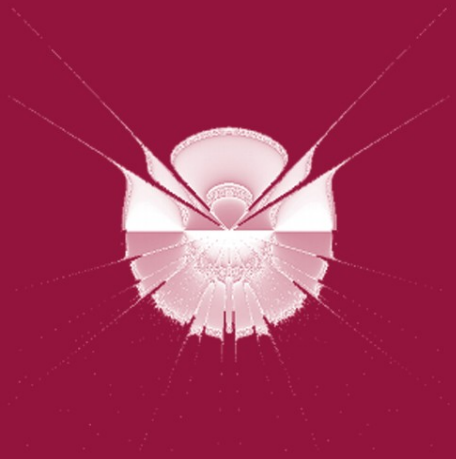
Cecilia Di Chio et al. (Eds.)

Applications of Evolutionary Computation

LNCS 6025

EvoApplications 2010: EvoCOMNET, EvoENVIRONMENT, EvoFIN, EvoMUSART, and EvoTRANSLOG
Istanbul, Turkey, April 2010, Proceedings, Part II



Springer

Lecture Notes in Computer Science 6025

Commenced Publication in 1973
Founding and Former Series Editors:
Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

Cecilia Di Chio Anthony Brabazon
Gianni A. Di Caro Marc Ebner
Muddassar Farooq Andreas Fink
Jörn Grahl Gary Greenfield
Penousal Machado Michael O'Neill
Ernesto Tarantino Neil Urquhart (Eds.)

Applications of Evolutionary Computation

EvoApplications 2010: EvoCOMNET, EvoENVIRONMENT, EvoFIN, EvoMUSART, and EvoTRANSLOG
Istanbul, Turkey, April 7-9, 2010
Proceedings, Part II

Springer

Volume Editors

see next page

Cover illustration:
"Pelegrina Galathea" by Stayko Chalakov (2009) Aston University, UK

Library of Congress Control Number: 2010923234

CR Subject Classification (1998): D.2, C.2, H.4, C.2.4, D.4, D.1.3

LNCS Sublibrary: SL 1 – Theoretical Computer Science and General Issues

ISSN 0302-9743
ISBN-10 3-642-12241-8 Springer Berlin Heidelberg New York
ISBN-13 978-3-642-12241-5 Springer Berlin Heidelberg New York

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper 06/3180

Volume Editors

Cecilia Di Chio
Dept. of Mathematics and Statistics
University of Strathclyde, UK
cecilia@stams.strath.ac.uk

Anthony Brabazon
School of Business
University College Dublin, Ireland
anthony.brabazon@ucd.ie

Gianni A. Di Caro
"Dalle Molle" Institute
for Artificial Intelligence (IDSIA)
Lugano, Switzerland
gianni@idsia.ch

Marc Ebner
Wilhelm-Schickard-Institut
für Informatik
Universität Tübingen, Germany
marc.ebner@wsii.uni-tuebingen.de

Muddassar Farooq
National University of Computer
and Emerging Sciences
Islamabad, Pakistan
muddassar.farooq@nu.edu.pk

Andreas Fink
Fac. of Economics & Social Sciences
Helmut-Schmidt-University
Hamburg, Germany
andreas.fink@hsu-hamburg.de

Jörn Grahl
Department of Information Systems
Johannes Gutenberg-University
Mainz, Germany
grahl@uni-mainz.de

Gary Greenfield
Mathematics & Computer Science
Department
University of Richmond, USA
ggreenfi@richmond.edu

Penousal Machado
Department of Informatics Engineering
University of Coimbra, Portugal
Machado@dei.uc.pt

Michael O'Neill
School of Computer Science
and Informatics
University College Dublin, Ireland
m.oneill@ucd.ie

Ernesto Tarantino
Institute for High Performance
Computing and Networking
ICNAR-CNR, Naples, Italy
ernesto.tarantino@na.icar.cnr.it

Neil Urquhart
School of Computing
Edinburgh Napier University, UK
n.urquhart@napier.ac.uk

Preface

Evolutionary computation (EC) techniques are efficient, nature-inspired methods based on the principles of natural evolution and genetics. Due to their efficiency and simple underlying principles, these methods can be used for a diverse range of activities including problem solving, optimization, machine learning and pattern recognition. A large and continuously increasing number of researchers and professionals make use of EC techniques in various application domains. This volume presents a careful selection of relevant EC examples combined with a thorough examination of the techniques used in EC. The papers in the volume illustrate the current state of the art in the application of EC and should help and inspire researchers and professionals to develop efficient EC methods for design and problem solving.

All papers in this book were presented during EvoApplications 2010, which included a range of events on application-oriented aspects of EC. Since 1998, EvoApplications — formerly known as EvoWorkshops — has provided a unique opportunity for EC researchers to meet and discuss application aspects of EC and has been an important link between EC research and its application in a variety of domains. During these 12 years, new events have arisen, some have disappeared, while others have matured to become conferences of their own, such as EuroGP in 2000, EvoCOP in 2004, and EvoBIO in 2007. And from this year, EvoApplications has become a conference as well.

EvoApplications is part of EVO*, Europe's premier co-located events in the field of evolutionary computing. EVO* was held from the 7th to the 9th of April 2010 in the beautiful city of Istanbul, Turkey, which was European City of Culture in 2010. Evo* 2010 included, in addition to EvoApplications, EuroGP, the main European event dedicated to genetic programming; EvoCOP, the main European conference on EC in combinatorial optimization; EvoBIO, the main European conference on EC and related techniques in bioinformatics and computational biology. The proceedings for all of these events, EuroGP 2010, EvoCOP 2010 and EvoBIO 2010, are also available in the LNCS series (volumes 6021, 6022, and 6023).

Moreover, thanks to the large number of submissions received, the proceedings for EvoApplications 2010 are divided across two volumes. The present volume, which contains contributions for: EvoCOMNET, EvoENVIRONMENT, EvoFIN, EvoMUSART, and EvoTRANSLOG; and volume one (LNCS 6024), which contains contributions for: EvoCOMPLEX, EvoGAMES, EvoIASP, EvoINTELLIGENCE, EvoNUM, and EvoSTOC.

The central aim of the EVO* events is to provide researchers, as well as people from industry, students, and interested newcomers, with an opportunity to present new results, discuss current developments and applications, or just become acquainted with the world of EC. Moreover, it encourages and reinforces

possible synergies and interactions between members of all scientific communities that may benefit from EC techniques.

EvoApplications 2010 consisted of the following individual events:

- *EvoCOMNET*, the 7^{th} European Event on the Application of Nature-Inspired Techniques for Telecommunication Networks and other Parallel and Distributed Systems
- *EvoCOMPLEX*, the 1^{st} European Event on Evolutionary Algorithms and Complex Systems
- *EvoENVIRONMENT*, the 2^{nd} European Event on Nature-Inspired Methods for Environmental Issues
- *EvoFIN*, the 4^{th} European Event on Evolutionary and Natural Computation in Finance and Economics
- *EvoGAMES*, the 2^{nd} European Event on Bio-inspired Algorithms in Games
- *EvoIASP*, the 12^{th} European Event on Evolutionary Computation in Image Analysis and Signal Processing
- *EvoINTELLIGENCE*, the 1^{st} European Event on Nature-Inspired Methods for Intelligent Systems
- *EvoMUSART*, the 8^{th} European Event on Evolutionary and Biologically Inspired Music, Sound, Art and Design
- *EvoNUM*, the 3^{rd} European Event on Bio-inspired Algorithms for Continuous Parameter Optimization
- *EvoSTOC*, the 7^{th} European Event on Evolutionary Algorithms in Stochastic and Dynamic Environments
- *EvoTRANSLOG*, the 4^{th} European Event on Evolutionary Computation in Transportation and Logistics

EvoCOMNET addresses the application of EC techniques to problems in distributed and connected systems such as telecommunication and computer networks, distribution and logistic networks, interpersonal and interorganizational networks, etc. To address these challenges, this event promotes the study and the application of strategies inspired by the observation of biological and evolutionary processes, that usually show the highly desirable characteristics of being distributed, adaptive, scalable, and robust.

EvoCOMPLEX covers all aspects of the interaction of evolutionary algorithms (and metaheuristics in general) with complex systems. Complex systems are ubiquitous in physics, economics, sociology, biology, computer science, and many other scientific areas. Typically, a complex system is composed of smaller aggregated components, whose interaction and interconnectedness are non-trivial. This leads to emergent properties of the system, not anticipated by its isolated components. Furthermore, when the system behavior is studied from a temporal perspective, self-organization patterns typically arise.

EvoENVIRONMENT is devoted to the use of nature-inspired methods for environmental issues. It deals with many diverse topics such as waste management, sewage treatment, control of greenhouse gas emissions, biodegradation of materials, efficient energy use, or use of renewable energies, to name but a few.

EvoFIN is the only European event specifically dedicated to the applications of EC, and related natural computing methodologies, to finance and economics. Financial environments are typically hard, being dynamic, high-dimensional, noisy and co-evolutionary. These environments serve as an interesting test bed for novel evolutionary methodologies.

EvoGAMES aims to focus the scientific developments onto computational intelligence techniques that may be of practical value for utilization in existing or future games. Recently, games, and especially video games, have become an important commercial factor within the software industry, providing an excellent test bed for the application of a wide range of computational intelligence methods.

EvoIASP, the longest-running of all EvoApplications which celebrated its 12th edition this year, has been the first international event solely dedicated to the applications of EC to image analysis and signal processing in complex domains of high industrial and social relevance.

EvoINTELLIGENCE is devoted to the use of nature-inspired methods to create all kinds of intelligent systems. The scope of the event includes evolutionary robotics, artificial life and related areas. Intelligent systems do not necessarily have to exhibit human or animal-like intelligence. Intelligent behavior can also be found in everyday devices such as a digital video recorder or handheld devices such as an MP3 player which learn from the human who is operating the device.

EvoMUSART addresses all practitioners interested in the use of EC techniques for the development of creative systems. There is a growing interest in the application of these techniques in fields such as art, music, architecture and design. The goal of this event is to bring together researchers that use EC in this context, providing an opportunity to promote, present and discuss the latest work in the area, fostering its further developments and collaboration among researchers.

EvoNUM aims at applications of bio-inspired algorithms, and cross-fertilization between these and more classical numerical optimization algorithms, to continuous optimization problems in engineering. It deals with theoretical aspects and engineering applications where continuous parameters or functions have to be optimized, in fields such as control, chemistry, agriculture, electricity, building and construction, energy, aerospace engineering, design optimization.

EvoSTOC addresses the application of EC in stochastic and dynamic environments. This includes optimization problems with changing, noisy, and/or approximated fitness functions and optimization problems that require robust solutions. These topics recently gained increasing attention in the EC community and EvoSTOC was the first event that provided a platform to present and discuss the latest research in this field.

EvoTRANSLOG deals with all aspects of the use of evolutionary computation, local search and other nature-inspired optimization and design techniques for the transportation and logistics domain. The impact of these problems on the modern economy and society has been growing steadily over the last few decades, and the event aims at design and optimization techniques such as

evolutionary computing approaches allowing the use of computer systems for systematic design, optimization, and improvement of systems in the transportation and logistics domain.

Continuing in the tradition of adapting the list of the events to the needs and demands of the researchers working in the field of evolutionary computing, EvoINTERACTION, the European Event on Interactive Evolution and Humanized Computational Intelligence, and EvoHOT, the European Event on Bio-inspired Heuristics for Design Automation, decided not to run in 2010 and will run again in 2011. Two new events were also proposed this year: EvoCOMPLEX, the First European Event on Evolutionary Algorithms and Complex Systems, and EvoINTELLIGENCE, the First European Event on Nature-Inspired Methods for Intelligent Systems.

The number of submissions to EvoApplications 2010 was once again very high, cumulating 188 entries (with respect to 133 in 2008 and 143 in 2009). The following table shows relevant statistics for EvoApplications 2010 (both short and long papers are considered in the acceptance statistics), compared with those from the 2009 edition:

Event	2010			2009		
	Submissions	Accept	Ratio	Submissions	Accept	Ratio
EvoCOMNET	17	12	71%	21	15	71%
EvoCOMPLEX	12	6	50%	-	-	-
EvoENVIRONMENT	5	4	80%	5	4	80%
EvoFIN	17	10	59%	14	8	57%
EvoGAMES	25	15	60%	15	10	67%
EvoIASP	24	15	62%	14	7	50%
EvoINTELLIGENCE	8	5	62%	-	-	-
EvoMUSART	36	16	44%	26	17	65%
EvoNUM	25	15	60%	16	9	56%
EvoSTOC	11	6	54%	11	7	64%
EvoTRANSLOG	11	5	45%	11	6	54%
Total	191	109	57%	143	91	64%

As for previous years, accepted papers were split into oral presentations and posters. However, this year, each event made their own decision on paper length for these two categories. Hence, for some events, papers in both categories are of the same length. The acceptance rate of 57.1% for EvoApplications 2010, along with the significant number of submissions, is an indicator of the high quality of the articles presented at the events, showing the liveliness of the scientific movement in the corresponding fields.

Many people have helped make EvoApplications a success. We would like to thank the following institutions:

- Computer Engineering Department of Istanbul Technical University, Turkey, for supporting the local organization
- Istanbul Technical University, Microsoft Turkey, and the Scientific and Technological Research Council of Turkey, for their patronage of the event

– Centre for Emergent Computing at Edinburgh Napier University, Scotland, for administrative help and event coordination

We want to especially acknowledge our invited speakers: Kevin Warwick (University of Reading, UK), Luigi Luca Cavalli-Sforza (Stanford School of Medicine, USA); and Günther Raidl (Vienna University of Technology, Austria) and Jens Gottlieb (SAP, Walldorf, Germany) for their special EvoCOP 10th anniversary talk.

We are also very grateful to all the people who provided local support, in particular Sanem Sarıel-Talay, Şule Gündüz-Öğüdücü, Ayşegül Yayımlı, Gülşen Cebiroğlu-Eryiğit, and H. Turgut Uyar.

Even with an excellent support and location, an event like EVO* would not have been feasible without authors submitting their work, members of the Program Committees dedicating their energy in reviewing those papers, and an audience. All these people deserve our gratitude.

Finally, we are grateful to all those involved in the preparation of the event, especially Jennifer Willies for her unfaltering dedication to the coordination of the event over the years. Without her support, running such a type of conference with a large number of different organizers and different opinions would be unmanageable. Further thanks to the local organizer A. Şima (Etaner) Uyar for making the organization of such an event possible and successful. Last but surely not least, we want to specially acknowledge Stephen Dignum for his hard work as Publicity Chair of the event, and Marc Schoenauer for his continuous help in setting up and maintaining the MyReview management software.

April 2010

Cecilia Di Chio
Anthony Brabazon
Gianni Di Caro
Marc Ebner
Muddassar Farooq
Andreas Fink
Jörn Grahl
Gary Greenfield
Penousal Machado
Michael O'Neill
Ernesto Tarantino
Neil Urquhart

Organization

EvoApplications 2010 was part of EVO* 2010, Europe's premier co-located events in the field of evolutionary computing, that also included the conferences EuroGP 2010, EvoCOP 2010, and EvoBIO 2010.

