not provide any direct purchase, making it impossible to track what items users liked and actually bought. Hence, not all recommendations can be guaranteed to be accurate.

25.3.3.5 Farfetch

Farfetch[6] is a collection of over 1000 boutiques from all over the world gathered on one web page. Users can shop directly on the page, and get the item delivered to their doorstep with only one checkout process.

When browsing an item the user is presented with a set of recommendations related to the current item, and previous browsing history. The item can be added to a "want list", as well as to the shopping chart.

[6]See http://www.farfetch.com.



Wunderl, Sollenau, Austria

Black leather messenger bag from Santoni featuring a foldover top with magnetic closure, internal slip pocket and a detachable and adjustable shoulder strap.
Item ID:10697408

Need help? Click here to contact our Customer Services team.

Fig. 25.3 Example of Farfetch's recommendations

Figure 25.3 shows a part of the user interface of the Farfetch system. In this figure, the thick line box shows how Farfetch recommends items that might be of interest to the user. As we can observe in this example, the first item is a shoe, which seemed to the last item accessed by the user and that the next four items are related to the currently viewed item.

Farfetch is similar to both Myntra and Asos in that it allows the user to view and buy items directly from their website. It has most of the advantages of both these systems including a list of most popular items, a list of similar items, and wish list. In addition, Farfetch has a list of items tailored to a user that it recommends to see and possibly buy. The only weakness is that it does not allow a user to follow another user.

25.3.3.6 Summary

As can be inferred from our discussion in this section, there is a wide variety of online stores and smartphone apps that allow users access fashion at their finger tips. Focusing on recommender systems, it has not been easy to do a deep scientific analysis of how recommendation is done. Nevertheless, it is likely that many of these websites base their recommendations on data gathered from products that have previously been bought or viewed together. Although this is a fairly easy way of providing recommendations, it is often not personalized enough and is based on averages.

25.4 Case Study: SoBazaar

We have developed a fashion portal that allows us to study users' shopping behaviour [30]. This is done by analysing their interaction with the portal through a smartphone app called SoBazaar.[7] In this section, we will present our work developing recommendation approach for this application as a case study.

The SoBazaar app provides feeds and a built-in search functionality that will lead the user to storefronts, brands, or specific items. Then, users can indicate interest in products by looking at them (Clicks), marking them as "loved"(Wants), or as "intended for purchase" (Purchases). Note that since SoBazaar does not have an "in-app" purchase possibility, there is no way to track the user is in fact buying the product he/she has indicated to purchase. For this reason, in this section, "purchase" refers to "intended for purchase". Figure 25.4 shows a part of the user interface for the smartphone app.

Our main goal with developing SoBazaar recommender has been to make fashion recommendation more robust and more personalized than the existing ones. In Sect. 25.4.1, we first give a brief overview of the SoBazaar approach. This includes discussing how we address the challenges from Sect. 25.3.1. Then, in Sect. 25.4.2 we present a preliminary results.

25.4.1 Overview of the SoBazaar Approach

Figure 25.5 gives an overview of how we implemented the recommendation support in SoBazaar. As can be seen from the figure, our proposed solution consists of interconnected processes. First, a conversion from implicit feedback to ratings is made (see Sect. 25.4.1.1). Second, we combine product features extracted from a product database together with the generated implicit ratings for *cold-start boosting*, which by simulating stereotypical users is based on probabilities that we are able to remove some sparsity issues. Third, we use the ratings in a variety of different approaches for doing the actual recommendations. We investigate the

[7] SoBazaar has been acquired by another company and is currently called Villoid (see http://www.villoid.com).

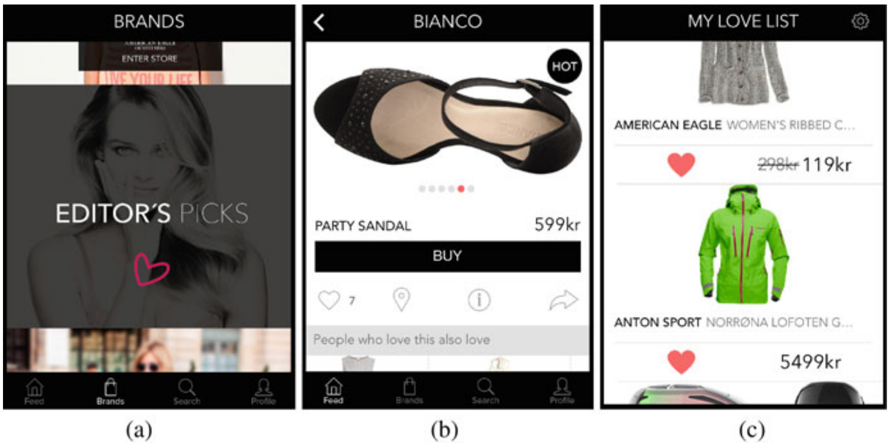


Fig. 25.4 Screenshots from the application. From the left to right: (**a**) editorial information, (**b**) product details, and (**c**) a "love list"

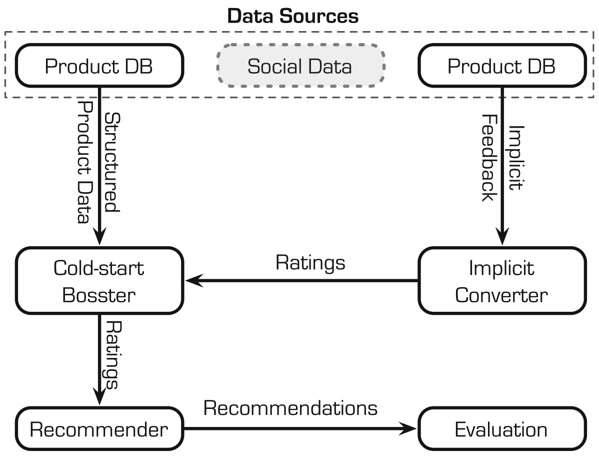


Fig. 25.5 Overview of how the recommendation approach in the SoBazaar System was implemented

usage of traditional and simple approaches such as looking at popularity and averages. We compare these with others approaches that use modern matrix-factorization techniques. More specifically, as discussed in Sect. 25.4.2.1, we study the effect of applying the *k*-nearest neighbour item-based collaborative filtering (kNN-item) and Bayesian personalized ranking (BPR) [35]. Fourth, we evaluate the recommendations with respect to the relevant metrics for both domain and application (see Sect. 25.4.2.2).

25.4.1.1 From Implicit Feedback to Ratings

The first step towards building a recommender system is to translate data capturing users' actions into a number that can be understood as the "score" the users would give to particular items. The most important factor when creating such numbers is to understand the data available and their implications for user preferences. Once the data is analysed, suitable generalizations can be chosen.

Often, counting events can be useful; in our case a natural assumption is that a high number of clicks correlates with a preference or a liking for that item. Also, scores are dynamic in the sense that an item viewed a long time ago will probably be less relevant today than if it had been clicked yesterday. Therefore, the time when a user last viewed items and in which order should affect the score calculation.

Note that when scores are not explicitly given by users, computed implicit scores become a ground truth equivalent. The underlying assumption is that the generated implicit scores represent users' preferences as well. As discussed previously, however, the main challenge for recommender systems is that the inputs for users are sparse and often noisy. Nevertheless, we can address this challenge by having optimal methods for choosing relevant features and score generation.

We use the work by Ghosh [13] as a starting point. Similar to this work, we also compute the ratings based on users' interaction with the system, which we call events. However, while [13] seems to define the scores for each event manually, we compute the weights of different events based on the statistical properties of the dataset. Moreover, we create scores on a continuous scale between a minimum and a maximum value.

We focus on three events, *Clicks*, *Wants*, and *Purchases*. Notice that the events are naturally ordered, i.e., wanting an item is a stronger indicator than clicking it, and intending purchase is stronger than wanting the item. We decided to let a user's interaction with an item be defined through the strongest event. For example, we disregard all click events when an item is wanted. Among the three events, 61% of the registrations in our dataset are of the type *Clicks*, 35% are *Wants*, and the remaining 4% are *Purchases*. This distribution is used to generate the ranges of scores in Table 25.1. Note that the intervals for each event ensure natural ordering, and the split at, e.g., 96 is due to 96% of the events in our dataset being *Wants* or weaker.

Table 25.1 Score mapping

Event type	*Clicks*	*Wants*	*Purchases*
Score range	[0, 61]	[61, 96]	[96, 100]

Important properties in the fashion domain that must be captured by the system include seasons and trends, price sensitivity and popularity. Based on this, we adjust the relevance of a user's action wrt. an action's *recentness* as well as an item's *price* and overall *popularity*. We will do so using a *penalization function*, giving each event a specific value inside the range of possible scores.

In general, when a user u triggers an event e, e.g., *Clicks*, we have a range of possible scores to give this event. We use S_e to denote this score, and let m_e and M_e denote the minimum and maximum score possible for event e, respectively. We then use a penalization function $p_u(x)$ taking a feature value x (e.g., related to an item's price), and return a number between zero and one to adjust the score inside the possible range. Specifically, we use the following equation to compute S_e:

$$S_e = M_e - (M_e - m_e) \cdot p_u(x). \tag{25.1}$$

This formalizes a procedure for finding S_e, the score given to event e after penalization, which was originally assumed to be in the interval $[m_e, M_e]$. Note that penalization also implicitly adds negative feedback. This is an important aspect to keep in mind when working with implicit feedback, as modern recommender engines work better when based on both positive and negative feedback.

25.4.1.2 Considering Recentness

As mentioned, fashion is about timing and following recent trends. Thus, *recentness* is a natural feature to determine how well an item is liked. In our system we penalize items the user has not considered recently. To do this, we look at the number of days since the user did the event in question (denoted by x), and compare this to the oldest event this user has in the database, denoted by F_u.

We then enforce a *linear penalization*, letting $p_u(x) = x/F_u$. Thus, an event happening today ($x = 0$) will be penalized with $p_u(0) = 0$, while the oldest event in the database will be penalized with $p_u(0) = 1$. Some examples of the use of this penalization (assuming $F_u = 14$) are given in Table 25.2.

Table 25.2 Example of penalization. x is the number of days since the user did the event, $p_u(x)$ denotes the linear penalization function for a given x, and S_e is the resulting rating score

Event type	x	$p_u(x)$	S_e
Purchases	3	0.21	99.2
Wants	7	0.50	78.5
Clicks	0	0.00	62.0
Clicks	14	1.00	0.00

A linear penalty function does not fit well with the idea of seasons. As an example, a store may be selling warm clothes to customers in the northern hemisphere from November to March, but wants to focus its recommendations at summer-clothes when the season changes. To mimic this behaviour, we introduce a sigmoid function, and we chose to work with a parametrization where we can fine tune both the steepness (s) and shift (c) of the "S"-shape to fit well with the data. This results in a new penalization function $p_u(x)$, given by:

$$p_u(x) = \frac{1}{1 + \exp\left(-s \cdot x/F_u + c\right)}. \tag{25.2}$$

Considering recentness in this way could obscure the preferences of users who have been absent from the app for some time, because all these users' events would be heavily penalized. In order to mitigate this problem, we adjust our features to consider the *ordering* of events instead of timing. Assume the user has triggered N events in total. Now, let x be the number of events triggered after the event e. In this case, it is appropriate to use a linear penalization, with $p_u(x) = x/(N - 1)$. This definition ensures that the difference in penalization between the two most recent items is equal to the difference between the two oldest items. In total, this gives us two different ways of incorporating recency into the score of an event e. We discuss how these are combined in Sect. 25.4.1.5.