Organizing Committee

EvoApplications Chair:	Cecilia Di Chio, University of Strathclyde, UK
Local Chairs:	A. Şima (Etaner) Uyar, Istanbul Technical University, Turkey
Publicity Chair:	Stephen Dignum, University of Essex, UK
EvoCOMNET Co-chairs:	Gianni A. Di Caro, IDSIA, Switzerland Muddassar Farooq, National University of Computer and Emerging Sciences, Pakistan Ernesto Tarantino, Institute for High Performance Computing and Networking, Italy
EvoCOMPLEX Co-chairs:	Carlos Cotta, University of Malaga, Spain Juan J. Merelo, University of Granada, Spain
EvoENVIRONMENT Co-chairs:	Marc Ebner, University of Tübingen, Germany Neil Urquhart, Edinburgh Napier University, UK
EvoFIN Co-chairs:	Anthony Brabazon, University College Dublin, Ireland Michael O'Neill, University College Dublin, Ireland
EvoGAMES Co-chairs:	Mike Preuss, TU Dortmund University, Germany Julian Togelius, IT University of Copenhagen, Denmark Georgios N. Yannakakis, IT University of Copenhagen, Denmark
EvoIASP Chair:	Stefano Cagnoni, University of Parma, Italy

EvoINTELLIGENCE Co-chairs:	Marc Ebner, University of Tübingen, Germany Cecilia Di Chio, University of Strathclyde, UK
EvoMUSART Co-chairs:	Penousal Machado, University of Coimbra, Portugal Gary Greenfield, University of Richmond, USA
EvoNUM Co-chairs:	Anna Isabel Esparcia-Alcazar, ITI-Universidad Politécnica de Valencia, Spain Anikó Ekárt, Aston University, UK
EvoSTOC Co-chairs:	Ferrante Neri, University of Jyväskylä, Finland Chi-Keong Goh, Advanced Technology Centre Rolls-Royce, Singapore
EvoTRANSLOG Co-chairs:	Andreas Fink, Helmut-Schmidt-University Hamburg, Germany Jörn Grahl, Johannes Gutenberg University, Germany

Program Committees

EvoCOMNET Program Committee

Özgür B. Akan	Middle East Technical University, Turkey
Enrique Alba	University of Malaga, Spain
Qing Anyong	National University of Singapore, Singapore
Payman Arabshahi	University of Washington, USA
Mehmet E. Aydin	University of Bedfordshire, UK
Iacopo Carreras	CREATE-NET, Italy
Arindam K. Das	University of Washington, USA
Falko Dressler	University of Erlangen, Germany
Frederick Ducatelle	IDSIA, Switzerland
Luca Gambardella	IDSIA, Switzerland
Jin-Kao Hao	University of Angers, France
Malcolm I. Heywood	Dalhousie University, Canada
Byrant Julstrom	St. Cloud State University, USA
Graham Kendall	University of Nottingham, UK
Kenji Leibnitz	Osaka University, Japan
Manuel Lozano-Marquez	University of Granada, Spain
Domenico Maisto	ICAR CNR, Italy
Ronaldo Menezes	Florida Institute of Technology, USA
Martin Middendorf	University of Leipzig, Germany
Roberto Montemanni	IDSIA, Switzerland
Chien-Chung Shen	University of Delaware, USA

Tony White	Carleton University, Canada
Lidia Yamamoto	University of Basel, Switzerland
Nur Zincir-Heywood	Dalhousie University, Canada

EvoCOMPLEX Program Committee

Antonio Córdoba	Universidad de Sevilla, Spain
Carlos Cotta	Universidad de Málaga, Spain
Jordi Delgado	Universitat Politècnica de Catalunya, Spain
Carlos Gershenson	UNAM, Mexico
Mario Giacobini	Università di Torino, Italy
Anca Gog	Babes-Bolyai University, Romania
Márk Jelasity	University of Szeged, Hungary
Juan Luis Jiménez	University of Granada, Spain
Jose Fernando Mendes	Universidade de Aveiro, Portugal
Juan J. Merelo	Universidad de Granada, Spain
Joshua L. Payne	University of Vermont, USA
Mike Preuss	Universität Dortmund, Germany
Katya Rodríguez-Vázquez	UNAM, Mexico
Kepa Ruiz-Mirazo	Euskal Herriko Unibertsitatea, Spain
Luciano Sánchez	Universidad de Oviedo, Spain
Robert Schaefer	AGH University of Science and Technology, Poland
Marco Tomassini	Université de Lausanne, Switzerland
Fernando Tricas	Universidad de Zaragoza, Spain
Sergi Valverde	Universitat Pompeu Frabra, Spain
Leonardo Vanneschi	University of Milano-Bicocca, Italy

EvoENVIRONMENT Program Committee

Stefano Cagnoni	University of Parma, Italy
Pierre Collet	Université de Strasbourg, France
Kevin Cullinane	Edinburgh Napier University, UK
Marc Ebner	Universität Tübingen, Germany
James A Foster	University of Idaho, USA
Nanlin Jin	University of Leeds, UK
Rhyd Lewis	Cardiff University, UK
William Magette	University College Dublin, Ireland
R I (Bob) McKay	Seoul National University, Korea
Michael O'Neill	University College Dublin, Ireland
Stefano Pizzuti	Energy New Tech. and Environment Agency, Italy
Tom Rye	Edinburgh Napier University, UK
Carlo Santulli	University of Rome "La Sapienza", Italy
Marc Schoenauer	INRIA, France
Terence Soule	University of Idaho, USA
John Summerscales	University of Plymouth, UK

Neil Urquhart	Edinburgh Napier University, UK
Tina Yu	Memorial University of Newfoundland, Canada
Mengjie Zhang	University of Wellington, New Zealand

EvoFIN Program Committee

Eva Alfaro-Cid	Instituto Tecnológico de Informática, Spain
Antonia Azzini	Università degli Studi di Milano, Italy
Anthony Brabazon	University College Dublin, Ireland
Louis Charbonneau	Concordia University, Canada
Gregory Connor	National University of Ireland Maynooth, Ireland
Ian Dempsey	Pipeline Trading, USA
Rafal Drezewski	AGH University of Science and Technology, Poland
Manfred Gilli	University of Geneva and Swiss Finance Institute, Switzerland
Philip Hamill	University of Ulster, UK
Ronald Hochreiter	WU Vienna University of Economics and Business, Austria
Youwei Li	Queen's University Belfast, UK
Dietmar Maringer	University of Basel, Switzerland
Michael O'Neill	University College Dublin, Ireland
Philip Saks	University of Essex, UK
Robert Schafer	AGH University of Science and Technology, Poland
Andrea Tettamanzi	Università Degli Studi di Milano, Italy
Garnett Wilson	Memorial University of Newfoundland, Canada

EvoGAMES Program Committee

Lourdes Araujo	UNED, Spain
Wolfgang Banzhaf	Memorial University of Newfoundland, Canada
Luigi Barone	University of Western Australia, Australia
Simon Colton	Imperial College London, UK
Ernesto Costa	Universidade de Coimbra, Portugal
Carlos Cotta	Universidad de Málaga, Spain
Marc Ebner	University of Tübingen, Germany
Anikó Ekárt	Aston University, UK
Anna Esparcia Alcázar	Instituto Tecnológico de Informática, Spain
Francisco Fernández	Universidad de Extremadura, Spain
Antonio J Fernández Leiva	Universidad de Málaga, Spain
Mario Giacobini	Università degli Studi di Torino, Italy
Johan Hagelbäck	Blekinge Tekniska Högskola, Sweden
John Hallam	University of Southern Denmark, Denmark
David Hart	Fall Line Studio, USA

Philip Hingston	Edith Cowan University, Australia
Stefan Johansson	Blekinge Tekniska Högskola, Sweden
Rilla Khaled	IT University of Copenhagen, Denmark
Elias Kosmatopoulos	Dimocritian University of Thrace, Greece
Krzysztof Krawiec	Poznan University of Technology, Poland
Pier Luca Lanzi	Politecnico di Milano, Italy
Simon Lucas	University of Essex, UK
Penousal Machado	Universidade de Coimbra, Portugal
Juan J. Merelo	Universidad de Granada, Spain
Risto Miikkulainen	University of Texas at Austin, USA
Antonio Mora	Universidad de Granada, Spain
Mike Preuss	Universität Dortmund, Germany
Steffen Priesterjahn	University of Paderborn, Germany
Moshe Sipper	Ben-Gurion University, Israel
Terence Soule	University of Idaho, USA
Julian Togelius	IT University of Copenhagen, Denmark
Georgios N. Yannakakis	IT University of Copenhagen, Denmark

EvoIASP Program Committee

Antonia Azzini	University of Milan-Crema, Italy
Lucia Ballerini	University of Edinburgh, UK
Leonardo Bocchi	University of Florence, Italy
Stefano Cagnoni	University of Parma, Italy
Oscar Cordon	European Center for Soft Computing, Spain
Sergio Damas	European Center for Soft Computing, Spain
Ivanoe De Falco	ICAR - CNR, Italy
Antonio Della Cioppa	University of Salerno, Italy
Laura Dipietro	MIT, USA
Marc Ebner	University of Tübingen, Germany
Francesco Fontanella	University of Cassino, Italy
Špela Ivekoviç	University of Dundee, UK
Mario Koeppen	Kyushu Institute of Technology, Japan
Krisztof Krawiec	Poznan University of Technology, Poland
Jean Louchet	INRIA, France
Evelyne Lutton	INRIA, France
Luca Mussi	University of Parma, Italy
Ferrante Neri	University of Jyväskylä, Finland
Gustavo Olague	CICESE, Mexico
Riccardo Poli	University of Essex, UK
Stephen Smith	University of York, UK
Giovanni Squillero	Politecnico di Torino, Italy
Kiyoshi Tanaka	Shinshu University, Japan
Andy Tyrrell	University of York, UK
Leonardo Vanneschi	University of Milan Bicocca, Italy
Mengjie Zhang	Victoria University of Wellington, New Zealand

EvoINTELLIGENCE Program Committee

Wolfgang Banzhaf	Memorial University of Newfoundland, Canada
Peter Bentley	University College London, UK
Stefano Cagnoni	University of Parma, Italy
Cecilia Di Chio	University of Strathclyde, UK
Marc Ebner	Eberhard Karls Universität Tübingen, Germany
Mario Giacobini	University of Turin, Italy
Greg Hornby	University of California Santa Cruz, USA
Christian Jacob	University of Calgary, Canada
Gul Muhammad Kahn	University of Engineering and Technology, Pakistan
Gabriela Kokai	Fraunhofer Inst. für Integrated Circuits, Germany
William B. Langdon	King's College, London, UK
Penousal Machado	University of Coimbra, Portugal
Julian Miller	University of York, UK
Gustavo Olague	CICESE, Mexico
Michael O'Neill	University College Dublin, Ireland
Thomas Ray	University of Oklahoma, USA
Marc Schoenauer	INRIA, France
Moshe Sipper	Ben-Gurion University, Israel
Ivan Tanev	Doshisha University, Japan
Mengjie Zhang	Victoria University of Wellington, New Zealand

EvoMUSART Program Committee

Mauro Annunziato	Plancton Art Studio, Italy
Peter Bentley	University College London, UK
Eleonora Bilotta	University of Calabria, Italy
Tim Blackwell	Goldsmiths College, University of London, UK
Simon Colton	Imperial College, UK
Oliver Bown	Monash University, Australia
Paul Brown	University of Sussex, UK
Stefano Cagnoni	University of Parma, Italy
Amilcar Cardoso	University of Coimbra, Portugal
Vic Ciesielski	RMIT, Australia
Palle Dahlstedt	Göteborg University, Sweden
Hans Dehlinger	Independent Artist, Germany
Steve DiPaola	Simon Fraser University, Canada
Alan Dorin	Monash University, Australia
Erwin Driessens	Independent Artist, The Netherlands
Philip Galanter	Texas A&M College of Architecture, USA
Pablo Gervás	Universidad Complutense de Madrid, Spain
Andrew Gildfind	Google, Inc., Australia
Carlos Grilo	Instituto Politécnico de Leiria, Portugal

David Hart	Independent Artist, USA
Amy K. Hoover	University of Central Florida, USA
Andrew Horner	University of Science & Technology, Hong Kong
Christian Jacob	University of Calgary, Canada
Colin Johnson	University of Kent, UK
Craig Kaplan	University of Waterloo, Canada
Matthew Lewis	Ohio State University, USA
Alain Lioret	Paris 8 University, France
Bill Manaris	College of Charleston, USA
Ruli Manurung	University of Indonesia, Indonesia
Jonatas Manzolli	UNICAMP, Brazil
Jon McCormack	Monash University, Australia
James McDermott	University of Limerick, Ireland
Eduardo Miranda	University of Plymouth, UK
Nicolas Monmarché	University of Tours, France
Gary Nelson	Oberlin College, USA
Luigi Pagliarini	PEAM, Italy & University of Southern, Denmark
Rui Pedro Paiva	University of Coimbra, Portugal
Alejandro Pazos	University of A Coruna, Spain
Somnuk Phon-Amnuaisuk	Multimedia University, Malaysia
Rafael Ramirez	Pompeu Fabra University, Spain
Juan Romero	University of A Coruna, Spain
Brian Ross	Brock University, Canada
Artemis Sanchez Moroni	Renato Archer Research Center, Brazil
Antonino Santos	University of A Coruna, Spain
Kenneth O. Stanley	University of Central Florida, USA
Jorge Tavares	University of Coimbra, Portugal
Stephen Todd	IBM, UK
Paulo Urbano	Universidade de Lisboa, Portugal
Anna Ursyn	University of Northern Colorado, USA
Maria Verstappen	Independent Artist, The Netherlands
Gerhard Widmer	Johannes Kepler University Linz, Austria

EvoNUM Program Committee

Eva Alfaro-Cid	ITI – Universidad Politécnica de Valencia, Spain
Anne Auger	INRIA, France
Wolfgang Banzhaf	Memorial University of Newfoundland, Canada
Xavier Blasco	Universidad Politécnica de Valencia, Spain
Hans-Georg Beyer	Vorarlberg University of Applied Sciences, Austria
Ying-ping Chen	National Chiao Tung University, Taiwan
Carlos Cotta	Universidad de Malaga, Spain
Marc Ebner	Universität Würzburg, Germany

Gusz Eiben	Vrije Universiteit Amsterdam, The Netherlands
Şima Etaner-Uyar	Istanbul Technical University, Turkey
Francisco Fernández de Vega	Universidad de Extremadura, Spain
Nikolaus Hansen	INRIA, France
José Ignacio Hidalgo	Universidad Complutense de Madrid, Spain
Andras Joo	Aston University, UK
Bill Langdon	King's College London, UK
Juan J. Merelo	Universidad de Granada, Spain
Boris Naujoks	Log!n GmbH, Germany
Ferrante Neri	University of Jyväskylä, Finland
Gabriela Ochoa	University of Nottingham, UK
Petr Pošík	Czech Technical University, Czech Republic
Mike Preuss	University of Dortmund, Germany
Günter Rudolph	University of Dortmund, Germany
Marc Schoenauer	INRIA, France
Hans-Paul Schwefel	University of Dortmund, Germany
P.N. Suganthan	Nanyang Technological University, Singapore
Ke Tang	University of Science and Technology of China, China
Olivier Teytaud	INRIA, France
Darrell Whitley	Colorado State University, USA

EvoSTOC Program Committee

Hussein Abbass	University of New South Wales, Australia
Dirk Arnold	Dalhousie University, Canada
Hans-Georg Beyer	Vorarlberg University of Applied Sciences, Austria
Peter Bosman	Centre for Mathematics and Computer Science, The Netherlands
Juergen Branke	University of Karlsruhe, Germany
Andrea Caponio	Technical University of Bari, Italy
Ernesto Costa	University of Coimbra, Portugal
Kalyanmoy Deb	Indian Institute of Technology Kanpur, India
Andries Engelbrecht	University of Pretoria, South Africa
Yaochu Jin	Honda Research Institute Europe, Germany
Anna V. Kononova	University of Leeds, UK
Jouni Lampinen	University of Vaasa, Finland
Xiaodong Li	RMIT University, Australia
John McCall	Robert Gordon University, UK
Ernesto Mininno	University of Jyväskylä, Finland
Yew Soon Ong	Nanyang Technological University of Singapore, Singapore
Zhang Qingfu	University of Essex, UK
William Rand	University of Maryland, USA

Khaled Rasheed	University of Georgia, USA
Hendrik Richter	University of Leipzig, Germany
Philipp Rohlfshagen	University of Birmingham, UK
Kay Chen Tan	National University of Singapore, Singapore
Ke Tang	University of Science and Technology of China, China
Yoel Tenne	Sydney University, Australia
Renato Tinos	Universidade de Sao Paulo, Brazil
Ville Tirronen	University of Jyväskylä, Finland
Shengxiang Yang	University of Leicester, UK
Gary Yen	Oklahoma State University, USA

EvoTRANSLOG Program Committee

Christian Blum	Univ. Politecnica Catalunya, Spain
Peter A.N. Bosman	Centre for Mathematics and Computer Science, The Netherlands
Marco Caserta	University of Hamburg, Germany
Loukas Dimitriou	National Technical University of Athens, Greece
Karl Doerner	University of Vienna, Austria
Martin Josef Geiger	Helmut-Schmidt-University Hamburg, Germany
Stefan Irnich	RWTH Aachen University, Germany
Hoong Chuin Lau	Singapore Management University, Singapore
Christian Prins	University of Technology of Troyes, France
Franz Rothlauf	University of Mainz, Germany
Kay Chen Tan	National University of Singapore, Singapore
Theodore Tsekeris	Center of Planning and Economic Research, Greece
Stefan Voß	University of Hamburg, Germany
Oliver Wendt	University of Kaiserslautern, Germany

Sponsoring Institutions

- Istanbul Technical University, Istanbul, Turkey
- Microsoft Turkey
- Scientific and Technological Research Council of Turkey
- The Centre for Emergent Computing at Edinburgh Napier University, Scotland

Table of Contents – Part II

EvoCOMNET Contributions

EvoENVIRONMENT Contributions

EvoFIN Contributions

EvoMUSART Contributions

EvoTRANSLOG Contributions

Table of Contents – Part I

EvoCOMPLEX Contributions

EvoGAMES Contributions

EvoIASP Contributions

EvoINTELLIGENCE Contributions

EvoNUM Contributions

EvoSTOC Contributions

Detection of DDoS Attacks via an Artificial Immune System-Inspired Multiobjective Evolutionary Algorithm

Uğur Akyazı[1] and A. Şima Uyar[2]

[1] Computer Engineering Department, Turkish Air Force Academy,
Yesilyurt, 34149, Istanbul, Turkey
u.akyazi@hho.edu.tr
[2] Computer Engineering Department, Istanbul Technical University,
Maslak, 34469, Istanbul, Turkey
etaner@itu.edu.tr

Abstract. A Distributed Denial of Service Attack is a coordinated attack on the availability of services of a victim system, launched indirectly through many compromised computers. Intrusion detection systems (IDS) are network security tools that process local audit data or monitor network traffic to search for specific patterns or certain deviations from expected behavior. We use an Artificial Immune System (AIS) as a method of anomaly-based IDS because of the similarity between the IDS architecture and the Biological Immune Systems. We improved the jREMISA study; a Multiobjective Evolutionary Algorithm inspired AIS, in order to get better true and false positive rates while detecting DDoS attacks on the MIT DARPA LLDOS 1.0 dataset. We added the method of r-continuous evaluations, changed the Negative Selection and Clonal Selection structure, and redefined the objectives while keeping the general concepts the same. The 100% true positive rate and 0% false positive rate of our approach, under the given parameter settings and experimental conditions, shows that it is very successful as an anomaly-based IDS for DDoS attacks.