25.4.1.3 Considering Price

Users differ in the price range of items they frequent, and as people tend to be price sensitive, the prices of the items should also come into the score calculation. Using the user's "typical" price-range, we can create personalized scores, penalizing items that are not in the price range preferred by the user. This procedure is done in two steps: first we find the average price a_u of all items related to user u. Second, we calculate the difference between the price of item i triggering event e (denoted by π_i) and a_u. We define the penalization function as:

$$p_u(\pi_i) = \begin{cases} \min\left(1, \frac{\pi_i - a_u}{F}\right) & \text{if } \pi_i \geq a_u \\ \min\left(1, \frac{a_u - \pi_i}{2F}\right) & \text{if } \pi_i < a_u, \end{cases}$$

where F is a constant controlling the price elasticity of the user-group. Note that we make the assumption that items being cheaper than the user's average (i.e., $\pi_i < a_u$) should also be penalized, but less strictly, making cheaper items half as sensitive to penalization, compared to more expensive items ($\pi_i \geq a_u$). Note that the above scoring mechanism only works if we already know the user's previous interaction history. For new users $p_u(\pi_i) = 0$.

25.4.1.4 Considering Popularity

As fashion is strongly related to the (global) popularity of items, we will consider popularity as a feature. By comparing a user's behaviour to the rest of the population, we can tell if the user's activities conforms to common standards or if her taste is more unique, giving significant clues about items to recommend.

The goal is to classify to what degree a user likes popular items, then define a penalty function that reduces the score of items that are too popular (if the user is not under peer-pressure) or too obscure (if the user prefers popular items). Assume we have ordered all items wrt. decreasing popularity. We define t_i, the popularity of item i, as the fraction of items that are more popular than item i.

Thus, t_i is zero if item i is the most popular, one if it is the least popular. The average popularity of the items user u looks at (denoted a_u) can now be calculated by the average of the relevant t_i values. We build the penalty function using two linear functions: one when the popularity of item i is below a_u and the one for when the popularity is equal or higher. We also introduce a constant c, being the penalization given to the most popular item overall (when $t_i = 0$):

$$p_u(t_i) = \begin{cases} c \cdot (a_u - t_i) / a_u & \text{if } t_i < a_e \\ (t_i - a_u) / (1 - a_u) & \text{if } t_i \geq a_u. \end{cases}$$

25.4.1.5 Combining the Different Penalizations

There are four penalization functions, each capturing a specific aspect of the domain and defining sets of implicit scores: *price*, *popularity*, and two different uses of *recentness*. To infer a user's implicit score for an item, we *blend* these features.

When linearly combining M models $m_1 \ldots m_M$, we choose M weights ω_j, where each weight represents the importance we give the corresponding model in the final blend. For a given combination of a user and an item, each model m_j proposes calculates a score S_j, and the blended results is a sum over all models:

$$S_e = \sum_{j=1}^{M} \omega_j \cdot S_j.$$

This requires setting weights for M different factors. Optimally we would like to run one blend, compare the result to a gold standard, and adjust the weights accordingly. This is done in simple blending schemes using basic approaches like linear regression, or in advanced schemes, like binned linear regression, bagged gradient boosted decision trees, and kernel Ridge regression blending [20].

However, as discussed, we lack a ground truth for making implicit scores, and cannot use supervised learning techniques to optimize the weights. In fact, we make several assumptions on how implicit features contribute to higher preference and thus higher scores, yet we lack the means to confirm these assumptions. Then for M different models we assume that they all contribute an equal amount to the final results, thus having the weights $\omega_j = 1/M$.

25.4.2 Preliminary Results

In this section we describe how we set up our experiments. First, we present the dataset we used in the experiments. Then, we describe the evaluation metrics used in our evaluation. Finally, we give an overview of the implemented methods that we base our experiments on.

25.4.2.1 Experimental Setup

We describe how the dataset was obtained. Later we explain how our approach was evaluated by the following metrics: *Area Under the Curve* (AUC), *Recall* (R), and Mean Average Precision (MAP).

Generating the Dataset

To generate our dataset we use data that we collected from users performing actions in SoBazaar, i.e., triggered events. Such actions range from accessing a storefront or scrolling the page to purchasing a product. We collected the data while SoBazaar was still in a beta-testing stage, and not yet officially released as final product. Table 25.3 gives an overview of the SoBazaar data.

In summary, the average amount of *Purchases* per user is below one. Hence, using purchases alone would often imply incomplete recommendations. *Clicks* and *Wants*, on the other hand, have a much higher occurrence. Together, these three events result in more than 20 observations per user. Although, the data still is sparse, it gives a richer description of user behaviour than the purchases alone.

Evaluation Metrics

The Area Under the Curve (AUC) is part of performance metric of a logistic regression and is a commonly used to evaluate binary classification problems such as predicting a "Buy" or "Sell" decision. We can interpret it as follows. For a given random positive observation and negative observation, the AUC gives the proportion of the time of the correct guesses. Hence, a perfect model would have an AUC score of 1, while random guessing would score 0.5. The fact that we are working with a highly sparse dataset with a large item collection, the likelihood of finding a hidden item in the top-20 recommended items is not very high. Thus, AUC alone would not be enough to evaluate our methods. Recall is computed as the proportion of the number of interaction events (i.e., *Clicks*, *Wants*, and *Purchases*) retrieved in the top 20 prediction lists over the total number of interaction events in the test set. Mean Average Precision (MAP) is a measure borrowed from the information retrieval field to compute the mean of the average precision across different item retrieval. Here,

Table 25.3 Overview of the SoBazaar dataset

Attribute	Count
Total number of product events	35,324
Unique user ids	1532
Unique item ids	5688
Unique storefronts	144
Unique retailer brands	22
Item clicks	21,400
Item wants	12,436
Item purchases	1488
First event	Mon, 07 October 2013 10:59:57 GMT
Last event	Mon, 19 May 2014 22:51:19 GMT
Lifetime of data	224 days
Average item click count per user	13.9686
Average item want count per user	8.1174
Average item purchase count per user	0.9712
Average user interaction count per item	6.2102
Average item interaction count per user	23.0574
Median item interaction count per user	7.0

we specifically consider the top 20 items, and count how many of these items are recommended correctly to the user. In other words, MAP will give us a score of how many items we are able to retrieve and how high up in the recommendation lists they are placed.

Implemented Recommendation Algorithms and Settings

For the experiment purpose, we selected different recommendation algorithms, including both binary and non-binary approaches. These algorithms are then compared with a non-personalized baseline, which recommends the most popular items overall without personalization. As an input to the algorithms, we used the *most-popular* items.

For *binary recommenders*, we only use binary data of users' interactions. We apply two algorithms, namely the k-nearest neighbour item-based collaborative filtering (kNN-item) and Bayesian personalized ranking (BPR) [35]. We used the kNN-item implementation in Mahout.[8] The BPR algorithm, which learns the latent factors of a matrix factorization using binary implicit feedback, has been implemented in MyMediaLite [11]. BPR is a framework that can be used to optimize different types of models based on training data containing implicit feedback. For this reason it fits very well with our approach. BPR's main idea is to focus

[8]See http://mahout.apache.org.

on pairs of entities rather than single entities in computing its loss functions. In this way, it allows the interpretation of positive-only data as partial ranking data, and to learn the model parameters using a generic algorithm based on stochastic gradient descent [7]. *Non-binary recommenders*, on the other hand, base their recommendations on the inferred implicit scores. Here, we use the alternating least-squares with weighted λ-regularization (ALS-WR) [40], which was initially developed for the Netflix competition [5]. For our experiment we have used the Mahout implementation at the URL given before. Alternating-least-squares (ALS) is a method to solve the optimization problem to minimize the regularized square error function, as part of matrix factorization approach. ALS deals with the fact that this function is not convex. The idea is to alternate between fixing each of the two unknown latent matrices (representing users and items, respectively) at a time, which results in a quadratic optimization problem for the other matrix that now can be solved directly. This, in turns, ensures that each step decreases the error until convergence. There are particularly two features that make ALS favourable over a simpler and faster stochastic gradient descent: First, ALS can be parallelized since the system computes each factor independently of other item factors. The same can also be applied to the user factors. Second, since we are concerned with implicit data, although the test set is likely to be sparse, the training set can be rather complete. Therefore looping over each single training case as with a gradient descent method would not be practical. Additionally, we included the k-nearest neighbour user-based collaborative filtering with cosine similarity measure (kNN-user).

25.4.2.2 Results

In the first experiment we divided the dataset randomly into training and test sets, without looking at the ordering of the events. We compare the results with a time-based split.

Random Split

For this approach, we chose to only factor in popularity and price, and did not use the recentness feature. Table 25.4 shows the results from this.

The non-binary recommenders outperformed the binary methods as well as the baseline ("Most-Popular") approach with respect to all metrics. In particular, the ALS-WR with price provided the best results. This is somewhat surprising, as algorithms optimized by minimizing the root mean square error typically do not perform well when asked to give top-k recommendations [9]. Nevertheless, this supports the intuition that users are generally price-aware.

Table 25.4 Random split results 90:10—the results are averaged over 10 runs

Model	AUC	MAP	MAP_c	MAP_w	MAP_p	R_c	R_w	R_p
Most-popular	0.7515	0.013	0.015	0.011	0.010	0.048	0.033	0.027
kNN-item, $k = 120$	0.7358	0.007	0.004	0.010	0.006	0.029	0.043	0.049
BPR	0.7286	0.010	0.012	0.006	0.013	0.038	0.025	0.039
ALS-WR-popular	**0.8357**	0.021	0.020	0.019	0.013	0.060	0.068	0.068
ALS-WR-price	0.8251	**0.036**	**0.031**	**0.040**	0.025	**0.083**	**0.096**	0.084
ALS-WR-blend 1	0.8310	0.031	0.027	0.031	0.019	0.075	0.089	**0.097**
kNN-user, $k = 200$	0.7963	0.033	0.027	0.039	**0.028**	0.076	0.087	0.091

Blend combines price and popularity. Subscripts "C", "W", and "P" refer to results for *Clicks*, *Wants*, and *Purchases*, respectively. The MAP-results report MAP20. The best score in each column is type-set boldface

Time-Based Split

By splitting the dataset wrt. time, we make the evaluation more realistic and also get the opportunity to factor in and measure the effect of recentness. We used the first 80% of the events in chronological order for training the models, making predictions for the last 20%. The test-set was slightly skewed compared to the training set, as only 0.07% of the last 20% of the data were purchases (4% overall). The results are shown in Table 25.5.

The low AUC scores can be attributed to the fact that 143 new users and 870 previously unrated items were introduced during the test period. To our surprise, none of the binary recommenders outperformed the baseline in terms of AUC. The ALS-WR systems performed best overall, with the exception of the metrics related to *purchases* only (MAP_p, R_p), where user-based kNN gave the superior results. Among the ALS-WR models, those that factored in recentness gave the best performance. This is an interesting finding which may be relevant to most recommender systems, but especially to fashion. Our finding has confirmed our expectation of this being an important variable.

25.5 Concluding Remarks

In this chapter we have looked at how recommender systems are used in online fashion stores to enhance the user experience and increase sales. Fashion recommender systems are faced with many of the same challenges that are common to most recommender systems, but we also discussed some issues that are specific to this domain.

We exemplified solution strategies by considering the SoBazaar system.[9] In this analysis we gave particular focus to the fact that we do not have access to explicit

[9]SoBazaar has been acquired by VILLOID after our study was completed. The current version of the system can be found at http://villoid.com.

Table 25.5 Time-based split results

Model	AUC	MAP	MAP_c	MAP_w	MAP_p	R_c	R_w	R_p
Most popular	0.6132	0.0049	0.0046	0.003	0.000	0.009	0.004	0.000
kNN-item, $k = 100$	0.4643	0.0026	0.0026	0.001	0.003	0.005	0.003	0.028
BPR	0.6017	0.0049	0.0046	0.003	0.005	0.007	0.004	0.022
ALS-WR-popular	0.6382	0.0015	0.0004	0.006	0.001	0.003	0.005	0.012
ALS-WR-price	0.6487	0.0044	0.0029	0.009	0.022	0.006	**0.007**	0.082
ALS-WR-recentness	**0.6648**	0.0060	0.0037	**0.018**	0.048	**0.008**	**0.007**	0.077
ALS-WR-blend 1	0.6510	0.0050	0.0038	0.007	0.027	**0.008**	0.006	0.056
ALS-WR-blend 2	0.6545	**0.0064**	**0.0048**	0.012	0.017	0.007	**0.007**	0.056
kNN-user, $k = 200$	0.5854	0.0044	0.0029	0.008	**0.053**	0.006	0.006	**0.083**

All scores are averaged over 10 runs. Blend 1 combines price and popularity, blend 2 combines price, popularity, and recentness. Subscripts "C", "W", and "P" refer to results for *Clicks*, *Wants*, and *Purchases*, respectively. The MAP-results report MAP20. The best score in each column is type-set boldface

ratings that are commonly used in more traditional domains. Rather, we analysed users' behaviour to infer their implicit preference scores. These scores serve as "ratings" that can be utilized by standard recommendation algorithms. While the experimental results demonstrate the effectiveness and viability of our method, there are still a number of open questions worth further exploration. For example, using implicit rating and stock images, how can one recommend *outfits*, including shoes, dress, and accessories? And how can social graphs (defined both inside and outside the fashion store) best be leveraged? While the number of competing stores is steadily increasing, the market potential is huge. Therefore, recommender systems for fashion is an interesting field of science, seen both from an academic and an economic point of view.

Acknowledgements We would like to thank the SoBazaar team at Telenor Digital for providing the SoBazaar dataset and thus making our experiments possible. The dataset is available at https://github.com/hainguyen-telenor/Learning-to-rank-from-implicit-feedback under the Open Database (ODbL) v1.0 Licence.[10]

References

1. Gediminas Adomavicius and Alexander Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Transaction on Knowledge and Data Engineering (TKDE)*, 17(6):734–749, June 2005. ISSN 1041–4347.
2. Xavier Amatriain, Josep M. Pujol, and Nuria Oliver. I like it… I like it not: Evaluating user ratings noise in recommender systems. In *User Modeling, Adaptation, and Personalization, 17th International Conference, UMAP 2009, formerly UM and AH, Trento, Italy, June 22–26, 2009. Proceedings*, pages 247–258, 2009. URL https://doi.org/10.1007/978-3-642-02247-0_24.

[10]See http://opendatacommons.org/licenses/odbl/1.0/.

3. Josef Bauer and Alexandros Nanopoulos. A framework for matrix factorization based on general distributions. In *Eighth ACM Conference on Recommender Systems, RecSys '14, Foster City, Silicon Valley, CA, USA - October 06 – 10, 2014*, pages 249–256, 2014. URL https://doi.acm.org/10.1145/2645710.2645735.
4. Robert M. Bell and Yehuda Koren. Scalable collaborative filtering with jointly derived neighborhood interpolation weights. In *Proceedings of the 7th IEEE International Conference on Data Mining (ICDM 2007), October 28–31, 2007, Omaha, Nebraska, USA*, pages 43–52, 2007. URL https://doi.org/10.1109/ICDM.2007.90.
5. James Bennett and Stan Lanning. The Netflix prize. In *KDD Cup and Workshop in conjunction with KDD*, 2007.
6. Deng Cai, Xiaofei He, Jiawei Han, and Thomas S. Huang. Graph regularized nonnegative matrix factorization for data representation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 33(8): 1548–1560, 2011. URL https://doi.org/10.1109/TPAMI.2010.231.
7. Xiongcai Cai, Michael Bain, Alfred Krzywicki, Wayne Wobcke, Yang Sok Kim, Paul Compton, and Ashesh Mahidadia. Learning collaborative filtering and its application to people to people recommendation in social networks. In *ICDM 2010, The 10th IEEE International Conference on Data Mining, Sydney, Australia, 14–17 December 2010*, pages 743–748, 2010. URL https://doi.org/10.1109/ICDM.2010.159.
8. Allison June-Barlow Chaney, David M. Blei, and Tina Eliassi-Rad. A probabilistic model for using social networks in personalized item recommendation. In *Proceedings of the 9th ACM Conference on Recommender Systems, RecSys 2015, Vienna, Austria, September 16–20, 2015*, pages 43–50, 2015. URL https://doi.acm.org/10.1145/2792838.2800193.
9. Paolo Cremonesi, Yehuda Koren, and Roberto Turrin. Performance of recommender algorithms on top-n recommendation tasks. In *Proceedings of the 2010 ACM Conference on Recommender Systems, RecSys 2010, Barcelona, Spain, September 26–30, 2010*, pages 39–46, 2010. URL https://doi.acm.org/10.1145/1864708.1864721.
10. Wei Di, Neel Sundaresan, Robinson Piramuthu, and Anurag Bhardwaj. Is a picture really worth a thousand words?: - on the role of images in e-commerce. In *Seventh ACM International Conference on Web Search and Data Mining, WSDM 2014, New York, NY, USA, February 24–28, 2014*, pages 633–642, 2014. URL https://doi.acm.org/10.1145/2556195.2556226.
11. Zeno Gantner, Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. Mymedialite: a free recommender system library. In *Proceedings of the 2011 ACM Conference on Recommender Systems, RecSys 2011, Chicago, IL, USA, October 23–27, 2011*, pages 305–308, 2011. URL https://doi.acm.org/10.1145/2043932.2043989.
12. Rayid Ghani and Andrew Fano. Building recommender systems using a knowledge base of product semantics. In *Proceedings of the Workshop on Recommendation and Personalization in ECommerce at the 2nd International Conference on Adaptive Hypermedia and Adaptive Web based Systems*, pages 27–29, 2002.
13. Pranab Ghosh. From explicit user engagement to implicit product rating. URL http://bit.ly/1lidNQn, 2014. [Last accessed 2014-08-11].
14. David Goldberg, David Nichols, Brian M. Oki, and Douglas Terry. Using collaborative filtering to weave an information tapestry. *Commun. ACM*, 35 (12): 61–70, December 1992. ISSN 0001-0782.
15. Quanquan Gu, Jie Zhou, and Chris H. Q. Ding. Collaborative filtering: Weighted nonnegative matrix factorization incorporating user and item graphs. In *Proceedings of the SIAM International Conference on Data Mining, SDM 2010, April 29 - May 1, 2010, Columbus, Ohio, USA*, pages 199–210, 2010. URL https://doi.org/10.1137/1.9781611972801.18.
16. C.-Hennig Hanf and Bernhard Wersebe. Price, quality, and consumers' behaviour. *Journal of Consumer Policy*, 17 (3): 335–348, 1994.
17. Yifan Hu, Yehuda Koren, and Chris Volinsky. Collaborative filtering for implicit feedback datasets. In *Proceedings of the 8th IEEE International Conference on Data Mining (ICDM 2008), December 15–19, 2008, Pisa, Italy*, pages 263–272, 2008. URL https://doi.org/10.1109/ICDM.2008.22.

18. Tomoharu Iwata, Shinji Wanatabe, and Hiroshi Sawada. Fashion coordinates recommender system using photographs from fashion magazines. In *IJCAI 2011, Proceedings of the 22nd International Joint Conference on Artificial Intelligence, Barcelona, Catalonia, Spain, July 16–22, 2011*, pages 2262–2267, 2011. URL http://ijcai.org/papers11/Papers/IJCAI11-377.pdf.
19. Vignesh Jagadeesh, Robinson Piramuthu, Anurag Bhardwaj, Wei Di, and Neel Sundaresan. Large scale visual recommendations from street fashion images. In *The 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, New York, NY, USA - August 24 – 27, 2014*, pages 1925–1934, 2014. URL https://doi.acm.org/10.1145/2623330.2623332.
20. Michael Jahrer, Andreas Töscher, and Robert A. Legenstein. Combining predictions for accurate recommender systems. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, July 25–28, 2010*, pages 693–702, 2010. URL https://doi.acm.org/10.1145/1835804.1835893.
21. Gawesh Jawaheer, Martin Szomszor, and Patty Kostkova. Characterisation of explicit feedback in an online music recommendation service. In *Proceedings of the 2010 ACM Conference on Recommender Systems, RecSys 2010, Barcelona, Spain, September 26–30, 2010*, pages 317–320, 2010. URL https://doi.acm.org/10.1145/1864708.1864776.
22. Mikito Kobayashi, Fumiaki Minami, Takayuki Ito, and Satoshi Tojo. An implementation of goal-oriented fashion recommendation system. In NgocThanh Nguyen and Radoslaw Katarzyniak, editors, *New Challenges in Applied Intelligence Technologies*, volume 134 of *Studies in Computational Intelligence*, pages 77–86. Springer Berlin Heidelberg, 2008.
23. Matt J. Kusner, Yu Sun, Nicholas I. Kolkin, and Kilian Q. Weinberger. From word embeddings to document distances. In *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6–11 July 2015*, pages 957–966, 2015. URL http://jmlr.org/proceedings/papers/v37/kusnerb15.html.
24. Yanen Li, Jia Hu, ChengXiang Zhai, and Ye Chen. Improving one-class collaborative filtering by incorporating rich user information. In *Proceedings of the 19th ACM Conference on Information and Knowledge Management, CIKM 2010, Toronto, Ontario, Canada, October 26–30, 2010*, pages 959–968, 2010. URL https://doi.acm.org/10.1145/1871437.1871559.
25. Jyh-Han Lin and Jeffrey Scott Vitter. A theory for memory-based learning. *Machine Learning*, 17 (2–3): 143–167, 1994. URL https://doi.org/10.1007/BF00993469.
26. Yu-Chu Lin, Yuusuke Kawakita, Etsuko Suzuki, and Haruhisa Ichikawa. Personalized clothing-recommendation system based on a modified bayesian network. In *12th IEEE/IPSJ International Symposium on Applications and the Internet, SAINT 2012, Izmir, Turkey, July 16–20, 2012*, pages 414–417, 2012. URL https://doi.org/10.1109/SAINT.2012.75.
27. Jiahui Liu, Peter Dolan, and Elin Rønby Pedersen. Personalized news recommendation based on click behavior. In *Proceedings of the 2010 International Conference on Intelligent User Interfaces, February 7–10, 2010, Hong Kong, China*, pages 31–40, 2010. URL https://doi.acm.org/10.1145/1719970.1719976.
28. Si Liu, Jiashi Feng, Zheng Song, Tianzhu Zhang, Hanqing Lu, Changsheng Xu, and Shuicheng Yan. Hi, magic closet, tell me what to wear! In *Proceedings of the 20th ACM Multimedia Conference, MM '12, Nara, Japan, October 29 - November 02, 2012*, pages 619–628, 2012. URL https://doi.acm.org/10.1145/2393347.2393433.
29. Shinya Miura, Toshihiko Yamasaki, and Kiyoharu Aizawa. SNAPPER: fashion coordinate image retrieval system. In *Ninth International Conference on Signal-Image Technology & Internet-Based Systems, SITIS 2013, Kyoto, Japan, December 2–5, 2013*, pages 784–789, 2013. URL https://doi.org/10.1109/SITIS.2013.127.
30. Hai Thanh Nguyen, Thomas Almenningen, Martin Havig, Herman Schistad, Anders Kofod-Petersen, Helge Langseth, and Heri Ramampiaro. Learning to rank for personalised fashion recommender systems via implicit feedback. In *Mining Intelligence and Knowledge Exploration - Second International Conference, MIKE 2014, Cork, Ireland, December 10–12, 2014. Proceedings*, pages 51–61, 2014. URL https://doi.org/10.1007/978-3-319-13817-6_6.
31. David M. Nichols. Implicit rating and filtering. In *The DELOS Workshop on Filtering and Collaborative Filtering*, pages 31–36, 1997.