Keywords: Intrusion Detection, Distributed Denial of Service Attack, DARPA LLDOS Dataset, Artificial Immune System, Multiobjective Evolutionary Algorithm.

1 Introduction

An intrusion detection system (IDS) is used to detect intrusions, which are actions that attempt to compromise the integrity, confidentiality or availability of a resource. Usually, an intruder first gains access to a single host by exploiting the software flaws, then tries to break-into other hosts in the network via the formerly compromised host, like Denial of Service (DoS) attacks. The objective of a DoS attack is to cause the target system to fail the services it normally provides. In a Distributed Denial of Service (DDoS) attack, one target is attacked simultaneously from a large number of sources. DDoS attacks often use the computers that have been previously exploited, so that an outsider can use them to launch an attack [1, 2, 3]. These zombie computers play their roles in the intermediate phase of the attack.

C. Di Chio et al. (Eds.): EvoApplications 2010, Part II, LNCS 6025, pp. 1–10, 2010.

We used the artificial immune system (AIS) as a method of anomaly-based intrusion detection because of the similarity between the IDS architecture and the biological immune system (BIS), which is a parallel and distributed adaptive system for detecting antigens. An AIS-based IDS classifies network traffic as either self or non-self by training a population of antigen detectors. jREMISA [4] is a multiobjective evolutionary algorithm (MOEA) inspired AIS has been developed previously. In this study, we enhanced jREMISA in order to get better true and false positive rates while detecting DDoS attacks on the MIT DARPA LLDOS 1.0 dataset.

2 Background

2.1 Intrusion Detection Systems (IDS)

The main objective of IDS is detecting wrong, unauthorized and malicious usage of computer systems by inside and outside intruders. The key is to maximize accurate alerts (true-positive) while at the same time minimizing the occurrence of non-justified alerts (false-positive). The metrics used in the evaluation of IDS are:

- True positive (TP) which is a real attack correctly categorized as an attack,
- False positive (FP) which is a false alert erroneously raised for normal data,
- True negative (TN) which is normal data that correctly does not generate an alert,
- False negative (FN) which is a missed attack erroneously categorized as normal.

IDS are classified into two groups as misuse detection and anomaly detection. In the misuse detection approach, network and system resources are examined in order to find known wrong usages by pattern matching techniques [5]. In anomaly detection systems, decisions are based on the normal network and system behaviors by using statistical or machine learning techniques to find both known and unknown attacks. A small deviation from normal behavior is detected as an intrusion [6, 7].

2.2 Distributed Denial of Service Attack (DDoS)

A DDoS attacker uses a large number of hosts to launch DoS attacks of SYN flooding, UDP flooding, and ICMP flooding against any target system. DDoS tools, like TFN, Trinoo, Stacheldraht, and Mstream install daemon programs on all of the compromised hosts which are controlled by a master program [2]. DDoS attacks can cause serious damages to Internet services. Tools to gain root access to other machines are freely available on the Internet [8].

2.3 Datasets

Releasing intrusion detection evaluation data is a problem because of privacy concerns. To overcome this problem, Lincoln Laboratory (LL), under sponsorship of Defense Advanced Research Projects Agency (DARPA), created the Intrusion Detection Evaluation Dataset (IDEVAL) that serves as a benchmark [9].

In 1998, 1999 and 2000, they built a network to simulate an Air Force base. They gathered tcpdump, Sun BSM, process and file system information after the background activities were produced with scripts, and attacks were injected at well defined points. More than 200 instances of 58 different attacks were embedded in the test data [10]. The first attack scenario example dataset to be created for DARPA in 2000 is LLDOS 1.0 which includes a DDoS attack.

In the LLDOS 1.0 scenario, the attacker uses the Solaris sadmind exploit to gain root access to three Solaris hosts of the simulated network and the Mstream DDOS tool to launch the attack. An Mstream "server" is installed on each of the three abused intermediate hosts, while an Mstream "master", which controls the "servers" is installed on one of these hosts. The DDoS attack is started by these "servers" simultaneously [11].

The attack scenario has five phases:

1. IPsweep of the network,
2. Probe of active hosts to look for the sadmind tool running on Solaris hosts,
3. Break-ins via the sadmind exploits,
4. Installation of the trojan mstream DDoS software on three hosts,
5. Launching the DDoS attack.

3 JREMISA (Java REtrovirus-inspired Multiobjective Immune System Algorithm)

In [4], an Artificial Immune System [12] is used together with Multiobjective Evolutionary Algorithms [13] in order to get good detectors with the best classifying fitness degree and multiobjective hypervolume size. Network traffic is classified as self and non-self with the help of antigen detectors which are trained using a dataset.

Multiobjective evolutionary algorithms (MOEA) are added to the AIS. A MOEA is preferred because it presents a set of trade-off solutions to the decision maker instead of one solution after evaluating the data for more than one objective. The objectives used in [4] are:

1. Minimization of the classification error rate which is obtained by adding the number of contradicting bits in true positive evaluations and adding the number of non-contradicting bits in true negative evaluations, since efficiency of the detector increases while total score of this objective decreases.
2. Minimization of the deviation from the negative selection affinity threshold. The scope of the detectors should not be too high to label the normal traffic as anomaly and too low to label the anomaly traffic as normal.

As a result, it is aimed to maximize the hypervolume of detectors which is the rectangular area covered by Pareto-front points and a reference point in the target space that shows the quality of the solutions.

3.1 Representation of Antigens and Antibodies

Antigen (Ag) and Antibody (Ab) chromosomes are binary arrays. Antigens are represented differently for three most common IP protocols of TCP, UDP and ICMP

traffic, where TCP Ag is coded using 240 bits, UDP Ag is coded using 170 bits and ICMP Ag is coded using 138 bits as representing all possible fields of IP, TCP, UDP and ICMP headers (IP=122, TCP= 118, UDP=48, ICMP=16). Decimal values of Ag packet header fields are transformed to equivalent binary values.

Ab chromosomes are composed of three parts as DNA (binary), RNA (binary), and seven state properties (integer). DNA bits are created during negative selection and is the only part that is evaluated against Ag chromosomes. RNA is a copy of DNA and is used to escape local optima. If the fitness value of the after-mutation DNA is better than before-mutation DNA, the new DNA is copied to RNA; otherwise RNA is copied to DNA in order to return the old gene values. There are seven characters at the end of the chromosome: λ=name, α=number of false detections, ρ= (true positive + true negative) fitness score, φ= (false positive + false negative) fitness score, η=deviation from affinity threshold, β=whether broadcasted on the network (yes/no), ψ=number of Ab's dominating to this Ab. Hamming distances are used as the affinity measure.

3.2 Immune Algorithm

Pseudocode of jREMISA is given Algorithm 1. Crossover is not applied since mutation is considered to be sufficient to make Ab's move in the objective search space, and not corrupt the good solutions. Through the 3-7 lines of the algorithm, the negative selection phase is implemented in which Ab's are created randomly and evaluated against all of the Ag's in a self-only dataset according to a pre-determined affinity threshold. The creation of this dataset is explained in detail in Section 5.1-Test Designs. If an Ab shows similarity to a self Ag, it is discarded without a replacement. The primary population is separated into three groups according to the IP protocols, so that a non-TCP Ab is not compared with a TCP Ab. Every Ag in the evaluation window represents a new generation and operations explained below are applied to all of the Ab's in the population:

- *Fitness function*: Hamming distance (H) which is defined as the similarity of the Ab and Ag DNA genes is calculated. One of the below cases will occur when the affinity threshold and the truth set are evaluated together:
  - True negative (Ag=self, Ab evaluates it as self): obj1+= H, copy DNA to RNA, obj2 += %1;
  - True positive (Ag= non-self , Ab evaluates it as non-self): obj1+= (Aglength – H), copy DNA to RNA, obj2+= %1;
  - False positive (Ag=self, Ab evaluates it as non-self): falseDetections++, copy RNA to DNA, obj2-= %1;
  - False negative (Ag= non-self , Ab evaluates it as self): falseDetections ++, copy RNA to DNA, obj2-= %1.

  First objective is penalized in true positives/negatives and second objective is penalized in false positives/negatives. H should be zero in an ideal true negative; otherwise the number of non-contradicting bits is added to the obj1 value. H should be equal to the length of Ag in an ideal true positive; otherwise the number of contradicting bits is added to obj1 value.

- *Cauchy Mutation* is applied on the penalized Ab bits.
- *P*-Test*: It is applied to each Ab in order to calculate how many of the other Ab's dominate it. All of the Ab's are sorted using the Quicksort algorithm according to these domination values.

```
procedure jREMISA
begin
repeat
     Creation of Primary TCP, UDP and ICMP Population (Pop_p)
     Empty Initialization of Secondary Population (Pop_s)
     Negative_selection(Pop_p,data_set_clean,threshold)
until (end of data_set_clean)
repeat
    FitnessFunction (ag,threshold)
    MutationCauchy(Pop_p)
    P_optimality()
    ClonalSelection(0.05)
    MutationUniform(Pop_s)
    Pop_p ← Pop_s //Copy the best Pop_s to the Pop_p of next generation
    if (networking)
        broadcast(Pop_s) // Offer non-dominated Ab's to other AIS
        processReceived()
    Endif
until (end of data_set_attack)
End
```

Algorithm 1. The pseudocode of the jREMISA algorithm

- *Clonal Selection*: %5 of the non-dominated Ab's of the primary population are selected with an elitist selection and copied to the secondary population. Copied Ab's are cloned six times in order to have a large population. Copied and cloned Ab's are mutated for n-random bit position (n=objective number (2) + Pareto-dominance value). The Ab's from the secondary population with the highest fitness values are copied to the primary population instead of the discarded Ab's because of the maximum false detection count; so that Pop_p reaches its original size. At last, all of the dominated Ab's are discarded from Pop_s.

4 Proposed Improvements on jREMISA

In this study, we did the following improvements on jREMISA in order to get better false-positive rates:

1. Hamming distance evaluations in the Negative Selection, Fitness Function and Clonal Selection steps are enhanced using r-continuous bit evaluations. Two compared chromosomes need to have at least r-continuous bits the same to be considered alike. We used the r-continuous bits requirement with the previous Hamming distance calculations in order to get a stronger evaluation.

2. The random Ab's of Negative Selection which recognize the self Ag's are discarded but they are replaced with new random Ab's until the number of mature Ab's reaches the predefined population sizes. Therefore, there will be a sufficient number of mature Ab's under any condition.
3. Minimization of false positive and negative fitness scores is used as the second objective instead of minimization of deviation from the affinity threshold.
4. Not all of the dominated Ab's in the secondary population are discarded at the end of the Clonal Selection, but the size of the secondary population is trimmed to the size of the primary population and the rest of the dominated Ab's are sorted according to their dominance values.
5. Cloned Ab's are re-evaluated with a new Fitness Function and P-optimality test in Clonal Selection since their chromosomes have changed after Uniform Mutation. Therefore, later selected Ab's of the secondary population will have even parameter values with the Ab's of the primary population.
6. Uniform Mutation is not applied to the originals of the cloned Ab's in Clonal Selection in order not to destroy their elitism.

We made all of the above additions in order to get not only good training results as the original jREMISA but also good test results. R-continuous bits evaluation decreases the false detections since it puts another strict criterion for Ab similarities. Before the second improvement we couldn't have enough number of non-self Ab populations after the Negative Selection process under high threshold values like 42 and over. We decided not to aim the minimization of the deviation from the affinity threshold since the system had better values under larger deviations as stated in third improvement.

5 Experiments

5.1 Test Designs

The tests are implemented on Pentium Core 2 Duo 2.4 GHz computers which have Windows XP SP3 operating systems. There are three different tests of the improved jREMISA with different settings of the parameters of affinity threshold values, r-continuous values and primary population sizes in order to find the best parameter group to get better true and false positive results. Finally, the original and the improved jREMISA's are compared according to the changing threshold values in order to see the improvement. Each test has 20 runs to get the mean and standard deviation values of the evaluation metrics of true positive rate, which is the fraction of all attacks that are actually detected and false positives rate, which is the fraction of all normal data that produces (false) alerts.

DARPA LLDOS 1.0 dataset is used in all of the tests. However, the truth set of this dataset is not given on the Lincoln Laboratory website. So, we created the truth set ourselves since we knew the structure of the attack and the identity of the attacker. We obtained a self-only data set to train our Ab population by removing all of the incoming traffic related to the DDoS attack by using Ethereal [14]. Secondary population Ab chromosomes are used in the tests with the original dataset which includes the attack traffic.

5.2 Results

Affinity Threshold Tests. Affinity threshold values changing from 38% up to 54% are applied to the improved jREMISA to decide the best threshold value that yields the best true and false positives. These threshold values are selected with respect to the original jREMISA study [4] in which they are experimentally obtained. TCP, UDP and ICMP population sizes are 100 chromosomes each and 10 is used as the r-continuous value. The system doesn't work for the threshold values over 54%. As it is seen on the Figures 1 and 2, there is not a great difference in these results; most of the true positive rates are above 97%, some of them are 100% and most of the false positive rates are below 1,5 %.

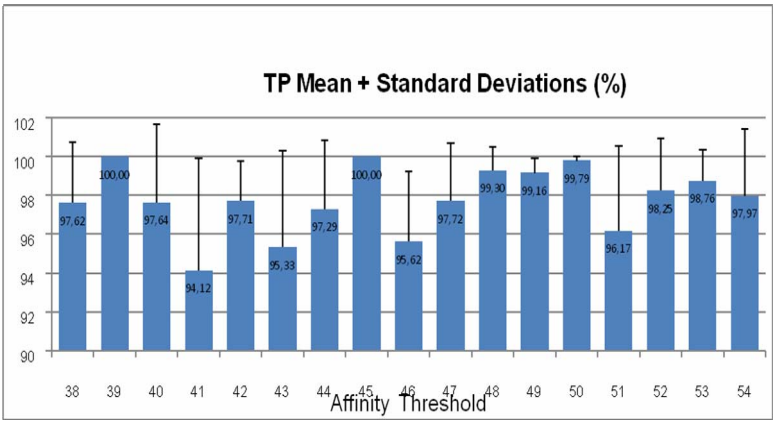


Fig. 1. True positives rates of changing affinity thresholds (%)

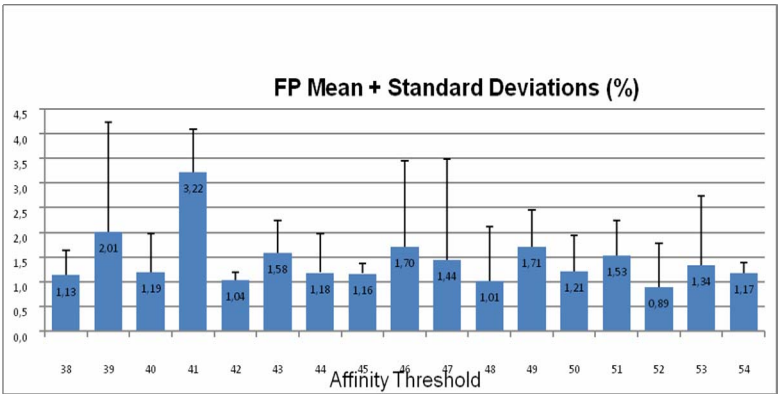


Fig. 2. False positives rates of changing affinity thresholds (%)

R-continuous Tests. R-continuous values changing from 8 up to 15 are applied to the improved jREMISA to decide the best r-continuous value that yields the best true and false positives. TCP, UDP and ICMP population sizes are 100 chromosomes each and 50% is used as affinity threshold value. As it is seen in Fig. 3, there is not a great difference in the results of true positives; most of them are above 97%. False positive rates are below 1,5 % except the first two. R-continuous value of 15 has 0% false positive rate.

Fig. 3. True and false positives rates of changing r-continuous values

Pop_p size Tests. TCP Ab numbers changing from 100 up to 500 under the condition of constant UDP and ICMP Ab population sizes of 100 chromosomes each, are applied to the improved jREMISA to decide the best TCP Ab population size that yields the best true and false positives. 50% is used as affinity threshold value and 10 is used as the r-continuous value.

Fig. 4. True and false positives rates of changing Pop_p TCP Ab numbers

As it is seen in Fig. 4, there is not a great difference in the results of true positives; all of them except the first one are 100%. Figure 4 shows that false positive rates are below 0.1 % except the first one. 500 TCP Ab number has a 0% false positive rate. The computational overhead of increasing the population size can be ignored since this training part of the IDS will be executed offline.

Comparison of the original and the improved jREMISA. When we compare the performance of the original and the improved jREMISA over the DARPA LLDOS 1.0 dataset, we see that the improved version is better than the original one. All of the

true positive rates of the improved jREMISA with TCP, UDP and ICMP population sizes of 300,100 and 100 respectively and 10 as r-continuous value, are 100% as seen in Fig. 5. On the other hand, the true positive rates of the original jREMISA are above 92% except the last two ones. The big difference in the false positive rates can be seen in the Fig. 5. All of the false positive rates of the improved jREMISA are 0% where all of the false positive rates of the original jREMISA are 99,98%. The orginal jREMISA has these high false detection rates since it was left with only successful training results which was not enough to get good test results.

Fig. 5. Comparison of true and false positives rates with changing thresholds (%)

6 Conclusion

We used the artificial immune system as a method of anomaly type intrusion detection because of the similarity between the IDS architecture and the biological immune system. We improved jREMISA [4]; a multiobjective evolutionary algorithm inspired artificial immune system, in order to get better true and false positive rates while detecting DDoS attacks on the MIT DARPA LLDOS 1.0 dataset. We added the r-continuous evaluation method, changed the Negative Selection and Clonal Selection structure, redefined the second objective while keeping the general concept the same.

We made three different tests of the improved jREMISA with different settings for the parameters of affinity threshold values, r-continuous values and primary population sizes in order to find the best parameter group to get better true and false positive results. At last, the original and the improved jREMISA's are compared using the determined good parameter groups. The tests are performed by changing the threshold values in order to see the improvement. The 100% true positive rate and 0% false positive rate of our improved algorithm is a very noteworthy success as an anomaly intrusion detection system.

This study is an important part of our project in which our objective is distributed detection of DDoS attacks in the intermediate phase using mobile agents and nature-inspired algorithms; and informing the security managers before the attack succeeds. We will combine our previous study [15] and this one in order to reach our objective. Other intrusion detection datasets will be tested with this algorithm in order to see its overall performance. Finally, this IDS can be used with real-time network traffic with an adaptive truth set.

References

1. Abraham, A., Grosan, C., Chen, Y.: Cyber Security and the Evolution of Intrusion Detection Systems. Journal of Educational Technology, Special Issue in Knowledge Management (2005), ISSN 0973-0559
2. Kannadiga, P., Zulkernine, M.: DIDMA: A Distributed Intrusion Detection System Using Mobile Agents. In: Proceeding of the ACIS 6th International Conference on Software Engineering, Networking and Parallel/Distributed Computing (SNPD/SAWN), pp. 238–245 (2005)
3. Chandler, J.A.: Security in Cyberspace: Combatting Distributed Denial of Service Attacks. University of Ottawa Law & Technology Journal 1, 231 (2003-2004)
4. Haag, C.R., Lamont, G.B., Williams, P.D., Peterson, G.L.: An artificial immune systeminspired multiobjective evolutionary algorithm with application to the detection of distributed computer network intrusions. In: GECCO 2007: Genetic and evolutionary computation Conference, London, UK (2007)
5. Du, Y., Wang, H.-Q., Pang, Y.-G.: IADIDS-Design of A Distributed Intrusion Detection System Based on Independent Agents. In: Proceedings of International Conference on Intelligent Sensing and Information Processing, pp. 254–257 (2004)
6. Mark, C., Gene, S.: Defending a Computer System using Autonomous Agents. In: Proceedings of the 18th National Information Systems Security Conference (1995)
7. Uwe, A., Julie, G., Jamie, T.: Immune System Approaches to Intrusion Detection - A Review. In: Nicosia, G., Cutello, V., Bentley, P.J., Timmis, J. (eds.) ICARIS 2004. LNCS, vol. 3239, pp. 316–329. Springer, Heidelberg (2004)
8. Lin, S.: A Survey on Solutions to Distributed Denial of Service Attacks, Research Proficiency Examination Report, TR-201, Experimental Computer System Lab, SUNY at Stony Brook (2006)
9. Mahoney, M.V., Chan, P.K.: An Analysis of the 1999 DARPA/Lincoln Laboratory Evaluation Data for Network Anomaly Detection. In: Vigna, G., Krügel, C., Jonsson, E. (eds.) RAID 2003. LNCS, vol. 2820, pp. 220–237. Springer, Heidelberg (2003)
10. Brugger, S.T., Chow, J.: An Assessment of the DARPA IDS Evaluation Dataset Using Snort, UC Davis Technical Report CSE-2007-1, Davis, CA (2007)
11. MIT Lincoln Laboratory, Information Systems Technology, `http://www.ll.mit.edu/mission/communications/ist/corpora/ideval/data/2000/LLS_DDOS_1.0.html`
12. Aickelin, U., Dasgupta, D.: Artificial Immune Systems Tutorial. In: Burke, E., Kendall, G. (eds.) Search Methodologies: Introductory Tutorials in Optimization and Decision Support Methodologies, ch. 13. Springer, Heidelberg (2005)
13. Coello, C.A., Lamont, G.B., Van Veldhuizen, D.A.: Evolutionary Algorithms for Solving Multi-Objective Problems, Genetic and Evolutionary Computation, 2nd edn. Springer, Heidelberg (2007)
14. Ethereal: Open-source network protocol analyzer, `http://www.ethereal.com`
15. Akyazi, U., Etaner-Uyar, A.S.: Distributed Intrusion Detection using Mobile Agents against DDos Attacks. In: 23rd International Symposium on Computer and Information Sciences (ISCIS). IEEE, Los Alamitos, DOI:10.1109/ISCIS, 4717920, ISBN: 978-1-4244-2880-9 (2008)

Performance Evaluation of an Artificial Neural Network-Based Adaptive Antenna Array System

Muamar Al-Bajari[1], Jamal M. Ahmed[2], and Mustafa B. Ayoob[2]

[1] Electrical Engineering and Computer Science Dept./ TU Berlin-Germany
[2] Department. of Communication/ University of Mosul-Iraq
muamar@cs.tu-berlin.de, {altaee_msc,jamal1961eng}@yahoo.com

Abstract. Efficient tracking systems are needed to constantly track multiple desired signals simultaneously in different modern wireless applications such as mobile communication, radar, and localization.

The adaptive antenna tracking system presented in this paper mainly consists of three units: data processing, Artificial Neural Network Processor (ANNP) and the optimum weights processing. The data processing unit is used to calculate the correlation matrix of the received signals, which is eventually handled by the ANNP unit. The ANNP unit is based on the architecture of a family of Radial Basis Function Neural Network (RBFNN) to perform both detection and Direction of Arrival (DOA) estimation. The optimum weights processing unit utilizes the Linear Constraint Minimum Variance (LCMV) approach, using the estimated angles of the desired signals generated by the ANNP unit, to calculate the steering matrix of the AAA system.

The performance evaluation of the system is conducted experimentally using simulation techniques in a variety of angular separations, number of sources and various Signal to Noise Ratios (SNRs).

1 Introduction

Adaptive antenna systems have been the subject of increasing interest in the recent years because of their capability of rejecting noise and interference. This is mainly due to the beam forming flexibility, which allows steering the main lobes to the desirable signals and steering nulls towards interference signals [3], [5], [8], [11].

Multiple targets and signal tracking can be considered as an indirect method for enhancing the mobile communication bandwidth utilization efficiency. This enhancement is partly due to the utilization of adaptive arrays in the base stations, which makes the performance of tracking multiple users higher and more reliable. For this antenna system to be influential, it must be capable of operating robustly and fast.

This paper is concerned with the development of an Adaptive Antenna Array (AAA) system based on Artificial Neural Networks (ANNs), a massively parallel processor that improves the tracking performance of multiple desired signals in an efficient manner.

In order to accomplish multiple tracking of desired signals, direction finding algorithms make up an essential part of the beam forming processor of the AAA system.

C. Di Chio et al. (Eds.): EvoApplications 2010, Part II, LNCS 6025, pp. 11–20, 2010.

Superresolution algorithms for direction finding have been successfully applied to the problem of DOA estimation to locate radiating sources with additive noise, uncorrelated, and correlated signals. One of the main disadvantages of the super-resolution algorithms is that they require extensive computation and as a result they are difficult to meet the real world applications requirements efficiently.

Recently, neural networks-based direction finding algorithms have been proposed for single and multiple source direction finding [1], [4], [9], [11]. It has been shown that the neural networks have the capability to track sources in real time. [1], [10] suggested that a RBFNN could be used to track the locations of mobile users. The performance of these ANNs suffered from the variability of the number of users and of a fixed angular separation since different ANNs had to be used when the number of users changed. A computation of the optimum weights of the AAA, based on the evaluation of the DOA using RBFNN was presented in [9], [10]. The training process used in [9]'s work created sensitive sets of optimum weights that caused a tremendous degradation in the performanc of the overall tracking process of the AAA.

In this paper, an algorithm used for beamforming of the AAA system to robustly allow multiple source tracking with arbitrary angular separation and SNRs levels. Fig. 1 shows the basic functional block diagram that makes up the platform upon which the simulation of this AAA system is based.

RBFNN is used to estimate the location of the desired signals, while the LCMV approach is used to calculate the optimum weights of the AAA system, consequently, the production of the desired adaptive radiation pattern.

The performance of the overall system is studied for various target angular separations and various practical SNRs. The Additive White Gaussian Noise (AWGN) is assumed.

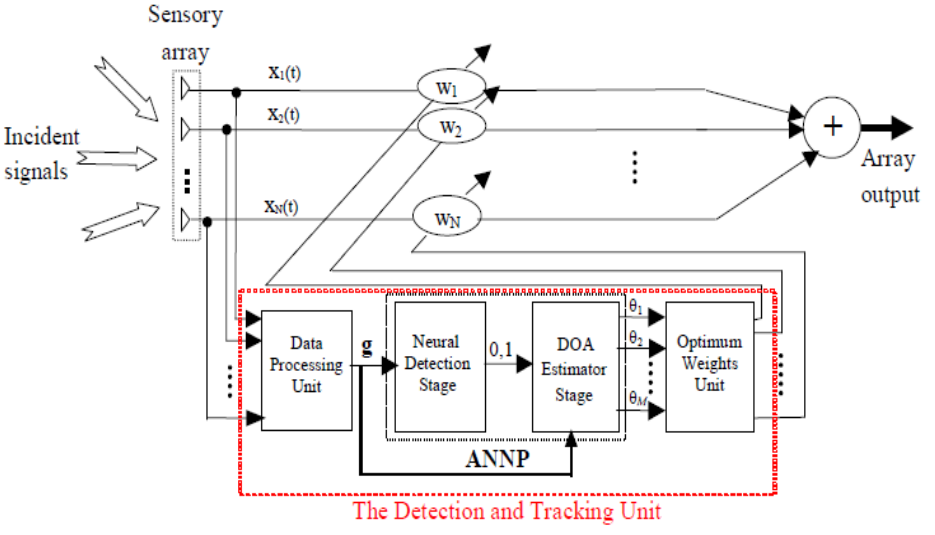


Fig. 1. Functional diagram of the Neural AAA system

2 Principle AAA System Configuration

The adaptive array functional diagram of Fig. 2 shows the principle system elements that an adaptive array must possess, if it is to achieve successfully the twin objectives of enhancing desired signal reception and rejection undesired interference signals.

The principle adaptive array system elements consist of the sensor array, the pattern-forming network, and the adaptive pattern control unit or adaptive processor that adjusts the variable weights in the pattern-forming network.

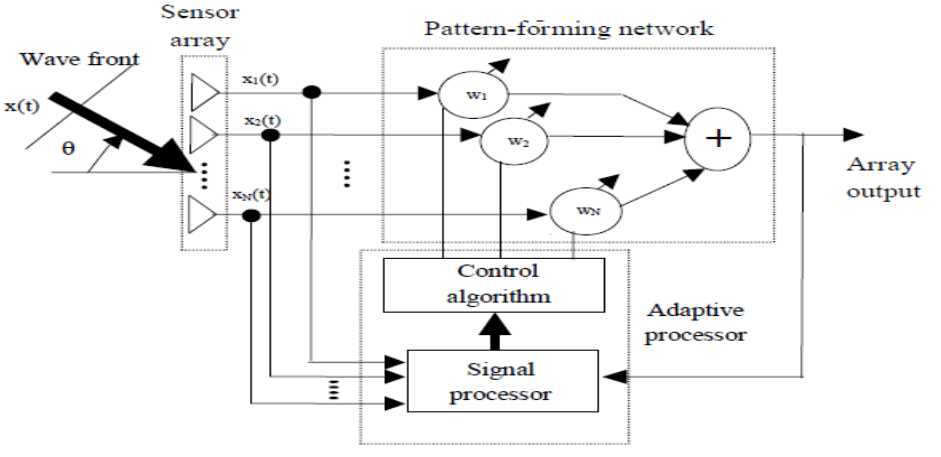


Fig. 2. Functional diagram of an N-element adaptive array

Evaluating the weights in AAA system determine the speed and accuracy of switching the beams. In statistically optimum adaptive antennas, the weights are chosen based on the data received at the array. Loosely speaking, the goal is to "optimize" the adaptive antenna array response such that the output contains minimal contributions due to noise and signals arriving from directions other than the desired signal direction. Different optimization procedures usually lead to different performance indices that would eventually fulfill the AAA system beam forming requirements. Maximum SNR, Reference Signal, and LCMV techniques are the most "popular techniques used to evaluate statistically, the weights. In many applications, none of these approaches is satisfactory. The desired signal may be of unknown strength and may not always be present, thus, preventing estimation of signal to noise correlation matrices in the maximum SNR processor. Lack of knowledge about the desired signal may prevent utilization of the Reference Signal approach. These limitations can be overcome through the application of linear constraints to the weight vector. Use of linear constraints is a general approach that permits tight control over the adapted response of the adaptive antenna [3].

The basis of operation of the LCMV approach is to constrain the response of the adaptive antenna so that the signals from the direction of interest are passed with specified gain and phase. The weights are chosen to minimize the output variance, or

power, subject to the response constraint. This has the effect of preserving the desired signal, while minimizing contributions to the output due to interfering signals and noise arriving from directions other than the direction of interest [2].