32. Douglas Oard and Jinmook Kim. Implicit feedback for recommender systems. In *Proc. of the AAAI Workshop on Recommender Systems*, pages 81–83, 1998.
33. David M. Pennock, Eric Horvitz, Steve Lawrence, and C. Lee Giles. Collaborative filtering by personality diagnosis: A hybrid memory- and model-based approach. In *Proceedings of the UAI*, pages 473–480. Morgan Kaufman, 2000.
34. Hua Quanping. Analysis of collaborative filtering algorithm fused with fashion attributes. *International Journal of u-and e-Service, Science and Technology*, 8 (10): 159–168, 2015.
35. Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. BPR: bayesian personalized ranking from implicit feedback. In *UAI 2009, Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence, Montreal, QC, Canada, June 18–21, 2009*, pages 452–461, 2009. URL https://dslpitt.org/uai/displayArticleDetails.jsp?mmnu=1&smnu=2&article_id=1630&proceeding_id=25.
36. Andrew I. Schein, Alexandrin Popescul, Lyle H. Ungar, and David M. Pennock. Methods and metrics for cold-start recommendations. In *SIGIR 2002: Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, August 11–15, 2002, Tampere, Finland*, pages 253–260, 2002. URL https://doi.acm.org/10.1145/564376.564421.
37. Edward Yu-Te Shen, Henry Lieberman, and Francis Lam. What am I gonna wear?: scenario-oriented recommendation. In *Proceedings of the 2007 International Conference on Intelligent User Interfaces, January 28–31, 2007, Honolulu, Hawaii, USA*, pages 365–368, 2007. URL https://doi.acm.org/10.1145/1216295.1216368.
38. Push Singh, Thomas Lin, Erik T. Mueller, Grace Lim, Travell Perkins, and Wan Li Zhu. Open mind common sense: Knowledge acquisition from the general public. In *On the Move to Meaningful Internet Systems, 2002 - DOA/CoopIS/ODBASE 2002 Confederated International Conferences DOA, CoopIS and ODBASE 2002 Irvine, California, USA, October 30 - November 1, 2002, Proceedings*, pages 1223–1237, 2002. URL https://doi.org/10.1007/3-540-36124-3_77.
39. Nathan Srebro and Tommi S. Jaakkola. Weighted low-rank approximations. In *Machine Learning, Proceedings of the Twentieth International Conference (ICML 2003), August 21–24, 2003, Washington, DC, USA*, pages 720–727, 2003. URL http://www.aaai.org/Library/ICML/2003/icml03-094.php.
40. Gábor Takács, István Pilászy, Bottyán Németh, and Domonkos Tikk. Scalable collaborative filtering approaches for large recommender systems. *Journal of Machine Learning Research*, 10: 623–656, 2009. URL https://doi.acm.org/10.1145/1577069.1577091.
41. Tran The Truyen, Dinh Q. Phung, and Svetha Venkatesh. Preference networks: Probabilistic models for recommendation systems. In *Data Mining and Analytics 2007, Proceedings of the Sixth Australasian Data Mining Conference (AusDM 2007), Gold Coast, Queensland, Australia, December 3–4, 2007, Proceedings*, pages 195–202, 2007. URL http://crpit.com/abstracts/CRPITV70Truyen.html.
42. G. Vignali and C. Vignali. *Fashion marketing & theory*. Access Press, 2009.
43. Jun Wang, Arjen P. de Vries, and Marcel J. T. Reinders. Unifying user-based and item-based collaborative filtering approaches by similarity fusion. In *SIGIR 2006: Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Seattle, Washington, USA, August 6–11, 2006*, pages 501–508, 2006. URL https://doi.acm.org/10.1145/1148170.1148257.
44. Elizabeth Wilson. *Adorned in dreams: Fashion and modernity*. IB Tauris, 2003.
45. Kenji Araki Ying Zhao. What to wear in different situations? A content-based recommendation system for fashion coordination. In *Proc. of the Japanese Forum on Information Technology (FIT2011)*, 2011.

Part VII
Appendix

Chapter 26
Datasets for Business and Consumer Analytics

Natalie Jane de Vries and Pablo Moscato

Abstract This extended appendix provides information, summaries and methodological details for publicly available datasets and, in particular, those used by various authors throughout this volume. Each of these datasets are publicly available and readers are highly encouraged to investigate these datasets for themselves for the continued journey and challenge of finding new ideas in business and consumer analytics.

Keywords Community detection · Community detection algorithm · Facebook · Game of Thrones

26.1 Introduction

As both the commercial and academic worlds move towards higher levels of transparency and increased use of open-access technologies, greater amounts of information, data and knowledge are shared around the world than ever before. An increasing number of publishing outlets are becoming open-access in order to encourage the open sharing of data, ideas, innovation and knowledge allowing for the greater advancement of research and the understanding of the world we live in. In line with this trend, this volume provides readers with a collection of online publicly available datasets that have been used by some contributions in this volume with additional suggestions for open datasets.

The consequences of data sharing are multiple and greatly beneficial. On one side, it is already impacting how we conduct research and development in all sectors. The sharing of data, and sometimes code, allows enhanced reproducibility of the scientific claims, quick prototyping of new solutions, simplification of secondary analysis, improvement of new mathematical models and algorithms used

N. J. de Vries (✉) · P. Moscato
School of Electrical Engineering and Computing, The University of Newcastle, Callaghan, NSW, Australia
e-mail: natalie.devries@newcastle.edu.au; Pablo.Moscato@newcastle.edu.au

to interrogate the same data, increases public awareness of the importance of digital repositories, establishment of better processes to manage data, strengthening of long-distance collaborations, economy of scale, acceleration of the research enterprise, minimization of safety and ethical risks, etc.

In line with this trend, one that we deeply support in terms of its high objectives for the public good, this volume provides readers with a collection of online publicly available datasets that have been used by some contributions in this volume with additional suggestions for open datasets. Specific details of the datasets are provided including a brief introductory description, their sources, collection methods, availability, size as well as their previous uses and publications. Readers and users of this volume are highly encouraged to investigate these datasets for further inspection, research and advancements of your own methodologies and techniques. This chapter seeks to act as a reference point for these datasets providing the basic understanding and information needed in order to proceed to investigate and use these data in your own research.

26.2 Datasets Used by Contributions in This Book

First, we present several datasets that have been used by chapter(s) in this volume and are publicly available. Some more background information is provided on the datasets, links to the original publishers of the dataset and where the dataset can be found. We have attempted to follow a coherent template so that readers could quickly browse them and have an 'eagle's view' of the main datasets used in the book in a single point of entry.

26.2.1 TripAdvisor Dataset

This dataset was first published by Honging et al. [22] and it is re-studied with a novel technique employed by Rego et al. in Chap. 21. It includes over 235,793 hotel reviews collected from TripAdvisor in 2009. The dataset contains written text reviews as well as scaled responses on certain aspects. The aspects on which TripAdvisor users rated the hotels on a 5-point scale are as follows: *'Value'*, *'Room'*, *'Location'*, *'Cleanliness'*, *'Check-in/Front Desk'*, *'Service'* and *'Business Service'*.

Data Collection Details
The publishers of this dataset crawled 235,793 hotel reviews in a one month period (February 14 to March 15, 2009) to capture a large amount of customers' reviews. After their own data cleaning and pre-processing the dataset contained 1850 hotels and 108,891 reviews. For each review, the free-text part of it was downloaded as well as the seven aspects that the hotel is rated on. The authors' statistics for the

dataset indicate 8.21 ± 4.02 sentences per review and 9.57 ± 6.21 words per aspect. Basic descriptive information of the TripAdvisor dataset is shown in Table 26.1.

Table 26.1 TripAdvisor dataset at a glance

Dataset information	
Year of data published	2009
Number of samples	235,793 hotel reviews (raw data)
Number of reviews/samples	108,891 (after pre-processing)
Number of hotels	1850 (after pre-processing)
Number of variables	Seven aspect ratings and varying amounts of text data per review
Type of dataset	Customers' hotel reviews crawled from the TripAdvisor website which includes 7 scale variables and text data
Data publisher	Honging et al. [22]
Publicly available?	Yes
Date of data collection	March 2009

Previous Publications Using the TripAdvisor Dataset
In [21] and [22], the authors used Latent Aspect Analysis on the text review data. In doing so, the variables relating to the seven aspects rated on a 5-point scale were taken as a form of 'ground truth'. The way the authors did this was to assign 'seed words' that relate to one of the seven aspects. One important factor these authors point out (by looking at an example of comparing three hotels) is that although the three hotels have the same overall ratings, they vary in terms of their specific rating (i.e. value, location and room conditions) [21]. This finding shows one of the many uses of analysing hotel review data. A better analysis of such data, organization, use and implementation of the findings and information could significantly improve a customer's experience when booking or planning a trip or holiday.

26.2.2 The Amazon Co-purchasing Dataset

The full dataset contains product reviews and metadata from Amazon, including 142.8 million reviews spanning May 1996–July 2014. The dataset includes reviews (ratings, text, helpfulness votes), product metadata (descriptions, category information, price, brand and image features) and links (also viewed/also bought graphs). In Chap. 9, a subset of the full Amazon co-purchasing dataset is used for the analysis. Details of this subset, and how it was extracted are provided below.