The optimal weight vector, $\mathbf{w}_{opt}$ such that the array output power is minimized while satisfying the constraint, please refer to [5], represented by equation 1,

$$\mathbf{w}_{opt} = \mathbf{\Phi}^{-1}\mathbf{C}(\mathbf{C}^{H}\mathbf{\Phi}^{-1}\mathbf{C})^{-1}\mathbf{f}_{r}. \tag{1}$$

Where the superscript "H" indicate the Hermitian (complex conjugate transpose), $\mathbf{\Phi}$ represents the spatial correlation matrix, while $\mathbf{f}_r$, represents the νx1 response vector, ν is the number of the desired signals and $\mathbf{C}$ is the constraint matrix, also called steering matrix, as will be discussed later.

3 The Smart Neural AAA System Model

In this section, $\mathbf{\Phi}$ and $\mathbf{g}$, the upper triangular part of $\mathbf{\Phi}$ that are used for training the RBFNN are developed, as shown in Fig.1.

The ANNP represents the second unit, which is mainly used for direction finding estimation. Finally, the computation of the optimum weights for the AAA is performed in the third unit using LCMV approach, which has been discussed in section 2.

3.1 Data Processing Unit

A linear array of N-elements is considered to derive a formal representation of the $\mathbf{\Phi}$ and $\mathbf{g}$. If M (M < N) is the number of narrow band plane waves, centered at frequency ω_o impinging on the array from directions θ_1, θ_2, ... θ_M The received signal at the *i*th array element would be represented by equation 2.

$$x_i(t) = \sum_{m=1}^{M} s_m(t)\, e^{-j(i-1)\Psi_m} + n_i(t) \qquad i = 1, 2, \ldots N. \tag{2}$$

Where, $s_m(t)$ is the desired mth signal, $n_i(t)$ is the noise signal received at the *i*th sensor and

$$\Psi_m = (\omega_o / l)\, d \sin(\theta_m). \tag{3}$$

where, d represents the inter-element separation, l, is the speed of light in m/s and θ_m is the angle of the *m*th wave. In matrix form the received signals can be written as

$$\mathbf{X}(t) = \mathbf{CS}(t) + \mathbf{N}(t). \tag{4}$$

$\mathbf{X}(t)$, $\mathbf{N}(t)$ and $\mathbf{S}(t)$ are given by:

$$\mathbf{X}(t) = [x_1(t)\ x_2(t)\ \ldots\ x_N(t)]^T \tag{5}$$

$$\mathbf{N}(t) = [n_1(t)\ n_2(t)\ \ldots\ n_N(t)]^T \tag{6}$$

$$\mathbf{S}(t) = [s_1(t)\ s_2(t)\ \ldots\ s_N(t)]^T \tag{7}$$

Where the superscripts "T" indicates the vector transpose, and **C** as defined in equation (8) is the NxM steering matrix of the array toward the direction of the incoming signals.

$$\mathbf{C}= [\mathbf{u}_{d1}\ \mathbf{u}_{d2} \ldots \mathbf{u}_{dm} \ldots \mathbf{u}_{dM}] \tag{8}$$

The steering vector $\mathbf{u}_{dm}$ is related to Ψ_m as given in equation (9).

$$\mathbf{U}_{dm}=[1\ e^{-j\Psi m}\ e^{-j2\Psi m} \ldots\ e^{-j(M-1)\Psi m}\ ] \tag{9}$$

Assuming that the noise signals (n_i (t), i=1,2, …, N) received at the different sensors are statistically independent white noise signals of zero mean and variance σ^2 and also independent of **S**(t), then the received spatial correlation matrix **Φ** of the received noisy signals can be expressed as in (10),

$$\mathbf{\Phi} = E\ [\mathbf{X}(t)^{*}\ \mathbf{X}(t)^{T}]\ . \tag{10}$$

Since **X**(t) and **N**(t) are independent, then

$$\mathbf{\Phi} = \mathbf{C}\ E\ [\mathbf{S}(t)\ \mathbf{S}^{H}\ (t)]\ \mathbf{C}^{H} + E\ [\ \mathbf{N}_o(t)\ \ \mathbf{N}_o(t)^{T}] \tag{11}$$

Φ for two narrowband desired signals impinging on two element Array Antenna, can be written as follows

$$\mathbf{\Phi} = a_{d1}{}^{2}\mathbf{u}_{d1}{}^{*}\mathbf{u}_{d1}{}^{T} + a_{d2}{}^{2}\mathbf{u}_{d2}{}^{*}\mathbf{u}_{d2}{}^{T} + a_{d2}{}^{2}\mathbf{u}_{d2}{}^{*}\mathbf{u}_{d2}{}^{T} + a_{d1}a_{d2}\mathbf{u}_{d1}{}^{*}\mathbf{u}_{d2}{}^{T} + a_{d2}a_{d1}\mathbf{u}_{d2}{}^{*}\mathbf{u}_{d1}{}^{T} + \sigma^{2}\mathbf{I} \tag{12}$$

Where,

a_{d1},a_{d2} are the amplitudes of $s_1(t)$ and $s_2(t)$.
$\mathbf{u}_{d1}$,$\mathbf{u}_{d2}$ are the steering vectors
σ^2 is the power of the noise.
I is the Identity matrix.

So the correlation matrix for two-desired signal will be,

$$\mathbf{\Phi}=\begin{bmatrix} a_{d1}{}^{2}+a_{d2}{}^{2}+\sigma^{2} & a_{d1}{}^{2}e^{-j\Psi d1}+a_{d2}{}^{2}e^{-j\Psi d2} \\ a_{d1}{}^{2}e^{j\Psi d1}+a_{d2}{}^{2}e^{j\Psi d2} & a_{d1}{}^{2}+a_{d2}{}^{2}+\sigma^{2} \end{bmatrix} \tag{13}$$

Array-processing algorithms, generally, utilize the correlation matrix for direction of arrival estimation purposes instead of the actual received signals **X**(t). A spatial correlation matrix that can be organized as a N^2-dimensional vector denoted by **g** is the input of the ANNP. It then follows that the number of input units at the input layer of the neural network is given by $2N^2$. This is due to the fact that twice as many input nodes for the neural network is needed since the network does not deal directly with complex numbers. The dimension of the hidden layer is equal to the number of the Gaussian functions **Q**, which can be chosen to be equal to the number of total input/output pairs in the training set if perfect recall is desired. By exploiting the symmetry in the correlation matrix **Φ** one needs only to consider either the upper or lower triangular part of the matrix. In this work, the upper triangular half of **Φ** is used. An NxN spatial correlation matrix **Φ** can be organized in a (N^2+N) dimensional vector of

real and imaginary parts denoted by **g**. For example, if N=3, then the correlation matrix as a vector will be

$$\mathbf{g_r} = [r_{11}\ r_{12}\ r_{13}\ r_{21}\ r_{22}\ r_{23}\ r_{31}\ r_{23}\ r_{33}]. \tag{14}$$

But by taking the upper triangular half of $\mathbf{\Phi}$, $\mathbf{g_r}$ will reduce to **g**:

$$\mathbf{g} = [r_{11}\ r_{12}\ r_{13}\ \ r_{22}\ r_{23}\ r_{33}]\ . \tag{15}$$

3.2 The ANNP Unit

The second unit in the adaptive processor is the ANNP. It identifies and determines the sector into which the desired signal lies and the second stage, gives an estimate of the angle of arrival. Each stage consists of 18 neural networks and each neural network consists of three layers of nodes: the input layer, the output layer, and the hidden layer. As is the case with most neural networks, the RBFNN is designed to perform an input/output mapping trained with examples. The purpose of the hidden layer in an RBFNN is to transform input data from an input space of source dimensionality to a new space of possibly higher dimensionality.

3.2.1 Architecture of an ANNP Unit

The weights from the hidden layer to the output layer are identified by following a supervised learning procedure, applied to a single layer network (the network from hidden to output layer). This supervised rule is referred to as the delta rule [6]. Once training of the ANNP is accomplished, the trained ANNP can operate in the performance mode (phase). In performance (testing) phase, the ANNP is expected to generalize, that is to respond to inputs it has never been seen before, but drawn from the same distribution as the inputs used in the training set. One way of explaining the generalization exhibited by the network during the performance phase is by remembering that after the training phase is complete the ANNP has established an approximation of the desired input/output mapping; hence, during the performance phase the ANNP produces outputs to previously unseen inputs by interpolation between the inputs used (seen) in the training phase.

The optimum antenna weights units handles the estimated direction of arrivals and computes the AAA system optimum weights, $\mathbf{w}_{opt}$ using the LCMV approach, consequently a beam pattern is generated from the AAA. The optimum weights vector is evaluated by the simulation of equation (1).

The following is a brief description of the algorithms used in the detection and DOA estimation processes.

3.2.2 Training of the Detection NN

In this adaptive processor, in some cases an arbitrary number of desired signals (sources) can be tracked without a prior knowledge of the number of the sources. The first stage consists of 18 RBFNN's, each of width θ_w. The entire angular spectrum, field of view of the antenna array, is divided into J sectors. The jth RBFNN is trained to determine if two or more signals exist within the [(j-1) θ_w, jθ_w] sector. If there is any signal present in the corresponding sector, the output of this ANN will have a maximum value in comparison with the remaining ANNs outputs. Consequently, this detection stage will output a value of (1). However, all the other ANNs detection stages will output a zero. This information is then passed to the second stage, the

DOA stage, which estimates the angles of these signals. The following is the training procedure of the detection stage.

1. Compute the correlation matrix of the received signals using equation (10), $\{\mathbf{\Phi}^q$, q=1, 2, 3, …,Q} where Q represents the of input-output pair.
2. Develop the vectors $\{\mathbf{g}^q$, q=1, 2,..., Q}.
3. Normalize the input vectors.
4. Generate input output pairs $\{\mathbf{g}^q$,100} for sources located in the sector, where q=1, 2,...,Q.
5. Employ an appropriate RBFNN in the detection to learn the training set generated in step 4.

3.2.3 Training of the DOA Estimator

The J networks of the DOA estimation stage are assigned the same spatial sectors as in the detection stage. When the output of one network from the first stage is 1, the corresponding second stage network is activated. The input to each second stage network is the normalized vector **g** as given in equation (15), while the output is the actual DOA angles of the sources. The number of the hidden nodes is the same as the number of the input nodes, i.e, it equals Nx(N+1). The optimum size of hidden nodes is not easy to determine. In this work, from extensive experimentation, it was found that acceptable results could be obtained by choosing the number of hidden nodes equal to or larger than the number of input nodes. The following is the training procedure of the DOA Estimator.

1. Compute the correlation matrix of the received signals $\{\mathbf{\Phi}^q$, q=1, 2, …, Q} using equation (10).
2. Develop the whole vectors in matrix with Q columns and N^2+N rows.
3. Normalize the matrix.
4. Generate the output matrix with Q columns and M rows (where M represents the number of the desired signals).
5. Employ an appropriate RBFNN training procedure to learn the training set generated in step (4).

4 Discussion of Simulation Results

Results pertaining to the performance of the AAA system obtained in this work can be divided into the following three parts, performance of the detection stage, DOA estimator, and beam forming system. All of these results are based on the basic configuration fig.1 and influenced by the training procedures discussed in the previous subsections. It should be remembered that a large number of research cases can be dealt with, however, for lack of space, we take some sample cases of performance of the developed beamforming system for that may be encountered in real world applications.

The adaptation and beamforming processes, in response to the optimum weight calculation, by handling the angles obtained in the DOA NN, can be demonstrated using the sample Figs. 3 – 5.

These figures demonstrate some of the main goals achieved in this research work, where different RBFNN architectures and parameters are explored. Hence, one can arbitrarily choose an angular resolution in any arbitrarily chosen sector. In addition, signals of arbitrary angles of arrival could be tracked with negligible errors.

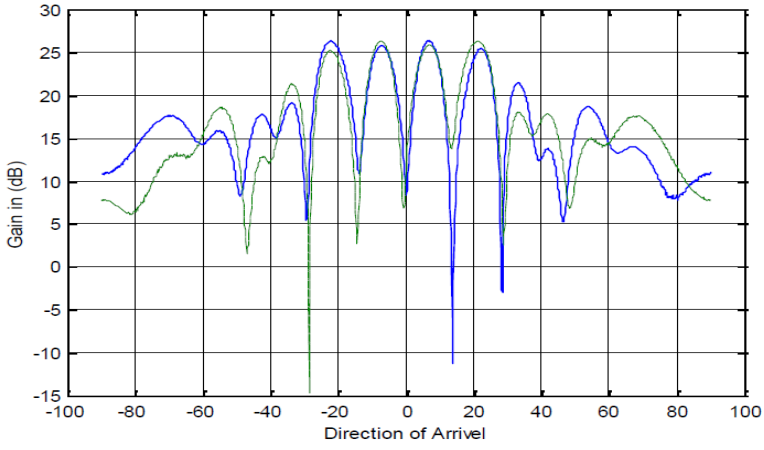


Fig. 3. Simulated real time AAA system pattern. Traning enviroment:(-45° – 45) sector, $\Delta\theta$=4°,8°,12°,16° and 4 dB SNR.Test signals: θ_1 = -22°, θ_2 = -8°, θ_3 =8°, θ_4 = 22°, $\Delta\theta$=15° and 4 dB SNR. Blue and green represent the true and test response respectively.

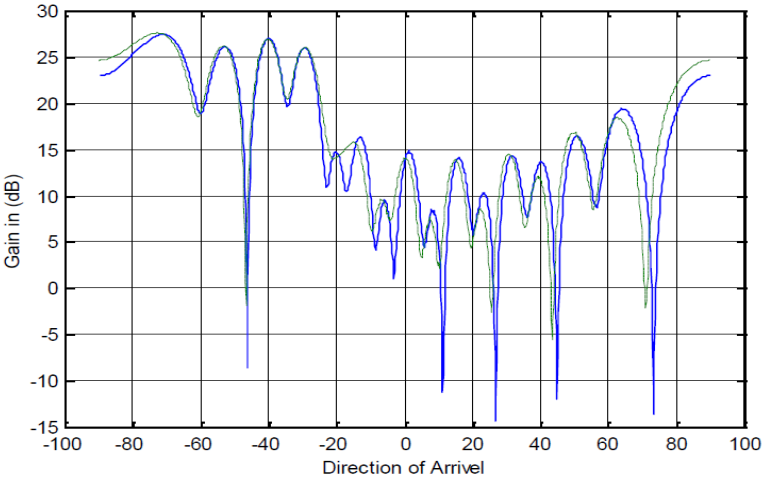


Fig. 4. Simulated real time AAA system pattern. Traning enviroment:(-90° – 0°) sector, $\Delta\theta$=4°,8°,12°,16°, and 4dB SNR.Test signals: θ_1= -30°, θ_2 = -40°, θ_3 = -55°, θ_4 = -70°, $\Delta\theta$=15°, and 4dB SNR. Blue and green represent the true and test response respectively.

Fig. 3 shows a beam pattern when angular separation between test sources is 12° and their SNR is 4 dB, although the training was performed on sources of $\Delta\theta$ = 10°.

The pattern shown in fig.4. is due to the existence of four signals in the (0° – - 90°) sector with a separation of 15°, however, the training was performed on 12°. This reflects the capability of the system to track signals it has never seen and in a sector of different sizes.

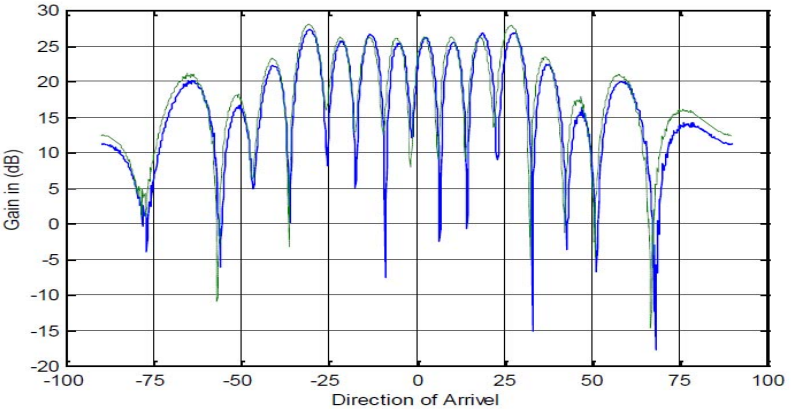


Fig. 5. Simulated real time AAA system pattern. Traning enviroment:(-45° – 45°) sector, $\Delta\theta$=2°,4°, 6°,8°,10°, and 5dB SNR. Test signals: θ_1 = -29°, θ_2 = -20°, θ_3 = 12°, θ_4 = 4°, θ_5 = -4°, θ_6 = -12°, θ_7 = -21°, θ_8 = -28°, $\Delta\theta$= 8°, and 4dB SNR. Blue and green represent the true and test responses.

In Fig.5, eight test signals existed in the (-45° – 45°) sector with an angular separation of 9° tracked perfectly with lower SNR than that trained on.

All this demonstrate the capability of the developed AAA system to form the main beam into multiple-beams, in these sample examples, to track up to eight desired signals. More results can be imparted to deal with more signals and more desired resolution in real-world applications.

5 Conclusions

A complete AAA system based on ANN has been set, simulated and tested for tracking of multiple desired signals simoultanously. The performance of each stage of the system was tested over a practical range of SNRs, arbitrary number of desired signals and over various angular separations between signals.

Using the LCMV in co-opertion with the RBFNN in beam forming and the training procedures of the NN reduced the sensitiviy of the AAA tracking to variations in the optimal weights. Some modification on this AAA system would result in more flexible cost effective multibeam tracking systems over 360° angular fields of view.

References

1. Ayestaran, et al.: Non uniform-antenna array synthesis using neural networks. Journal of Electromagnetic Waves and Applications 21(8), 1001–1011 (2007)
2. Barry, D.V., Kevin, M.B.: A Versatile Approach to Spatial Filtering. IEEE ASSP Magazine 21(9), 4–24 (1988)
3. Monzingo, R.A., Miller, T.W.: Introduction to Adaptive Arrays. John Wiley, New York (1980)

4. Mohamed, M.A., Soliman, E.A., El-Gamal, M.A.: Optimization and characterization of electromagnetically coupled patch antennas using RBF neural networks. Journal of Electromagnetic Waves and Applications 20(8), 1101–1114 (2006)
5. Compton Jr., R.T.: Adaptive Antennas Concepts and Performance. Prentice Hall, Englewood Cliffs (1988)
6. Haykin, S.: Neural Networks A Comprehensive Foundation, 2nd edn. Prentice Hall, New Jersey (1996)
7. Demuth, H., Beale, M.: Neural Network Toolbox for Use with Matlab. Mathworks Inc., Prime Park Way (1998)
8. Swales, S.M., et al.: The Performance Enhancement of Multibeam Adaptive Base-Station Antennas for Cellular Land Mobile Radio Systems. IEEE Transactions (1990)
9. Southall, H., et al.: Direction Finding In Phased Arrays with A Neural Network Beamformer. IEEE Transaction on Antennas and Propagation 43(3), 1369–1375 (1995)
10. El-Zooghby, A.H., et al.: Neural Network-Based Adaptive Beamferming for One-and-Two-Dimensional Antenna Arrays. IEEE Transactions on Antennas and Propagation 46(12), 1891–1893 (1998)
11. Sarevska, M., et al.: Null Steering Beamformer Based on RBF Neural Networks. In: 12th WSEAS International Conference on Communications, Heraklion, Greece (2008)

Automatic Parameter Tuning with Metaheuristics of the AODV Routing Protocol for Vehicular Ad-Hoc Networks

José García-Nieto and Enrique Alba

Dept. de Lenguajes y Ciencias de la Computación, University of Málaga,
ETSI Informática, Campus de Teatinos, Málaga - 29071, Spain
{jnieto,eat}@lcc.uma.es

Abstract. Communication protocol tuning can yield significant gains in energy efficiency, resource requirements, and the overall network performance, all of which is of particular importance in vehicular ad-hoc networks (VANETs). In this kind of networks, the lack of a predefined infrastructure as well as the high level of dynamism usually provoke problems such as the congestion of intermediate nodes, the appearance of jitters, and the disconnection of links. Therefore, it is crucial to make an optimal configuration of the routing protocols previously to the network deployment. In this work, we address the optimal automatic parameter tuning of a well-known routing protocol: Ad Hoc On Demand Distance Vector (AODV). For this task, we have used and compared five optimization techniques: PSO, DE, GA, ES, and SA. For our tests, a urban VANET scenario has been defined by following realistic mobility and data flow models. The experiments reveal that the produced configurations of AODV significantly improve their performance over using default parameters, as well as compared against other well-known routing protocols. Additionally, we found that PSO outperforms all the compared algorithms in efficiency and accuracy.

Keywords: Vehicular Ad Hoc Networks, On Demand Distance Vector Routing Protocol, Metaheuristics, *ns-2* Simulator.

1 Introduction

Vehicular Ad Hoc Networks (VANETs) [1] are dynamic networks composed of a set of communicating vehicles (nodes) equipped with devices which are able to spontaneously interconnect each other without any pre-existing infrastructure. This means that no service provider is present as it is usual in traditional cellular telephony. The most popular wireless networking technology available nowadays for establishing VANETs is the IEEE 802.11-b WLAN, also known as WiFi (*Wireless Fidelity*). New standards such as the IEEE 802.11p and *WiFi Direct* are promising but still not available to perform real tests with them. This implies that vehicles communicate within a limited range while moving, thus exhibiting a topology that may change quickly and in unpredictable ways.

C. Di Chio et al. (Eds.): EvoApplications 2010, Part II, LNCS 6025, pp. 21–30, 2010.

In such kind of networks, and given of limitations in coverage and energy consumption, optimizing the routing load is a high priority in the protocol design (done offline, previous to its network deployment). As a matter of fact, an optimal configuration can decisively improve QoS indicators such as the *packet delivery ratio*, the *end-to-end delay* and, the *routing load*, with their implications on enlarging bandwidth and lowering the energy consumption. However, the efficient protocol configuration for VANETs without using automatic intelligent design tools is practically impossible because of the enormous number of possibilities. This motivates the use of metaheuristic techniques [2] which appear as well-suited tools to solve this kind of problems. Unfortunately, few related approaches can be found in the specialized literature. Vanhatupa et al. [3] proposed a flexible Genetic Algorithm for optimizing channel assignment in Mesh wireless networks. In Alba et al. [4], a specialized Cellular Multi-Objective Genetic Algorithm was used for finding an optimal broadcasting strategy in urban Mobile Ah Hoc Networks (MANETs). Due to its specific design, Ant Colony Optimization (ACO) has been successfully adapted for implementing new routing protocols for MANETs (Di Caro et al. [5]), as well as for resource management (Chiang et al. [6]). More recently, Huang et al. [7] proposed a new routing protocol based on a Particle Swarm Optimizer (PSO) to make scheduling decisions for reducing the packet loss rate in a theoretical VANET scenario.

In the present work, instead of the use of an optimization technique itself as a protocol agent, our main contribution consists in improving the performance of an existing well-known routing protocol by optimally tuning its parameters. This protocol lies in the Ad Hoc On Demand Distance Vector (AODV) [8], whose performance is significantly influenced by the choice of its parameters as stated from its very initial definition in the RFC 3561. For this task, we have used and compared five optimization techniques: Particle Swarm Optimization (PSO) [9], Differential Evolution (DE) [10], Genetic Algorithm (GA) [2], Evolutionary Strategy (ES) [2], and Simulated Annealing (SA) [2]. The popular network simulator *ns-2* [11] is then used in the evaluation of the solutions (tentative routing parameters) generated by the aforementioned techniques, and providing them with the needed fitness values to guide the search.

We have chosen these algorithms because they constitute a representative subset of metaheuristics, with suitable operators for real parameter optimization, and having varied heterogeneous schemes of population and evolution. For our tests, an instance of a VANET scenario has been defined by following realistic mobility and data flow models from the urban area of Málaga in Spain. The experiments reveal that the produced configurations of AODV significantly improve the default performance of the protocol, as well as the performance of other well-known routing protocols (DSR, DSDV, FSR, TORA, and GPSR) [12].

The remaining of this paper is organized as follows. In the next section, the AODV routing protocol is introduced with its main parameters. Section 3 describes the optimization strategy, and Section 4 presents the VANET scenario evaluated here. Experiments, comparisons, and the result analysis are shown in Section 5. Finally, conclusions and further work are drawn in Section 6.

2 AODV Parameter Tuning

Ad hoc On Demand Distance Vector (AODV) is a routing protocol for ad hoc mobile networks presented in 1999 by C. Perkins and E. Royer [13]. It is one of the most studied MANET and VANET routing algorithms, often used as a de facto routing protocol. AODV was designed with the aim of reducing the high number of broadcasting packets and the latency described for its precedent: Destination Sequenced Distance Vector (DSDV) [14] routing protocol. AODV is a reactive on demand algorithm, meaning that, as in Dynamic Source Routing (DSR) [15] protocol, it builds routes among nodes only as desired by source nodes. Nevertheless, AODV uses routing tables in intermediate nodes, what makes the route discovery more efficient than in DSR, specifically in extensive networks with a large number of communicating nodes. In addition, AODV is loop-free, self-starting, and capable of both unicast and multicast routing. All these advantages, along with the fact of having lower complexity of storage than others proactive and reactive protocols (TORA, FSR, GPSR, etc.) [12], led us to choose AODV as the routing protocol to work with.

In spite of all these important advantages, one of the main drawbacks of AODV lies in the variability of its performance, which is significantly influenced by the choice of its control parameters [8]. Discovering the best values to assign to these parameters and understanding their impact on the network behavior tradeoff is still harder. In addition, tunable parameters are often defined without clear default values and even may be defined over an infinite range. Table 1 shows a set of the main AODV parameters with their default values as specified in RFC 3561. The range of values each parameter can take has been defined here by following AODV restrictions, with the aim of avoiding pointless configurations.

This way, we can use this set of parameters as a real vector solution to be automatically fine-tuned by an optimization technique, hopefully obtaining a considerably better configuration than the one of default parameters for a given VANET scenario. Additionally, analytic comparisons of both default and optimized AODV configurations as the ones done in this article can help the specialists to identify the main source of problems and assist them in the design of sophisticated routing protocols.

Table 1. Main AODV Parameters. Default values following the RFC 3561 specification.

Parameter	Default Values	Range
ACTIVE_ROUTE_TIMEOUT	3.0 s	$1.0 \cdots 10.0$
ALLOWED_HELLO_LOSS	2 HELLO packets	$1 \cdots 10$
MY_ROUTE_TIMEOUT	2.0×ACTIVE_ROUTE_TIMEOUT	$1.0 \cdots 10.0$
NET_DIAMETER	35 nodes	$1 \cdots 50$
NODE_TRAVERSAL_TIME	0.04 s	$0.01 \cdots 1.0$
NET_TRAVERSAL_TIME	2.0×NODE_TRAVERSAL_TIME ×NET_DIAMETER	$1.0 \cdots 10.0$
RREQ_RETRIES	2 tries	$1 \cdots 10$
RREQ_RATELIMIT	10.0 kbps	$1.0 \cdots 10.0$
TTL_START	1.0 s	$1.0 \cdots 10.0$
TTL_INCREMENT	2.0 s	$1.0 \cdots 10.0$
TTL_THRESHOLD	7.0 s	$1.0 \cdots 20.0$

In order to compare different AODV routing configurations (solutions), we have measured the resulted network performance (quality of service) by means of three commonly used metrics in this area [16]:

- *Packet delivery ratio (PDR).* Fraction of the data packets originated by an application that a routing protocol delivers. A data packet is counted as delivered when it is received complete and correct by the destination node.
- *Normalized routing load (NRL).* Ratio of administrative routing packet transmissions to data packets delivered. When counting transmissions, each hop is counted separately.
- *Average End-to-End delay of a data packet (AEED).* Average difference between the time the first data packet is originated by an application and the time this packet is received at its destination.

3 Optimization Strategy

Our optimization strategy is composed by basically two main parts: an optimization algorithm and a simulation procedure. The optimization part is carried out by (independently) one of the selected metaheuristic methods. All of them are specially adapted to find optimal (or quasi-optimal) solutions in continuous search spaces (which is the case in this work). The simulation procedure is the way of assigning a quantitative quality value (fitness) to the factors regulating AODV, thus leading to optimal configurations of this protocol tailored to a given VANET scenario instance. This procedure is carried out by means of the *ns-2* [11] network simulator, which has been modified in order to accept new routing parameters automatically for this present and similar future research.

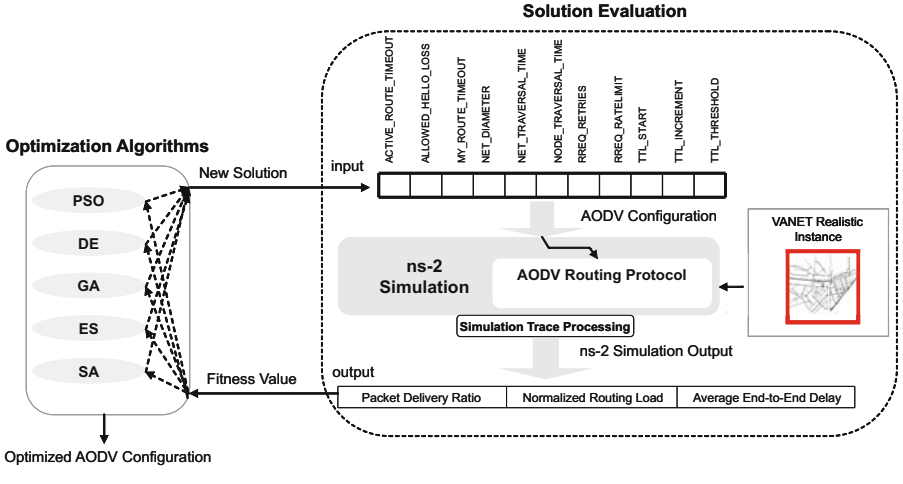


Fig. 1. Optimization strategy for AODV configuration in VANETs. The algorithms invoke the *ns-2* simulator for each solution evaluation.

In each optimization algorithm, the evaluation of a solution is carried out by means of the simulation component. As Fig. 1 illustrates, when a given algorithm generates a new solution it is immediately used for configuring the AODV. Then, *ns-2* is started and emulates the VANET scenario instance, taking its time in evaluating the scenario with buildings, signal loss, obstacles, traffic lights, vehicles, speed, covered area, etc., under the circumstances defined by the routing parameters of AODV from the algorithm. After the simulation, *ns-2* returns the global information about the *Packet Delivery Ratio* (PDR), the *Normalized Routing Load* (NRL), and the *Average End-to-End Delay* (AEED) of the whole mobile WiFi scenario (simulating 50 independent application sessions for each fitness computation). This information is used to compute the *fitness* function as follows:

$$fitness = w_1 \cdot (-PDR) + w_2 \cdot NRL + w_3 \cdot AEED \cdot C \quad (1)$$

The objective here consists in maximizing PDR, and minimizing both, NRL and AEED. As expressed in Equation 1, we used an aggregative minimizing function, and for this reason PDR was formulated with a negative sign. In this equation, factors w_1, w_2, and w_3 (0.5, 0.3 and 0.2, respectively) were used for weighing the influence of each metric on the resulted fitness value. This way, PDR takes priority over NRL and AEED since we firstly look for the routing effectiveness and secondly (but also important) for the communication efficiency. AEED is also multiplied by a constant $C = 0.01$ in order to deal with similar range to PDR and NRL.