Dataset Details

The dataset consists of metadata from products and purchases from the website Amazon.com. The dataset[1] is comprised of 191,000 products crawled from *Amazon.com* [14, 15], it uses the JSON[2] (JavaScript Object Notation) data-interchange format. To be able to perform queries on this dataset it has been transferred to a MongoDB[3] document-oriented NoSQL (not only SQL) database. Figure 26.1 shows a document sample and its fields.

```
"_id" : ObjectId("56f35d18b9b515dccfe95c52"),
"asin" : "B000C1VTRU",
"description" : "Introduced in 1992. Fragrance notes: mint, citrus, wood...",
"title" : "Pasha De Cartier By Cartier For Men. Eau De Toilette Spray 3.3 Oz.",
"related" : {"also_bought":["B000QF6SKU","B0020MMBQC","B00GJKBQM6","B0079VTKKO"],
  "bought_together" : ["B000QF6SKU"]},
"price" : 40.26,
"imUrl" : "http://ecx.images-amazon.com/images/I/41KUSBaLSkL._SY300_.jpg",
"brand" : "Cartier",
"categories" : [["Beauty","Fragrance","Men's","Sets"]]
```

Fig. 26.1 A document sample of the dataset used in the case studies involving the Amazon co-purchasing dataset

The fields $ASIN$[4] and *'title'* define the node *'id'* and *'label'*. The field *related* can have four kinds of relationships listed, which are: *'also bought'*, *'also viewed'*, *'bought together'* and *'buy after viewing'*. Not every product has the all four kinds of relationships. The *'also bought'* relationship implies that a customer who bought an item also bought this product, we used this information to connect items, therefore, creating a co-purchasing network of products.

This dataset (and its subsets) is useful for researchers in network analytics since it provides a testbed for their ideas or algorithms in co-purchasing scenarios. It is likely that we have only seen a few preliminary studies that its full potential is still to be delivered. It also provides a good case study for students of data mining and analytics courses to learn and understand the basics of business and customer analytics in a 'real-life' e-business context. We show one such interesting subgraph of interest in the next section.

[1] http://jmcauley.ucsd.edu/data/amazon/links.html.

[2] http://www.json.org.

[3] https://docs.mongodb.com.

[4] Amazon Standard Identification Numbers.

26.2.2.1 The Versace-Centric Co-purchasing Network

In one of this book chapters, entitled*'Overlapping communities in co-purchasing and social interaction graphs: a memetic approach'*, for the Amazon dataset, the authors first select a group of purchased products (*seeds*) to generate a 'brand-centric' network. They selected the items of the Italian luxury fashion company Versace[5],[6] as many co-purchasing patterns involve at least one element of this brand in the dataset. The criteria of choice arose from the following characteristics:

- Versace is a luxury brand which has a presence in e-Commerce.
- The brand covers a diversity of items of different types, e.g. perfumes, accessories, clothes, watches, etc.
- Versace has enough products to produce a network with relevant community structures of interest.
- The co-purchasing network (retrievable from the dataset) may give some insights on other luxury brands that can also be purchased via Amazon.

The final dataset used contained 100 products of the brand Versace that have 'also bought' relationships with other products, including other 470 'non-Versace' products from 176 distinct brands; therefore, the network is comprised of 570 nodes. Each pair of products in the network are connected by an edge regardless of its brand if they have been 'bought together'. This network has 5933 edges.

Previous Publications that Employed the Amazon Co-purchasing Network
Readers may wish to check other publications that use this dataset. In addition to McAuley and Leskovec's paper [14, 15] this dataset was studied in a project in which the 'boundary of brands' was identified [8]. It has also been investigated by other authors [18, 20]. The original Amazon dataset (and its subsets, such as the one used in this book) provides a good base for data scientists to test their new ideas. It also provides a good case study for students of data mining and analytics courses to learn the basics of network analytics and understand their usefulness in a 'real-life' business context (Table 26.2).

26.2.3 Customer Churn Dataset

This is a dataset of customers of a telecommunications service including information about the customers who have 'left' the company (churners) as well as those customers stayed with the company. The dataset is publicly available through the website[7] accompanying *Discovering Knowledge in Data: An Introduction to Data*

[5]http://www.versace.com.

[6]https://en.wikipedia.org/wiki/Versace.

[7]Check for the file 'churn.txt' found inside the compressed file at: http://dataminingconsultant.com/DKD2e_data_sets.zip.

Table 26.2 Amazon co-purchasing dataset at a glance

Dataset information	
Year of data published	2014
Number of reviews	142.8 million
Product variables available	Metadata for 9.4 million products
Type of dataset	Product co-purchasing network
Data publisher	McAuley et al. [14, 15]
Publicly available?	Yes
Date of data collection	May 1996–July 2014

Mining by Larose [12]. It was originally published (and is still available) through the UCI Machine Learning Repository of The University of California Irvine.[8]

Detailed Data Information

The dataset contains 20 attributes (variables) of 3333 customers along with the 'churning' information. The total number of 'churners' in this dataset is 483. The 20 attributes are outlined and explained in a paper by Obiedat et al. [17] who also used this dataset in a clustering and genetic programming study. As stated, the data is publicly available through the UCI Machine Learning Data Repository. The 20 attributes available for each customer are:

- **State:** a categorical variable, for the 50 states and the district of Columbia.
- **Account length:** an integer-valued variable that measures for how long the account has been active.
- **Area code:** a categorical variable.
- **Phone number:** essentially a surrogate key for customer identification.
- **International Plan:** a dichotomous categorical (i.e. having 'Yes' or 'No' values).
- **VoiceMail Plan:** another dichotomous categorical variable having 'Yes' or 'No' values.
- **Number of voice mail messages:** an integer-valued variable.
- **Total day minutes:** a continuous variable, the number of minutes the customer has used the service during the day.
- **Total day calls:** an integer-valued variable.
- **Total day charge:** a continuous variable based on foregoing two variables.
- **Total evening minutes:** a continuous variable, the number of minutes the customer has used the service during the evening.
- **Total evening calls:** an integer-valued variable.
- **Total evening charge:** a continuous variable based on the previous two variables values.
- **Total night minutes:** a continuous variable, the total number of minutes the customer has used the service during the night.
- **Total night calls:** an integer-valued variable.

[8]https://archive.ics.uci.edu/ml/datasets.html.

- **Total night charge:** a continuous variable based on the other two variables.
- **Total international minutes:** a continuous variable, the number of minutes customer has used service to make international calls.
- **Total international calls:** an integer-valued variable.
- **Total international charge:** a continuous variable based on previous two variables.
- **Customer Service calls:** Calls to customer service representatives.

Further basic descriptive information is provided in Table 26.3.

Table 26.3 Customer churn dataset at a glance

Dataset information	
Year of data published	Unknown
Number of samples	3333 customers
Number of variables	23 (including churning information)
Type of dataset	Customer attrition dataset from telecommunications company
Data publisher	UCI machine learning data repository
Publicly available?	Yes
Date of data collection	Unknown

Previous Publications

The Customer Churn Dataset was extensively used by Larose [12] in a study presenting various data analytics and statistical problems in a textbook-style publication. Since then, it has been used and/or cited by many researchers.

Verbeke et al. [49] used rule-based classification algorithm based on Ant Colony Optimization for churn prediction. Generated rules are combined using RIPPER and C4.5 classifier to improve the accuracy in churn prediction. Sharma et al. [48] proposed a Neural Network (NN) to predict the customer churn rate. Their approach able to achieve 92.35% accuracy and identified *Customer Service Calls, International Plan, Day Minutes* and *Day Charge* are the most important variables to identify the churners. Rodriguez and Shin [43] used decision trees to predict customer churn rate and also studied the impact of features which were selected using principal component analysis (PCA). They report 13 influential attributes for the dataset that have a positive impact on the churn prediction. Lima et al. [38] used decision table to predict and utilized domain knowledge to understand the churner from the dataset. Their approach used 8 features from the dataset to create the decision table to explain the churner behaviours of the customer. Amin et al. [24] used rough set approach for churn prediction from the dataset. They applied information gain attribute rank approach and selected top 11 features to train the classifier. Recently, Yildiz and Albayrak [54] used a rotation forest approach to predict churn from the dataset. They used top 10 features selected by an information gain approach to train the algorithm which achieved 92.49% accuracy in their experimental settings. From these works we observe that several researchers have used a feature selection step prior to employ them as inputs for the classification methods.

26.2.4 Online Customer Engagement Dataset

This is a small dataset based on a survey in online consumer behaviour, customer engagement and social media [5]. Customers of brand pages on a social media platform (Facebook) answered questions about their engagement and involvement with the brand on Facebook as well as other consumer behaviours towards the brand, their value perceptions and some loyalty measures.

Data Collection and Questionnaire Tool

The data was collected through a paper-and-pen survey method at university based in New South Wales, Australia, in August 2013. The questionnaires were administered to existing Facebook users only who had previously used a Facebook brand page. In the questionnaire, each item was measured on a seven-point *Likert* scale anchored with (1) (for *'strongly disagree'*) to (7) (for *'strongly agree'*) as is consistent with other social science studies [5]. Questions were grouped by their 'theoretical construct' and each construct included between 3 and 6 items.

A total of 452 surveys were collected and after screening and data cleaning 371 usable surveys remained. Preliminary data analysis was conducted using the Statistical Software package SPPS v21 to test for the distribution of the data and gather an initial understanding of the dataset. Basic demographic information on the whole dataset was also computed. In the total sample there were 58.2% females and 41.8% males with an age range in the sample of 17–49. The average age in the total sample was 21.4. In total, 69 items were part of the survey which means the entire dataset is a 371×69 matrix ready for computation of a distance or correlation measure to be the basis of the clustering process. The basic descriptive information of the dataset is shown 'at a glance' in Table 26.4.

Table 26.4 Customer engagement dataset at a glance

Dataset information	
Year of data published	2014
Number of samples	371
Number of variables	69 variables + demographic information
Type of dataset	Survey Data with 7-scale variable questions
Data publisher	de Vries et al. [5]
Publicly available?	Yes
Date of data collection	August 2013

Primary Data Collection Ethics' Statement

The Review Board of The University of Newcastle Ethics Committee (Ethics Approval Number: H-2013-0226) approved the original data collection and the study. The questionnaire tool was anonymous and no identifying information was asked of participants. Prior to participation, respondents were given a 'Participation Information Statement' including all necessary information. Participation in this study was entirely voluntary (with no incentive given) and participants were able to discontinue their participation at any time whilst completing the survey.

Previous Publications

Besides the chapters in this book it is used in, the data has previously also been used and published in several articles produced between 2014 and 2017 by some of the contributors of this book and their collaborators [5, 6, 35]. These references complement the study presented in this book with the different perspective which have been derived from its previous analyses.

26.2.5 *Wine Quality Dataset*

This dataset includes two datasets that are related to red and white variants of the Portuguese 'Vinho Verde' wine [3]. Physicochemical (inputs) and sensory (the output) variables are available and due to privacy and logistics issues, there is unfortunately no data about grape types, wine brand, wine selling price, etc.

Modelling wine quality is an interesting topic for wine makers and computational researchers alike as wine quality evaluation is often part of the wine certification process, can be used to improve wine making and it provides challenging computational task for data mining researchers. This dataset is used by Chap. 16 in this book as a case study example of using a memetic algorithm for visualizing products. Table 26.5 provides some basic descriptive information of this dataset to provide a 'glance' of the wine quality dataset.