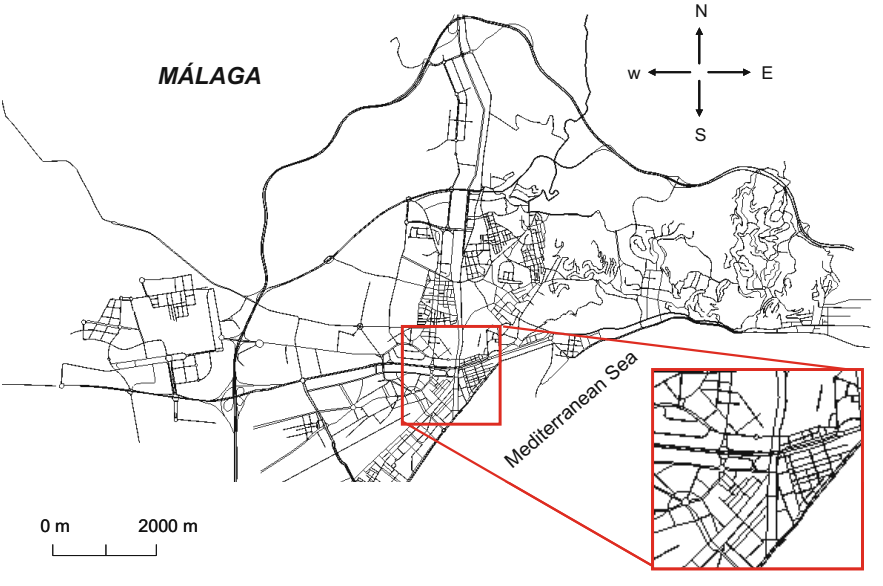


Fig. 2. Málaga real VANET scenario. Selected surface (2,000×2, 000 meters) in the downtown city.

4 VANET Scenario and Mobility Models

Since real vehicular traces are not available, and the generation of a real VANET scenario requires a great number of resources and people, we can use a traffic/network simulator to perform the movement of vehicles as well as the communication activity between them. Furthermore, we can generate realistic VANET environments by automatically selecting real city areas (taking into account road directions, signal lights, and traffic rules) from digital maps, and finally apply a realistic mobility and communication model to each vehicle agent.

Following this idea, we have generated in this work a VANET instance by mapping a metropolitan area of 2,000×2,000 m^2 from the city of Málaga (Spain). For this task, we first used the SUMO car traffic simulator [17] for describing in XML format the step by step movement of each vehicle within a 300 second time period. Fig. 2 shows the complete map of Málaga processed with SUMO traffic simulator (selected area is expanded in this figure). To use that traffic model we exported the XML input from SUMO into an *ns-2* [11] simulator movement pattern in Tcl format. A number of 50 vehicles are involved in the simulation with 4050 recorded vehicles direction/speed changes. Through the simulation time, we captured different levels of car density (between 15 and 45 vehicles/km^2), car speed (between 10 and 50 km/h), and network activity (from 2 to 50 connections).

The data flow model performs 50 sessions of the CBR (Constant Bit Rate) network application over UDP source agents in vehicles, thus interconnecting to each other by following our mobility model. The CBR data packet size is 512 bytes and packet rate is 4 packets per second. The remaining of simulation parameters are summarized in Table 2 for future reproduction purposes. We have chosen a fixed data rate since we do not aim to study the maximum throughput, but we want to investigate the ability of AODV to successfully find and maintain routes in a given VANET.

Table 2. Simulation parameters in *ns-2*

Parameter	Value
Simulation time	300 s
Simulation area	2,000×2,000 m^2
Number of vehicles	50
Vehicle speed	10-50 km/h
Propagation model	Two Ray Ground
Radio frequency	2.47 GHz
Channel bandwidth	5 Mbps
Mac protocol	802.11-b
Transmission range of vehicles	250 m
CBR data flow	50 sessions

5 Experiments

We have conducted a series of experiments by using the implementation of the five algorithms (PSO, DE, GA, ES, and SA) provided by MALLBA [18], a C++ based framework of metaheuristics for solving optimization problems. The simulation phase was carried out by running *ns-2* simulator v-2.31. For the experiments, we have made 30 independent runs of each optimization algorithm on machines with a Pentium IV 2.4 GHz core, 1 GB of RAM, and O.S. Linux Fedora core 6. Each one of these independent runs performs different vehicular mobility and communication patters based on independent random seeds inside each (*ns-2*) simulation, hence contributing to the generalization of the results. Therefore, a total of 30 different communication/mobility scenarios are analyzed for every optimization algorithm (we have 5 algorithms in our study).

5.1 Parameter Settings of the Optimization Algorithms

All studied algorithms were configured in order to perform 5,000 solution evaluations per run. Population based algorithms (PSO, DE, GA, and ES) were configured with 20 individuals. Since these algorithms perform quite different operations, we have set the parameters after preliminary executions where the computational effort in terms of time and number of evaluations was balanced. Table 3 summarizes the parameter setting specific to each algorithm.

Table 3. Summarized parameter settings of the optimization algorithms

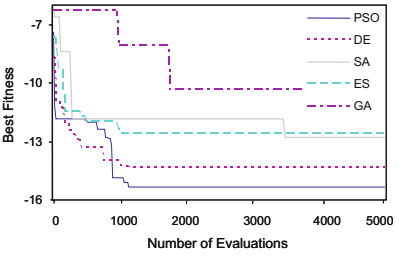
Algorithm	Operator	Parameter	Symbol	Value
PSO	Velocity Update	Local Coefficient	φ_1	2.0
		Social Coefficient	φ_2	2.0
		Inertia Weigh	w	0.5
DE	Differential Crossover	Crossover Probability	Cr	0.9
	Differential Mutation	Mutation Factor	μ	0.1
GA	Uniform Crossover	Crossover Probability	P_{cros}	0.8
	Uniform Mutation	Mutation Probability	P_{mut}	0.2
ES	Replacement	Replacement Strategy	(μ, λ)	$(20, 20)$
	Correlated Mutation	Mutation Probability	P_{mut}	0.1
	Correlated Crossover	Crossover Probability	P_{cros}	0.9
SA	Solution Update	Temperature Decay	T	0.8

5.2 Simulation Results and Comparisons

Table 4 contains the results obtained after the experimentation. The second column contains both, the mean and the standard deviation (*std*) of the resulted best fitness values (out of 30 independent runs) for each one of the five optimization algorithms. In order to provide statistical meaningful comparisons, we have applied *Friedman* and *Signed Ranked* (*Wilcoxon*) statistical tests [19] to the numerical distributions of results. We have used these non-parametric tests since resulted distributions usually violate the condition of normality required to apply parametric tests (Z *Kolmogorov-Smirnov* = 0.003). The confidence level was set to 95% (*p-value*=0.05), which allows us to ensure that these results are statistically different if they result in *p-value*<0.05.

Table 4. Results obtained by the five optimization algorithms when optimizing AODV on the Málaga VANET instance

Alg.	$Mean_{std}$	*Best*	*Fried.*	$Wilcox._p$
PSO	**-13.55 ± 1.24**	**-15.34**	**1.43**	-
DE	−12.93 ± 0.76	−14.35	1.73	2.18e-02
ES	−10.53 ± 1.74	−12.67	3.13	1.73e-06
SA	−6.12 ± 6.38	−12.76	4.10	2.87e-06
GA	−5.88 ± 2.81	−10.38	4.60	1.92e-06



Several observations can be made from these results. First of all, the best mean fitness (−13.55 ± 1.24) was reached by PSO, which also shows a reasonably low value of standard deviation. In addition, this algorithm has obtained the solution with the overall minimum fitness (−15.34), which corresponds with the AODV configuration that shows the best tradeoff in quality indicators (*PDR*=60%, *NRL*=43.06, and *AEED*=866.02) for the studied VANET. These results are statistically tested in column 4 where we can check that PSO effectively shows the lowest rank according to *Friedman* test (the lower, the better).

Second, DE obtained slightly higher mean fitness (−12.93 ± 0.76) than PSO, but with the lowest value of standard deviation. In spite of its moderate performance (ranked as second in *Friedman* test), DE shows the most robust behavior for this instance. The worst rank was obtained by GA, although showing in this case a lower standard deviation than SA. In this sense, we suspect that the trajectory search mechanism of SA can deteriorate the robustness for this problem. The graphic plotted below shows the trace of the *Best* performed runs of each algorithm where we can easily observe the early convergence and better behavior of PSO and DE with regard to SA, ES, and GA.

A last observation concerns the Signed Rank test ($Wilcox._p$ in Table 4), for which we have used PSO as control algorithm (the one with best rank) in order to confirm whether differences in distribution of results can be found or not. As we can observe in column 5, all algorithms obtained statistically worse results than PSO since *p-values* of distributions refuse the null hypothesis (< 0.05). Then we can state that, for the studied VANET instance, PSO shows the best performance compared to the rest of algorithms in the configuration of the AODV protocol.

5.3 QoS Analysis

After the analysis of the algorithms themselves, in this section we compare the results in terms of quality of service indicators. This comparison constitutes the main contribution of this work. Therefore, Table 5 shows the results of applying a set of well-known routing protocols to our VANET instance (Málaga), including AODV with its default configuration (RFC 3561). These protocols are: Dynamic Source Routing (DSR), Destination Sequence Distance Vector (DSDV), Fisheye State Routing (FSR), Temporally Ordered Routing (TORA), and Greedy Perimeter Stateless (GPSR). The best AODV configurations obtained by all studied algorithms are deployed in the bottom second half of Table 5.

Table 5. Comparison with other routing protocols

Protocol	fitness	*PDR*	*NRL*	*AEED*
AODV	−5.32	60.00%	75.33 kbps	1,038.79 ms
DSR	−5.10	42.00%	45.95 kbps	1,055.68 ms
DSDV	−4.17	28.00%	0.00 kbps	4,913.43 ms
FSR	+0.53	20.00%	0.00 kbps	5,268.15 ms
TORA	+6.57	66.66%	133.00 kbps	4.48 ms
GPSR	+47.03	100.00%	332.00 kbps	143.27 ms
AODV$_{PSO}$	**−15.34**	60.00%	43.06 kbps	866.02 ms
AODV$_{DE}$	−14.35	62.00%	47.06 kbps	1,271.45 ms
AODV$_{SA}$	−12.76	60.00%	47.00 kbps	1,552.67 ms
AODV$_{ES}$	−12.67	64.00%	53.43 kbps	1,644.97 ms
AODV$_{GA}$	−10.38	62.00%	59.32 kbps	1,409.79 ms

As we can clearly observe, all the AODV configurations computed by the metaheuristics obtained better fitness when compared to competitor routing protocols, including AODV with default parameters. This is a true improvement since even GA, the algorithm with the worst performance here, can generate a set of parameters that helps AODV to outperform all compared protocols. As to the best configuration, AODV$_{PSO}$, the protocol performance was improved by decreasing both, the NRL by 42.83% (from 75.33 kbps to 43.06 kbps) and the AEED by 16.63% (from 1,038.79 ms to 866.02 ms), whilst showing the same PDR as default AODV. Concerning other protocols, GPSR and TORA show a high packet delivery (100%) but provoking the overhead of the network (NRL=332.00 kbps). On the contrary, DSDV and FSR keep reduced the network overload but at the cost of performing a low ratio of packet delivery (PDR$\leq$28%). All this contrasts with our AODV$_{PSO}$, which always obtains percentages of PDR higher than 60% plus optimizing both, the network load and end-to-end delay, a capital need for an efficient performance in large VANETs scenarios. In summary, all these observations give us a real insight on the advantage of using PSO (and metaheuristics in general) for the tuning of routing protocols.

6 Conclusions

In this work we have addressed the optimal parameter tuning of the routing protocol AODV. For this task, we have used and compared five optimization techniques: PSO, DE, GA, ES, and SA. An instance of VANET for urban scenario has been defined by following realistic mobility and data flow models. The experiments reveal that the produced configurations of AODV significantly improve its performance with respect to using default parameters, as well as the performance of other well-known routing protocols. Specifically, for AODV, the *routing load* and the *end-to-end delay* were decreased by 42.83% and 16.63%, respectively. In addition, we found that PSO outperforms all the compared algorithms. As a future work we are presently studying the use of Multiobjective metaheuristics in the optimal configuration of routing protocols.

Acknowledgments. Authors acknowledge funds from the CICE, Junta Andalucia, under contract P07-TIC-03044 (DIRICOM project http://diricom.lcc.uma.es) and Spanish Ministry of Sciences and Innovation (MICINN) and FEDER under contract TIN2008-06491-C04-01 (M* project http://mstar.lcc.uma.es). José García-Nieto is supported by grant BES-2009-018767 from the MICINN.

References

1. Härri, J., Filali, F., Bonnet, C.: Mobility Models for Vehicular Ad Hoc Networks: A Survey and Taxonomy. Research Report RR-06-168 (March 2007)
2. Blum, C., Roli, A.: Metaheuristics in combinatorial optimization: Overview and conceptual comparison. ACM Computing Surveys 35(3), 268–308 (2003)
3. Vanhatupa, T., Hännikäinen, M., Hämäläinen, T.: Optimization of mesh WLAN channel assignment with a configurable genetic algorithm. In: WiMeshNets 2006 (2006)
4. Alba, E., et al.: A Cellular MOGA for Optimal Broadcasting Strategy in Metropolitan MANETs. Computer Communications 30(4), 685–697 (2007)
5. Di Caro, G.A., Ducatelle, F., Gambardella, L.M.: AntHocNet: An Adaptive Nature-Inspired Algorithm for Routing in Mobile Ad Hoc Networks. European Transactions on Telecommunications 16(5), 443–455 (2005)
6. Chiang, F., Chaczko, Z., Agbinya, J., Braun, R.: Ant-based topology convergence algorithms for resource management in VANETs. In: Moreno Díaz, R., Pichler, F., Quesada Arencibia, A. (eds.) EUROCAST 2007. LNCS, vol. 4739, pp. 992–1000. Springer, Heidelberg (2007)
7. Huang, C., Chuang, Y., Hu, K.: Using particle swarm optimization for QoS in ad-hoc multicast. Eng. Appl. of Artificial Intelligence (2009) (in Press)
8. Perkins, C.E., Belding-Royer, E.M., Das, S.: Ad Hoc on Demand Distance Vector (AODV) Routing. IETF RFC 3561 (2003), http://moment.cs.ucsb.edu/pub/rfc3561.txt
9. Kennedy, J., Eberhart, R.: Particle Swarm Optimization. In: IEEE International Conference on Neural Networks, November 1995, vol. 4, pp. 1942–1948 (1995)
10. Price, K.V., Storn, R., Lampinen, J.: Differential Evolution: A practical Approach to Global Optimization. Springer, London (2005)
11. The Network Simulator Project - Ns-2, http://www.isi.edu/nsnam/ns/
12. Toh, C.: Ad Hoc Wireless Networks: Protocols and Systems. Prentice Hall PTR, Upper Saddle River (2001)
13. Perkins, C.E., Royer, E.M.: Adhoc On Demand Distance Vector Routing. In: 2nd IEEE Workshop on MCSA, Metz, France, pp. 90–100 (1999)
14. Perkins, C.E., Bhagwat, P.: Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers. In: ACM SIGCOMM 1994, London, UK, pp. 234–244 (1994)
15. Johnson, D.B., Maltz, D.A., Broch, J.: DSR: the dynamic source routing protocol for multihop wireless ad hoc networks. In: Ad hoc Networking. Addison-Wesley Longman Publishing Co., Inc., Boston (2001)
16. Naumov, V., Baumann, R., Gross, T.: An evaluation of inter-vehicle ad hoc networks based on realistic vehicular traces. In: Proceedings of the 7th ACM MobiHoc, pp. 108–119. ACM, New York (2006)
17. Krajzewicz, D., Bonert, M., Wagner, P.: The open source traffic simulation package SUMO. In: RoboCup 2006, Bremen, Germany, pp. 1–10 (2006)
18. Alba, E., Luque, G., García-Nieto, J., Ordonez, G., Leguizamón, G.: MALLBA: A software library to design efficient optimisation algorithms. Int. Journal of Innovative Computing and Applications (IJICA) 1(1), 74–85 (2007)
19. Wilcox, R.: New statistical procedures for the social sciences, Hillsdale (1987)

WiMAX Network Planning Using Adaptive-Population-Size Genetic Algorithm

Ting Hu, Yuanzhu Peter Chen, and Wolfgang Banzhaf

Department of Computer Science, Memorial University, St. John's, Canada
{tingh,yzchen,banzhaf}@mun.ca

Abstract. IEEE 802.16, also known as WiMAX, is a new wireless access technology for currently increasing demand of wireless high-speed broadband service. Efficient and effective deployment of such a network to service an area of users with certain traffic demands is an important network planning problem. In this article, we resort to a Genetic Algorithm in order to yield good approximation solutions. In our method, individual representation and genetic variation operations are specifically designed to incorporate the feature of this application problem. Moreover, an adaptive population size approach inspired by neutral theory in molecular biology is applied in our algorithm to enhance its search ability. Simulation results show that our algorithm is fairly effective and robust to different scenarios of the network planning problem. By comparing to a conventional fixed population size scheme, our method is further verified to be able to accelerate the search process.