Table 26.5 Wine quality dataset at a glance

Dataset information	
Year of data published	2009
Number of samples	4898
Number of variables	12
Type of dataset	Wine quality dataset including quality score and chemical values information
Data publisher	Cortez et al. [3]
Publicly available?	Yes
Date of data collection	Unknown

Dataset Collection/Attribute Details

The features/attributes of this dataset are the physicochemical properties of the wines that are tested. These variables, individually and combined, have a great impact on the quality and the taste of the wine. However, researchers have not been able to find a direct correlation between the relationship of these variables and the qualities of the wines. There are 12 attributes and 4898 instances. The attributes are shown here.

Input variables:
Fixed acidity
Volatile acidity
Citric acid
Residual sugar
Chlorides
Free sulphur dioxide
Total sulphur dioxide
Density
pH
Sulphates
Alcohol
Output variable:
Quality (As given by wine experts' ratings using a score between 0 and 10)

Previous Publications
This dataset was developed and published by Cortez et al. [3]. The publishers state: 'These datasets can be viewed as classification or regression tasks. The classes are ordered and not balanced (e.g. there are much more normal wines than excellent or poor ones). Outlier detection algorithms could be used to detect the few excellent or poor wines'.

26.2.6 Other Small Datasets Used in This Volume

Throughout this volume several small other datasets are used, either as explanatory case examples or as benchmarking datasets for methods. We provide some basic information on them here.

26.2.6.1 Marvel Comic Book World Network Dataset

This network is the result of counting the 'interactions' of the characters that make part of the 'virtual world' developed by the company Marvel Comics over many years. Originally called Timely Comics Inc., the company has been publishing comic books for several decades [4]. Users of the online platform[9] subscribe monthly or yearly to get access to the comics. Lately, Marvel's characters' popularity has been revitalized due to the release of several recent movies that

[9] http://marvel.com/comics.

recreate some parts of the wealth of content the company has but now using spectacular modern special effects and blockbuster productions. This said, the interactions in movies may turn to be another ‘universe’ in its own right.

This virtual network composed of characters’ interactions has attracted the attention of many researchers interested in comparing its characteristics to the networks that are the result from the interactions of real-world collaborations, such as the Hollywood network, or the one created by scientists who work together in producing research papers [1].

This dataset gives the authors of Chap. 16 the opportunity to present the results of their new technique for visualization on a network of 6418 different characters (each one represented by a different node of a weighted undirected graph). These characters appear in 12,942 comic books. In a way, this dataset can be thought of as a social network just in the way real social networks and relations can be downloaded from social media and networking sites such as Facebook and Twitter. This dataset was analysed and visualized to test a novel visualization method in Chap. 16 of this volume.

26.2.6.2 Les Miserables Co-appearance Network Dataset

This dataset is based on a network produced by accounting for the co-appearances of characters in Victor Hugo’s novel *Les Miserables*. Nodes represent the characters and edges connect any pair of characters that appear in the same chapter of the book. This network is available online.[10] It consists of 77 nodes and 254 edges and it is commonly used as a benchmark dataset for basic research in community detection algorithms. See, for instance, Chap. 9 of this volume or many other research papers using this dataset [2, 9, 19].

Previous Use of the Miserables Dataset
The network and network visualizations are available for download at the FigShare repository[11] [32].

In 2005, Bagrow et al. used the Les Miserables network to test their Local method for detecting communities [2]. Evans and Lambiote used line graphs to build the overlapping communities of the characters of the novel Les Miserables published in the article *‘Line graphs of weighted networks for overlapping communities’* in September of 2010 [33]. The same year, Liu and Liu used the Les Miserables network as a dataset for testing a simulated annealing with the *k*-means algorithm for community detection [40]. Also in 2010, Lancichinetti et al. used it for showing the statistical significance of communities in networks in [37]. Psorakis et al. used it for testing an overlapping community detection method using Bayesian non-negative matrix factorization published in 2011 [19].

[10] http://www-personal.umich.edu/~mejn/netdata/lesmis.zip.

[11] https://doi.org/10.6084/m9.figshare.1573032.v1.

26.2.6.3 ‘Game of Thrones’ Co-appearance Network Dataset

This dataset includes the network of co-appearances of characters in the popular series *Game of Thrones*. It was actually derived from *‘A Storm of Swords’*[12,13] which is the multi-awarded third novel part of the book series *‘A Song of Ice and Fire’*, written by Martin [41]. The book was first published on August 2000, in the UK and featured the third season of the famous TV show *‘Game of Thrones’* first broadcasted by HBO on March 2013.

In some sense, it should be really be named as the *‘A Storm of Swords’* Co-Appearance Dataset. This novel poses an elaborate chain of plots and events performed by its characters in the fictional continents of *Westeros* and *Essos*. The main history focuses on the ‘Iron Throne of the Seven Kingdoms’ and follows a web of alliances, conspiracies and battles among the dynastic noble families either rivalling to claim the throne or fighting for independence.

This dataset is featured in the journal article *‘Network of Thrones’* in the scientific journal *Math Horizons*, published in April 2016 where the authors use community detection to identify groups of characters which frequently interact [28]. Liu and Albergante used the dataset in the paper *‘Balance of Thrones: a network study on Game of Thrones that unveils predictable popularity of the story’*, in which they *‘Constructed a social network of houses’* according to the families featured in the novel *‘A Storm of Swords’* [39].

26.3 Other Online Available Datasets for Business and Consumer Analytics Research

In order to provide the reader with further materials to pursue their own data and business analytics journey further, we have provided some details of other open access datasets. Some have been previously used by authors of this volume, others are simply interesting datasets that are openly available in the literature and online which we would like to highlight for consideration.

26.3.1 Charities Dataset

This dataset consists of a set of responses from a quantitative survey conducted on behalf of the Australian Charities and Not-for-profits Commission (ACNC) by ChantLink in 2013. The data as well as the survey and report are publicly available online and are maintained by the Office of the Australian Government http://data.

[12]https://www.macalester.edu/~abeverid/thrones.html.

[13]https://networkofthrones.wordpress.com.

gov.au/dataset/trust-and-confidence. This survey collected information about levels of trust and confidence in charities and factors that may affect these levels. It also collected information about awareness and support for a national regulator of charities and interest in a public register of charities.

Data Collection Details

The survey was conducted by ChantLink online between 22 and 29 April 2013 and obtained 1624 complete responses (including a pilot phase of 60 responses). Not every participant answered every question. Depending on the nature of some responses, individuals were directed to different sections of the survey and, consequently, they were forced to avoid answering several questions.

Some of the remaining questions, which, for instance, relate to the respondents' demographic information, are subsequently used to highlight possible differentiating characteristics of the clusters. We do not include this information in the clustering method, since segmenting a market based on consumers' behaviours, trust and confidence in charities is our primary goal and it provides a more detailed cohort analysis than segmenting based on demographic information alone. Table 26.6 provides basic descriptive information of the charities dataset.

Table 26.6 Charities dataset at a glance

Dataset information	
Year of data published	2013
Number of samples	1624 (including 60 pilot responses)
Number of variables	170 (raw data)
Type of dataset	Online survey responses
Data publisher	ACNC through ChantLink
Publicly available?	Yes
Date of data collection	April 2013

Previous Publications

Besides the ACNC's own publication report, this dataset has previously been investigated using clustering as well as community detection algorithms.

This dataset was the object of reanalysis in 2015 by de Vries et al. [7] in a clustering study using the MST-kNN algorithm explained in Chap. 3. The consumers were clustered based on their survey responses and subsequently, models to predict 'Involvement' levels were found for each cluster. Varying models and different clusters were found in this study.

Furthermore, Naeni et al. [16] conducted a community detection study on this dataset attempting to uncover groups of individuals who are similar to each other in terms of their trust and confidence in and donating behaviour towards charities and not-for-profits. In [16], the authors present an earlier version of the novel community detection algorithm that is employed in Chap. 9.

26.3.2 *Fashion 10000 Dataset*

This dataset is a large social image dataset made public by Loni et al. [13]. The dataset contains more than 32,000 images, their context and social metadata. As the authors explain, the images have been selected through a crawling strategy, retrieving only those images that have a Creative Common (CC) Attribution licence (hence allowing this dataset, its images and content to be freely used and shared for research and commercial purposes).

Data Collection Method
The images were crawled from the Flickr, the social photo sharing website by using Flickr's API. Query terms for this were potential fashion-related keywords that the authors extracted from Wikipedia. In total, a set of 470 fashion categories were collected from the Wikipedia index page of fashion items (such as shirt, tuxedo). The dataset was collected in June 2012 but the uploading dates of the photos range from March 2005 to July 2012 [13]. In total, 32,398 photos were downloaded and their social metadata were distributed in 262 fashion categories. Table 26.7 provides further descriptive information of the dataset. For the Annotation task, the authors employed more than 8000 Human Intelligent Tasks (HITs) on an online crowdsourcing platform. For each image, the HITs were expected to annotate six different aspects; *'Fashion/Clothing Related?'*, *'Specialty clothing item'*, *'Number of people in the picture'*, *'Professional model or not?'*, *'Person wearing fashion?'* and *'Formal/Informal'*. After this annotation process, the authors conducted statistical agreement analysis to ensure the robustness of the annotated information.

Table 26.7 Fashion 10000 dataset at a glance

Dataset information	
Year of data published	2013
Number of samples	32,398 photos (raw data—with attached metadata)
Number of fashion categories	262
Type of dataset	Fashion images with attached metadata, social data and human-generated annotation data
Data publisher	Loni et al. [13]
Publicly available?	Yes
Date of data collection	March 2005–July 2012

Previous Publications
As stated, this dataset was crawled, collected, annotated and published by Loni et al. and published at the Multimedia Systems Conference in 2014 [13].

26.3.3 MovieLens Dataset and Other 'Lens' Datasets

The MovieLens dataset(s) have been popularly used in research, education and industry, and they are downloaded thousands of times each year [10]. It is a useful dataset for business analytics researchers as it provides interesting problems relating to recommender systems, consumer preference prediction and more. It was first released in 1998 and it describes people's self-expressed preferences for movies. It was collected by a team of researchers (called GroupLens) at the University of Minnesota (https://grouplens.org/datasets/movielens/). They based MovieLens on a prior recommender system they had created named *EachMovie*. However, the difference was that with MovieLens they incorporated a Collaborative Filtering (CF) Approach to recommendations. Chapter 11 of this volume explains and elaborated on recommender systems and the different types.

Data Collection Details and Description
The website for MovieLens is in fact an online recommender system in which users express their preferences and can gain personalized recommendations from using the platform. The preference data is the result of a person expressing a preference (a 0–5 star rating) for a movie at a particular time. The research team collected many of the users' behaviours, preferences and ratings anonymously over different periods of time. You can find different sizes datasets and their dates through the GroupLens website where the datasets can be freely downloaded.[14]

The various datasets available have been annotated by the GroupLens team and they have provided extensive instructions and highlighted those datasets that are better for education or research. Briefly, we provide some information on the *MovieLens 20M Dataset* taken from the above website. In the data, MovieLens users were selected at random for inclusion and their ids have been anonymized. Only movies with at least one rating or tag are included in the dataset. These movie ids are consistent with those used on the MovieLens website which makes it easier to interpret. Further included are files including ratings, tags, the genre of the movie (with a possibility of 18 genres or no genre listed), a file containing links to the movie's other links such as its IMDb page and its 'tag genome'. Overall, the GroupLens group provided highly interesting datasets for consumer analytics, particularly for problems related to recommendations, preference prediction and online consumer behaviour patterns.

26.3.3.1 WikiLens Dataset

Besides the MovieLens dataset, the team at the University of Minnesota have collected (and made public) some related datasets, including the *WikiLens Dataset*-https://grouplens.org/datasets/wikilens/. The WikiLens dataset was a generalized

[14] https://grouplens.org/datasets/movielens/.

collaborative recommender system that allowed its community to define item types (e.g. beer) and categories (e.g. microbrews, pale ales, stouts), and then rate and get recommendations for items. The website notifies users now that 'It was taken offline in 2009 due to lack of system maintenance and support'. However, the dataset is still available online for interested researchers to use.

Previous Publications
Harper et al. [10] provide a comprehensive collection and summary on the work and research conducted using the MovieLens dataset in their publication titled *'The MovieLens Datasets: History and Context'*. For those readers interested in learning more we refer them to this comprehensive review by Harper et al. Some further examples of papers using the MovieLens dataset include the work by Karumur et al. [11] who investigated the value of personality in preference prediction. As Chap. 2 of this book explained, consumer behaviours and preferences are often highly heterogeneous and cannot be predicted by one type of descriptive factor alone. This makes the work by Karumur et al. [11] very interesting as they highlight personality types in the context of recommender systems. Furthermore, the MovieLens dataset has been used in other studies such as the one by Zeng et al. [23] who investigate trend prediction using MovieLens, Netflix and Digg data. Human behaviours such as ratings may be useful indicators for predicting product popularity or social trends.

26.3.4 Yahoo's Webscope Program

The company Yahoo! has made available a number of datasets that are said to be of scientific and commercial interest. According to the information given they are conforming the company's 'data protection standards, including strict controls on privacy'. Academics around the world are able to use them if they are a faculty member, a student or a research employee. We note that several conditions may apply to different academic institutions and we suggest always to consider a consultation with the group that governs ethical access of use of information that regulates their practices. We know that 'available on the web' does not necessarily guarantee that members of an academic institution can access and use the information, so we recommend the proper channels of communication with their ethical boards before its use. Information about this initiative is available online from the Yahoo's Webscope Program URL.[15] More than 30 datasets have been made available. Here we briefly summarize some of them.

[15] https://webscope.sandbox.yahoo.com.

26.3.4.1 Yahoo! Search Marketing Advertiser Bidding Data

Yahoo! has operated an auction-based platform for their advertising sells of the right to appear next to the result of particular queries. The company estimates that this sector of the company alone was responsible of nearly two billion US dollars of revenue in 2005. This is a quite dynamic environment with a high-level of competition between bidders. The dataset contains bids over time of all advertisers (anonymized) for the top 1000 search queries in during one year period (June 15, 2002, to June 14, 2003). This dataset has generated some interests during the past years; see, for instance, [25, 29, 31].

26.3.5 The 'Learning to Rank' Challenge

At present, most search engines, if not all, are based on ranking mechanisms by which information is provided to users. This has led to a huge interest in developing new heuristic methods to help address key issues in this domain. In some cases, ranking in web search is seen as an optimization problem which can be dealt with as a supervised machine learning problem. Assuming that we can order as an array some information of interest of a given document, a paired query associated with it, given a certain label associated with the pair the task is then to 'learn' from the array info in the label.[16] We may be required to 'rank' the documents according to relevance, etc. See [42, 50–53] for articles that have used it.

26.3.6 Yahoo! Music User Ratings

A dataset was made available containing over 717 million ratings of 136 thousand songs given by 1.8 million users of the services of the company (data collected between 2002 and 2006). Each song in the dataset has been accompanied by information on the artist, album, and genre (of at least 20 main ones) and a genre hierarchy is also given. A mapping from the genre id's to genre as well as the genre hierarchy is also given. Reviewers are guaranteed to have rated at least 30 songs. Analogously, each song in the dataset was rated by at least 20 reviewers. This dataset was used by several researchers, see [45–47].

[16]Note: most researchers call them 'vectors' but 'array' is probably more correct.

26.3.7 Yahoo! Movies User Ratings and Descriptive Content Information, v.1.0

This dataset is a sample of the company members' preferences for movies (on a scale from A+ to F). Whilst users are anonymized, descriptive information is given for movies released prior to November 2003 (e.g. cast, crew, synopsis, genre, average ratings, awards, etc.). Several papers have been published with this dataset including: [26, 27, 30, 34, 36, 44].

26.3.8 Santander Customer Satisfaction Dataset

Customer satisfaction is a key measure of success used by frontline support teams to senior executives (C-suites). An unhappy customer is most likely to be a 'churner'. Moreover, unhappy customers usually raise their dissatisfaction just after leaving the company. Hence, identifying unhappy customers at an early stage is very important to reduce the churner rate of an organization.

Santander Bank asked help in the popular data analytic competition platform, Kaggle Inc. ('Kaggle'),[17] for identifying dissatisfied customers early in their relationship. The *Santander Customer Satisfaction* competition is accessible at https://www.kaggle.com/c/santander-customer-satisfaction. The data can be downloaded upon registration at the Kaggle website, and the registration is free. The objective of the Santander Bank, by running a competition at Kaggle, was to take the advantages of analysts to identify the unhappy customers. From the competition outcome, they can take proactive steps to improve a customer's happiness before it is too late.

The dataset consists of 370 anonymized numerical features for about 151,000 customers. The dataset is divided into two sets: training and testing data. The train set consists of the 76,020 samples of customer's data, and test set consists of 75,818 customer's data samples. The task is to predict the probability that each customer in the test set is an unsatisfied customer. The submissions by the competitors were evaluated on area under the ROC curve between the predicted probability and the observed target. Total 5073 competitors participated in the competition and the winning solution achieved 0.829072 value for the evaluation score.

[17] https://www.kaggle.com.

26.4 Other Collections of Interest

There are a number of practitioners and researchers that maintain datasets of potential interest for the development of new techniques. We include links to one of them which can be particularly useful due to its variety.[18]

- *Last.fm*—Music Recommendation Data Sets http://www.dtic.upf.edu/~ocelma/MusicRecommendationDataset/index.html
- *Audioscrobbler*—Music Recommendation Data Sets http://www-etud.iro.umontreal.ca/~bergstrj/audioscrobbler_data.html
- *Amazon*—Audio CD recommendations http://131.193.40.52/data/
- *Institut für Informatik, Universität Freiburg*—Book Ratings Data Sets http://www.informatik.uni-freiburg.de/~cziegler/BX/
- *Chicago Entree*—Food Ratings Data Sets http://archive.ics.uci.edu/ml/datasets/Entree+Chicago+Recommendation+Data
- *Amazon*—Product Recommendation Data Sets http://131.193.40.52/data/
- *Nursing Home*—Provider Ratings Data Set http://data.medicare.gov/dataset/Nursing-Home-Compare-Provider-Ratings/mufm-vy8d
- *Hospital Ratings*—Survey of Patients Hospital Experiences http://data.medicare.gov/dataset/Survey-of-Patients-Hospital-Experiences-HCAHPS-/rj76-22dk
- Dating Recommendation http://www.libimseti.cz
- *Dating website recommendation (collaborative filtering)* http://www.occamslab.com/petricek/data/
- *Scholarly Paper Recommendation* http://www.comp.nus.edu.sg/~sugiyama/SchPaperRecData.html
- *Astro Physics collaboration network of authors* http://snap.stanford.edu/data/ca-AstroPh.html
- *Citation network of Pubmed's scientific publication on diabetes* https://github.com/jcatw/scnn/tree/master/scnn/data/Pubmed-Diabetes
- *Fashion-MNIST* https://github.com/zalandoresearch/fashion-mnist
- *Friendship network on Flick* http://socialcomputing.asu.edu/datasets/Flickr
- *The dataset of Slovakia's popular online-social network site named Pokec* http://snap.stanford.edu/data/soc-pokec.html
- *Spammers social network* https://linqs-data.soe.ucsc.edu/public/social_spammer/

26.5 Conclusions

We have provided in this chapter, in a relatively unified manner, details on some of the main datasets used in collection of research articles. In addition, we have also provided some pointers to other datasets that may be of interest as they

[18]https://gist.github.com/entaroadun/1653794.

provide a platform for studying the scalability of some algorithms and also open new questions of research interest. We hope that, with ethical procedures for data collection well established, the collection of public interest datasets will continue to grow. Open data initiatives may end up providing a firm ground for the ultimate quest of establishing data science for the social good.

Acknowledgements P.M. acknowledges this research by The University of Newcastle, and previous funding from the Australian Research Council grants Future Fellowship FT120100060 and Discovery Project DP140104183. P.M. also thanks M.N. Haque and A. Gabardo for discussions and in particular for verifying that all links cited in this chapter were operational at the time of completion. The authors would also like to thank all the publishers of the publicly available datasets and the authors and editors for supporting the open sharing of data and information.

References

1. Ricardo Alberich, Joe Miro-Julia, and Francesc Rosselló. Marvel universe looks almost like a real social network. *arXiv preprint cond-mat/0202174*, 2002.
2. James P. Bagrow and Erik M. Bollt. Local method for detecting communities. *Phys. Rev. E*, 72:046108, Oct 2005.
3. Paulo Cortez, António Cerdeira, Fernando Almeida, Telmo Matos, and José Reis. Modeling wine preferences by data mining from physicochemical properties. *Decision Support Systems*, 47(4):547–553, 2009.
4. Les Daniels. *Marvel: Five fabulous decades of the world's greatest comics*, volume 1. Harry N Abrams Inc, 1991.
5. Natalie Jane de Vries, Jamie Carlson, and Pablo Moscato. A data-driven approach to reverse engineering customer engagement models: Towards functional constructs. *PLOS ONE*, 9:e102768, 2014.
6. Natalie Jane de Vries and Jamie Carlson. Examining the drivers and brand performance implications of customer engagement with brands in the social media environment. *Journal of Brand Management*, 21(6):495–515, 2014.
7. Natalie Jane de Vries, Rodrigo Reis, and Pablo Moscato. Clustering consumers based on trust, confidence and giving behaviour: Data-driven model building for charitable involvement in the Australian not-for-profit sector. *PLOS ONE*, 10(4):1–28, 04 2015.
8. Ademir C. Gabardo, Regina Berretta, Natalie Jane de Vries, and Pablo Moscato. Where does my brand end? An overlapping community approach. In *Intelligent and Evolutionary Systems: The 20th Asia Pacific Symposium, IES 2016, Canberra, Australia, November 2016, Proceedings*, pages 133–148. Springer, 2017.
9. David F. Gleich and C. Seshadhri. Vertex neighborhoods, low conductance cuts, and good seeds for local community methods. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '12, pages 597–605, New York, NY, USA, 2012. ACM.
10. F. Maxwell Harper and Joseph A. Konstan. The MovieLens datasets: History and context. *TiiS*, 5(4):19:1–19:19, 2016.
11. Raghav Pavan Karumur, Tien T. Nguyen, and Joseph A. Konstan. Exploring the value of personality in predicting rating behaviors: A study of category preferences on MovieLens. In *Proceedings of the 10th ACM Conference on Recommender Systems, Boston, MA, USA, September 15–19, 2016*, pages 139–142, 2016.
12. Daniel T. Larose. *Discovering Knowledge in Data: An Introduction to Data Mining*. Wiley-Interscience, 2004.

13. Babak Loni, Lei Yen Cheung, Michael Riegler, Alessandro Bozzon, Luke Gottlieb, and Martha Larson. Fashion 10000: an enriched social image dataset for fashion and clothing. In *Multimedia Systems Conference 2014, MMSys '14, Singapore, March 19–21, 2014*, pages 41–46, 2014.
14. Julian McAuley, Rahul Pandey, and Jure Leskovec. Inferring networks of substitutable and complementary products. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 785–794. ACM, 2015.
15. Julian McAuley and Alex Yang. Addressing complex and subjective product-related queries with customer reviews. *arXiv preprint arXiv:1512.06863*, 2015.
16. Leila Moslemi Naeni, Natalie Jane de Vries, Rodrigo Reis, Ahmed Shamsul Arefin, Regina Berretta, and Pablo Moscato. Identifying communities of trust and confidence in the charity and not-for-profit sector: A memetic algorithm approach. In *2014 IEEE Fourth International Conference on Big Data and Cloud Computing, BDCloud 2014, Sydney, Australia, December 3–5, 2014*, pages 500–507, 2014.
17. Ruba Obiedat, Mouhammd Alkasassbeh, Hossam Faris, and Osama Harfoushi. Customer churn prediction using a hybrid genetic programming approach. *Scientific Research and Essays*, 8:1289–1295, 2013.
18. John O'Donovan, Shinsuke Nakajima, Tobias Höllerer, Mayumi Ueda, Yuuki Matsunami, and Byungkyu Kang. A cross-cultural analysis of explanations for product reviews. In Peter Brusilovsky, editor, *Proceedings of the Joint Workshop on Interfaces and Human Decision Making for Recommender Systems co-located with ACM Conference on Recommender Systems*, Boston, MA, USA, Sep 2016.
19. Ioannis Psorakis, Stephen Roberts, Mark Ebden, and Ben Sheldon. Overlapping community detection using Bayesian non-negative matrix factorization. *Phys. Rev. E*, 83:066114, Jun 2011.
20. Mengting Wan and Julian McAuley. Modeling ambiguity, subjectivity, and diverging viewpoints in opinion question answering systems. In *preprint arXiv*, 2016.
21. Hongning Wang, Yue Lu, and Chengxiang Zhai. Latent aspect rating analysis on review text data: a rating regression approach. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, July 25–28, 2010*, pages 783–792, 2010.
22. Hongning Wang, Yue Lu, and ChengXiang Zhai. Latent aspect rating analysis without aspect keyword supervision. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Diego, CA, USA, August 21–24, 2011*, pages 618–626, 2011.
23. An Zeng, Stanislao Gualdi, Matus Medo, and Yi-Cheng Zhang. Trend prediction in temporal bipartite networks: the case of MovieLens, Netflix, and Digg. *Advances in Complex Systems*, 16(4–5), 2013.
24. Adnan Amin, Saeed Shehzad, Changez Khan, Imtiaz Ali, and Sajid Anwar. *Churn Prediction in Telecommunication Industry Using Rough Set Approach*, pages 83–95. Springer International Publishing, Cham, 2015.
25. Jason Auerbach, Joel Galenson, and Mukund Sundararajan. An empirical analysis of return on investment maximization in sponsored search auctions. In *Proceedings of the 2Nd International Workshop on Data Mining and Audience Intelligence for Advertising*, ADKDD '08, pages 1–9, New York, NY, USA, 2008. ACM.
26. Linas Baltrunas and Francesco Ricci. Locally adaptive neighborhood selection for collaborative filtering recommendations. In Wolfgang Nejdl, Judy Kay, Pearl Pu, and Eelco Herder, editors, *Adaptive Hypermedia and Adaptive Web-Based Systems, 5th International Conference, AH 2008, Hannover, Germany, July 29 - August 1, 2008. Proceedings*, volume 5149 of *Lecture Notes in Computer Science*, pages 22–31. Springer, 2008.
27. Linas Baltrunas and Francesco Ricci. Item weighting techniques for collaborative filtering. In Bettina Berendt, Dunja Mladenic, Marco de Gemmis, Giovanni Semeraro, Myra Spiliopoulou, Gerd Stumme, Vojtech Svátek, and Filip Zelezný, editors, *Knowledge Discovery Enhanced with Semantic and Social Information*, volume 220 of *Studies in Computational Intelligence*, pages 109–126. 2009.

28. Andrew Beveridge and Jie Shan. Network of thrones. *Math Horizons*, 23(4):18–22, 2016.
29. Tilman Börgers, Ingemar Cox, Martin Pesendorfer, and Vaclav Petricek. Equilibrium bids in sponsored search auctions: Theory and evidence. *American Economic Journal: Microeconomics*, 5(4):163–187, 2013.
30. Travis Ebesu and Yi Fang. Neural semantic personalized ranking for item cold-start recommendation. *Inf. Retr. Journal*, 20(2):109–131, 2017.
31. Benjamin Edelman and Michael Ostrovsky. Strategic bidder behavior in sponsored search auctions. *Decision Support Systems*, 43(1):192–198, 2007.
32. Tim Evans. Information on Les Miserables network used in Evans and Lambiotte 2010. 10 2015.
33. Tim S Evans and Renaud Lambiotte. Line graphs of weighted networks for overlapping communities. *The European Physical Journal B-Condensed Matter and Complex Systems*, 77(2):265–272, 2010.
34. Jing Gao, Wei Fan, Yizhou Sun, and Jiawei Han. Heterogeneous source consensus learning via decision propagation and negotiation. In John F. Elder IV, Françoise Fogelman-Soulié, Peter A. Flach, and Mohammed Javeed Zaki, editors, *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Paris, France, June 28 - July 1, 2009*, pages 339–348. ACM, 2009.
35. Mohammad Nazmul Haque, Luke Mathieson, and Pablo Moscato. A memetic algorithm for community detection by maximising the Connected Cohesion. In *Proceedings of 2016 IEEE Symposium Series on Computational Intelligence (SSCI)*, Hawaii, USA, 2017. IEEE.
36. Rachsuda Jiamthapthaksin, Christoph F. Eick, and Ricardo Vilalta. A framework for multi-objective clustering and its application to co-location mining. In Ronghuai Huang, Qiang Yang, Jian Pei, João Gama, Xiaofeng Meng, and Xue Li, editors, *Advanced Data Mining and Applications, 5th International Conference, ADMA 2009, Beijing, China, August 17–19, 2009. Proceedings*, volume 5678 of *Lecture Notes in Computer Science*, pages 188–199. Springer, 2009.
37. Andrea Lancichinetti, Filippo Radicchi, and José J Ramasco. Statistical significance of communities in networks. *Physical Review E*, 81(4):046110, 2010.
38. E. Lima, C. Mues, and B. Baesens. Domain knowledge integration in data mining using decision tables: case studies in churn prediction. *Journal of the Operational Research Society*, 60(8):1096–1106, Aug 2009.
39. Dianbo Liu and Luca Albergante. Balance of thrones: a network study on 'Game of Thrones'. *arXiv preprint arXiv:1707.05213*, 2017.
40. Jian Liu and Tingzhan Liu. Detecting community structure in complex networks using simulated annealing with k-means algorithms. *Physica A: Statistical Mechanics and its Applications*, 389(11):2300–2309, 2010.
41. George R.R. Martin. *A Song of Ice and Fire, Book Three: A Storm of Swords*. HarperCollins Publishers, New York, NY, 2011.
42. Ananth Mohan, Zheng Chen, and Kilian Weinberger. Web-search ranking with initialized gradient boosted regression trees. In *Proceedings of the 2010 International Conference on Yahoo! Learning to Rank Challenge - Volume 14*, YLRC'10, pages 77–89. JMLR.org, 2010.
43. Stephen Rodriguez and Heechang Shin. Developing customer churn models for customer relationship management. *International Journal of Business Continuity and Risk Management*, 4(4):302–322, 2013.
44. Stephanie Rosenthal, Manuela M. Veloso, and Anind K. Dey. Online selection of mediated and domain-specific predictions for improved recommender systems. In Sarabjot S. Anand, Bamshad Mobasher, Alfred Kobsa, and Dietmar Jannach, editors, *Proceedings of the 7th Workshop on Intelligent Techniques for Web Personalization & Recommender Systems (ITWP'09), Pasadena, California, USA, July 11–17, 2009 in conjunction with the 21st International Joint Conference on Artificial Intelligence - IJCAI 2009*, volume 528 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2009.

45. Sebastian Schelter, Christoph Boden, and Volker Markl. Scalable similarity-based neighborhood methods with MapReduce. In *Proceedings of the Sixth ACM Conference on Recommender Systems*, RecSys '12, pages 163–170, New York, NY, USA, 2012. ACM.
46. Sebastian Schelter, Christoph Boden, Martin Schenck, Alexander Alexandrov, and Volker Markl. Distributed matrix factorization with MapReduce using a series of broadcast-joins. In *Proceedings of the 7th ACM Conference on Recommender Systems*, RecSys '13, pages 281–284, New York, NY, USA, 2013. ACM.
47. Sebastian Schelter, Stephan Ewen, Kostas Tzoumas, and Volker Markl. "all roads lead to Rome": Optimistic recovery for distributed iterative data processing. In *Proceedings of the 22Nd ACM International Conference on Information & Knowledge Management*, CIKM '13, pages 1919–1928, New York, NY, USA, 2013. ACM.
48. Anuj Sharma, Dr Panigrahi, and Prabin Kumar. A neural network based approach for predicting customer churn in cellular network services. *arXiv preprint arXiv:1309.3945*, 2013.
49. Wouter Verbeke, David Martens, Christophe Mues, and Bart Baesens. Building comprehensible customer churn prediction models with advanced rule induction techniques. *Expert Systems with Applications*, 38(3):2354–2364, 2011.
50. Zhixiang Xu, Matt Kusner, Gao Huang, and Kilian Weinberger. Anytime representation learning. In Sanjoy Dasgupta and David McAllester, editors, *Proceedings of the 30th International Conference on Machine Learning*, volume 28 of *Proceedings of Machine Learning Research*, pages 1076–1084, Atlanta, Georgia, USA, 17–19 Jun 2013. PMLR.
51. Zhixiang Xu, Matt J. Kusner, Kilian Q. Weinberger, and Minmin Chen. Cost-sensitive tree of classifiers. In *Proceedings of the 30th International Conference on International Conference on Machine Learning - Volume 28*, ICML'13, pages I–133–I–141. JMLR.org, 2013.
52. Zhixiang Eddie Xu, Matt J. Kusner, Kilian Q. Weinberger, Minmin Chen, and Olivier Chapelle. Classifier cascades and trees for minimizing feature evaluation cost. *Journal of Machine Learning Research*, 15(1):2113–2144, 2014.
53. Zhixiang Eddie Xu, Kilian Q. Weinberger, and Olivier Chapelle. The greedy miser: Learning under test-time budgets. In *Proceedings of the 29th International Conference on Machine Learning, ICML 2012, Edinburgh, Scotland, UK, June 26 - July 1, 2012*. icml.cc / Omnipress, 2012.
54. Mümin Yildiz and Songül Albayrak. Customer churn prediction in telecommunication with rotation forest method. In *DBKDA 2017, The Ninth International Conference on Advances in Databases, Knowledge, and Data Applications*, pages 26–29, 2017.

Index