1 Introduction

WiMAX (Worldwide Inter-operability for Microwave Access) is a telecommunication technology based on the IEEE 802.16 standard in order to provide broadband wireless networks at the metropolitan scale. It intends to replace the more expensive wireline-based access technologies such as TV cable and ADSL [7]. As the standard evolves, WiMAX supports a variety of data transmission methods. It originated from the first 802.16 standard in 2002, also called WirelessMAN, where a cellular-like *point-to-multipoint* (PMP) operation is adopted. In the PMP mode, all communications are limited to be between a base station (BS) and a subscriber station (SS). With a serial of amendments later on, currently, a new working group, 802.16j, is focusing on multi-hop extensions so that the network can operate in a *mobile multi-hop relay* (MMR) mode. With the relay stations (RSs) to help, the coverage of the BSs can be increased significantly, which alleviates the line-of-sight (LOS) problem further [3].

In this article, we focus on the PMP mode of WiMAX and leave the more general model which incorporates the MMR mode as our next step work. In the PMP mode of WiMAX, there can be two types of entities to form the wireless component of the network, the BSs and SSs. The BSs form the infrastructure for the SSs. An SS is allowed to communicate to a BS directly if the channel quality is sufficient for the given data rate. A network planning problem in this case is

C. Di Chio et al. (Eds.): EvoApplications 2010, Part II, LNCS 6025, pp. 31–40, 2010.

an optimization problem to cover the SSs in a geographical area using a small number of BSs. The BSs can only be placed on a subset of pre-selected candidate sites. Typically, the locations of the SSs and their bandwidth requirements are given. In addition, the channel gains between the locations of the SSs and all BS candidate sites can also be obtained. Thus, for a given candidate site, the set of SSs that can be serviced by this site is known as well. Note that, in practice, since every BS has a capacity upper limit, it may not necessarily service all these SSs within range. We assume that there is no power control mechanism at either end of the channel.

The network planning problem considered in our work can be formulated as an *unsplittable capacitated facility location* problem. It is the most difficult flavor among the various facility location problem variants (see Section 2.2). In a broader context, the constraints enforced by this problem render the techniques used to solve the bin-packing and p-median problems not suitable here.

Given these inherent challenges of the network planning problem, we resort to an Genetic Algorithm (GA) to utilize the renowned advantages of evolutionary computing. Compared to exact algorithms, evolutionary algorithms are fairly robust to varying problem instances, and can provide a set of near-optimal solutions with similar or identical utility for multiple options. A specifically customized GA with variable population size is proposed here to embrace the properties of the WiMAX network planning problem. The effectiveness of this algorithm is further verified by simulation.

2 Background

2.1 Problem Formulation

The *network planning problem* can be modeled as a minimization problem on a weighted graph $G = (V, E)$. There are two types of vertices in the graph, i.e., $V = B \cup S$, where B represents the candidate BS sites and S represents the SSs. For each $s \in S$ and $b \in B$, there is an edge between them if the channel gain $g(s, b)$ between s and b is greater than or equal to a given threshold δ for data reception. Therefore, graph G in this case is a bi-partite graph, where there is no edge within B or S themselves. Every $s \in S$ is associated with a capacity requirement of bandwidth c_s. The candidate BS sites each have a capacity limit of C, which caps the total amount of bandwidth of its connected SSs.

A *feasible plan* is a mapping $M : S \mapsto B$ that satisfies the following constraints.

1. For each $s \in S$,
$$g(s, M(s)) \geq \delta. \tag{1}$$
2. We define the *load* of a BS $b \in B$ with an enforced capacity limit as
$$l(b) = \sum_{M(s)=b, s \in S} c_s, l(b) \leq C. \tag{2}$$

The total infrastructure cost of the network lies in the number of BSs in service. Therefore, our goal is to minimize $|M(S)|$ over all feasible plans.

2.2 Related Works

WiMAX network planning as an optimization problem in different flavors, has attracted research interests recently. When a BS has a capacity limit, the problem is called *capacitated*; otherwise, it is *uncapacitated*. Amaldi et al. [2] study the problem in uncapacitated UMTS cellular networks by formulating the problem as an Integer Program (IP), and resort to randomized greedy search and tabu search. In Yu et al. [11], a two-tier assignment variant is considered to model 802.16j MMR with the assumptions that the RSs and BSs are uncapacitated. In their solution, a fixed number of BSs is considered so that the top-level assignment can be treated by a p-median clustering. Generally, when an SS is allowed to be serviced only by one BS (or RS), we say the problem is *unsplittable*, as in the work discussed above. Alternatively, with more sophisticated scheduling and channel assignment, an SS may be serviced by multiple BSs (or RSs) equivalently. This is called *splittable*. In Lin et al. [8], a flow-based heuristic is devised to relax the capacitated IP formulation essentially to a splittable variant. This is a generalization of the problem of *capacitated facility location* [6], where an SS can be potentially serviced by all BSs with different transportation costs.

Evolutionary algorithms have also been applied to variants of the network planning problem. For instance, in the context of 802.11 access point (AP) deployment where a wired line is needed to connect every AP, the problem appears to be a multi-objective optimization to minimize the costs of both placing APs and laying down cables to join them. Moreover, the locations of APs are not constrained as in our problem formulation. Toward this AP deployment problem, both GA [10] and GP [4] have been applied and shown to achieve successful solutions. Although on different problem formulations, the literature suggests that evolutionary algorithms can be effective and robust applied to such problems.

For the more general models of p-median and bin-packing problems, there are also successful examples of using evolutionary algorithms [1]. However, the problem variant that we consider in this work is the more difficult *unsplittable capacitated facility location* problem, where a BS can only service the SSs within range. When user demands are not allowed to be split, flow-based solutions are not useful any more. It is also more difficult than the better studied bin-packing and p-median problems. First, the geographical locations of the devices (BSs and SSs) and their communication range dictate that each BS has a different set of SSs in range. Therefore, they are not equivalent in terms of capability of servicing the users. This is distinctive from bin-packing where all bins are equal in pursuit of using a minimum number of them. Second, rather than assigning the SSs to a fixed number of BSs as in p-median, our goal here is to use as few BSs as possible to minimize the infrastructure costs. To the best of our knowledge, this is the first attempt to use GA to solve such a variant.

3 Adaptive-Population-Size Genetic Algorithm

For constrained combinatorial optimization problems, genetic variation operations in evolutionary algorithms are usually destructive to invalidate an individual

as a candidate solution. Simply applying general and conventional genetic operations without specific heuristics could not be able to exploit the automatic search power of evolutionary algorithms. There is an increasing need for customizing evolutionary methods to closely incorporate the feature of a problem.

We propose a GA tailored to specifically solve the network planning problem. The individual representation and genetic variations are specifically designed suited to the characteristics of the problem. Furthermore, we incorporate a population adjustment method to enhance its search ability. The framework of our APS-GA (Adaptive-Population-Size GA) is described with a view on four specific aspects. We start out with a description of how to represent a solution to the network planning problem using a two-tier genetic structure in order to encode the BS selection and SS assignment separately (Section 3.1). Next, we outline the iterative operations applied to the population to approach the optimum (Section 3.2). Then, Section 3.3 explains the incorporation of an adaptive population size scheme proposed by us in previous work [5]. Note that the fitness of an individual is defined as the number of BSs in service. Thus, there can be many tied solutions with the same fitness but not necessarily the same set of activated BSs and associated SSs. Although this neutral diversity is not observable at the fitness level, it plays an important role in expanding the genotypic search space. The adaptive population size scheme allows a system to dynamically enhance neutral search during different stages of the evolution by population size adjustment. Last, Section 3.4 describes evolutionary operations, including crossover, mutation and a repair heuristic, which are specifically tailored to the problem.

3.1 Individual Representation

Given a set of subscriber stations S and a set of base stations B with their location information, we encode a mapping M from S to B as a two-tier chromosome. At the higher level, i.e., the *BS activation level*, we use an array of length $m = |B|$ to represent the BSs. Each locus i of this chromosome stands for a BS b_i, referring to its service list containing all the SSs assigned to it. If there is no SS connected to a BS (i.e., this BS is not needed), its service list is $\emptyset$. This is referred to as the *SS assignment level*. Such a two-tier representation is depicted in Figure 1.

For a feasible solution, the total length of the service lists should add up to $|S|$, and for each b_i the total capacity demand in the list must not exceed the BS capacity. The division of information into two tiers separates the semantics embedded in an individual. That is, the activation of a BS and the assignment of an SS to an activated BS are encoded in two separate domains. This allows us to control the genetic variations at these two levels independently, which turns out to be fairly powerful as indicated by our experiments. This two-tier genotype is distinctive from the most common representations of GA solutions to combinatorial optimization problems. In such works, the genotype usually takes a fixed form to resemble a biological gene sequence. In particular, a genotype would consist of $|S|$ loci, each of which refers to the index of the BS servicing this SS. Alternatively in the fixed-structure genotype camp, a genotype would

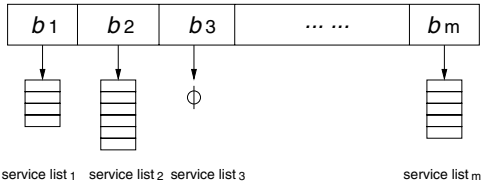


Fig. 1. Two-tier chromosome representation

represent a solution by an indicator matrix $\{0,1\}_{|M|\times|S|}$, where each column i ($i = 1, 2, \ldots, |S|$) contains exactly one 1 and $|B|-1$ 0's. One noticeable exception to this is the "multi-level encoding" in Meunier at al. [9]. In their model, the BS site activation, antenna type selection, and antenna configuration are encoded as three levels. However, the separation in our model is based on a more inherent difference of the information embedded in a solution, i.e., site activation and user assignment.

3.2 Evolution Framework

We evolve a population of individuals with adaptive size in the generational mode to approach the optimum. The process starts with randomly generating a population P_0 of a given size. The value of $|P_0|$, i.e., the initial population size, and other configuration parameters will be detailed in Section 4. Next, each individual's fitness in this initial population is evaluated. Then, the process enters a generational iteration outlined as follows.

1. Randomly pair up individuals of population P_t ($t = 0$ at the start);
2. Crossover each pair of individuals to generate $|P_t|$ offspring;
3. Repair the offspring of previous step;
4. Mutate the offspring;
5. Repair the output of previous step;
6. Evaluate offspring;
7. Calculate the next population size $|P_{t+1}|=f(|P_t|)$ (see Section 3.3);
8. Choose by truncation selection the next population P_{t+1} from the competition pool consist of $|P_t|$ parent and $|P_t|$ offspring individuals;
9. Go to Step 1 if termination criterion is not met.

The iterative process stops when the best fitness in the population has remained the same for s (stagnation threshold) individual evaluations. This termination condition will signal if the evolution stagnates. We measure how fast the algorithm leads the process to a possibly global/local optimum before stagnation by recording the number of individual evaluations elapsed so far.

3.3 Adaptive Population Size Adjustment

Adjusting population size to enhance neutral rearch has been shown by us to be able to accelerate GP evolution [5]. This idea is adopted here in the context of a

GA framework. The central idea is to measure the "rate of accepting nonsynonymous to synonymous genetic changes k_a/k_s" and to use this indicator to adjust the population size dynamically during evolution. Population size is thought to be important for accepting new genetic variations in a system. That is, if the rate of accepting nonsynonymous changes is greater than that of the synonymous changes, i.e., $k_a > k_s$, increasing the population size facilitates accepting new variations, especially the neutral or nearly neutral ones that seem to have little effect on fitness. On the other hand, if $k_a < k_s$, decreasing the population size encourages to accept more genetic variations. If $k_a = k_s$, population size is independent of such an effect. The k_a/k_s ratio is obtained for each generation for population size adjustment. This adjustment is inspired by the neutral theory from molecular biology.

Knowing the importance of neutral search, here for the network planning problem, we apply this adaptive population size approach to our GA. From one generation to the next, N_a denotes the number of *attempted* nonsynonymous changes and N_s for *attempted* synonymous changes. Specifically, for a crossover, if a valid offspring alters its fitness from either parent, this crossover is regarded as a nonsynonymous change. A mutation is regarded nonsynonymous if it changes the fitness of an individual. In evolutionary algorithms, not all genetic variations can be favored and accepted by selection. We use M_a and M_s to denote the number of *accepted* nonsynonymous and synonymous changes. Then according the definition in [5], k_a (k_s resp.) is obtained by dividing the accepted nonsynonymous (synonymous resp.) genetic changes by the attempted nonsynonymous (synonymous resp.) genetic changes.

For truncation selection (Section 3.2), the population size of a new generation is at most twice of its previous generation, and an absolute upper and lower limits of the population size is enforced, as described in Section 4.

3.4 Evolutionary Operations

Crossover. Crossover applied to two parents we denote by $x = \langle x_1, x_2, \ldots, x_m \rangle$ and $y = \langle y_1, y_2, \ldots, y_m \rangle$, to obtain two children, $x' = \langle x'_1, x'_2, \ldots, x'_m \rangle$ and $y' = \langle y'_1, y'_2, \ldots, y'_m \rangle$. It is non-trivial to design an efficient crossover operation since this operator has substantial effects on the performance of the algorithm. Here, we propose a Bi-polar Blend crossover that appropriately incorporates the feature of the network planning problem.

The Bi-polar Blend crossover strives to move the SS assignment from less loaded BSs to more loaded ones so that some will eventually no longer be needed and can be de-activated. The crossover operator is therefore a force to drive the activated BSs towards two extremes of being either very heavily or very lightly loaded. Thus, more BSs are expected to be released. To do that, we define that x' inherits the greater load from its parents and y' inherits the less load. Specifically, for each locus i ($1 \le i \le m$), we define $x'_i = \begin{cases} x_i \text{ if } l(x_i) \ge l(y_i) \\ y_i \text{ otherwise} \end{cases}$ and $y'_i = \begin{cases} x_i \text{ if } l(x_i) < l(y_i) \\ y_i \text{ otherwise.} \end{cases}$

Repair heuristic. An individual can become infeasible after the application of a genetic variation. Therefore, we conduct the following greedy repair procedure on a modified individual x. For each $s \in S$, we consider all BSs in x that service it, denoted by $\bar{B}$. We first remove all overloaded elements in $\bar{B}$, i.e., with load greater than capacity C. If $\bar{B} \neq \emptyset$, we keep the most loaded element in $\bar{B}$ and release the rest of $\bar{B}$. Otherwise, i.e., s is not serviced by any BS, we search through all BSs within range to find the *best fit* if any. Here, by best fit we mean, when s is added, the BS that has the least residual capacity. If such a best fit exists, s is added to its load. Otherwise, however, we claim that x cannot be repaired, the current genetic variation is aborted and the evolutionary process seeks to produce a new variant in the next iteration.

This repair procedure is applicable to the output of both crossover and mutation operators. Note that it also works in such a trend that the loads of activated BSs are driven towards two extremes.

Mutation. An individual is subject to a point mutation at the BS activation level. Specifically, we select an activated BS uniformly at random and simply clear its service list. We adopt such a mutation scheme for the following reasons.

First, a mutation at the BS activation level, as opposed to the SS assignment level, yields sufficient genetic alteration for the exploration of solutions. A mutation at the SS assignment level, in contrast, would yield a change which is usually too mild. Second, selecting a BS as a unit of mutation confines the changes to one locus of the network. It is, therefore, well modularized. Last, random selection of an activated BS rather than deterministic selection, say of the least loaded BS, has proved to be less directive and more effective in broadening the exploration space as of our preliminary tests.

Because this mutation inevitably invalidates the solution, the previously mentioned repair procedure is also needed.

4 Simulation

We are interested in the effectiveness and efficiency of our APS-GA to the WiMAX planning problem. In addition, we would like to verify the observation that neutral search is critical in our model by investigating the performance improvement yielded by population size adjustment.

4.1 Network Layouts and Algorithm Configuration

Considering that the size and configuration of a network layout may affect the performance of a network planning algorithm, we study two network sizes. The bandwidth demands of an SS is assumed to be 1 unit and the capacity limit for a BSs is 30 units, i.e., at most 30 SSs can be connected to any BS. The deployment area is a 1.0×1.0 2-dimensional space. We consider two network sizes with 30 (300, resp.) and 60 (600, resp.) BS candidate sites (SSs, resp.). All sites and nodes are distributed in the space uniformly at random. The coverage range of

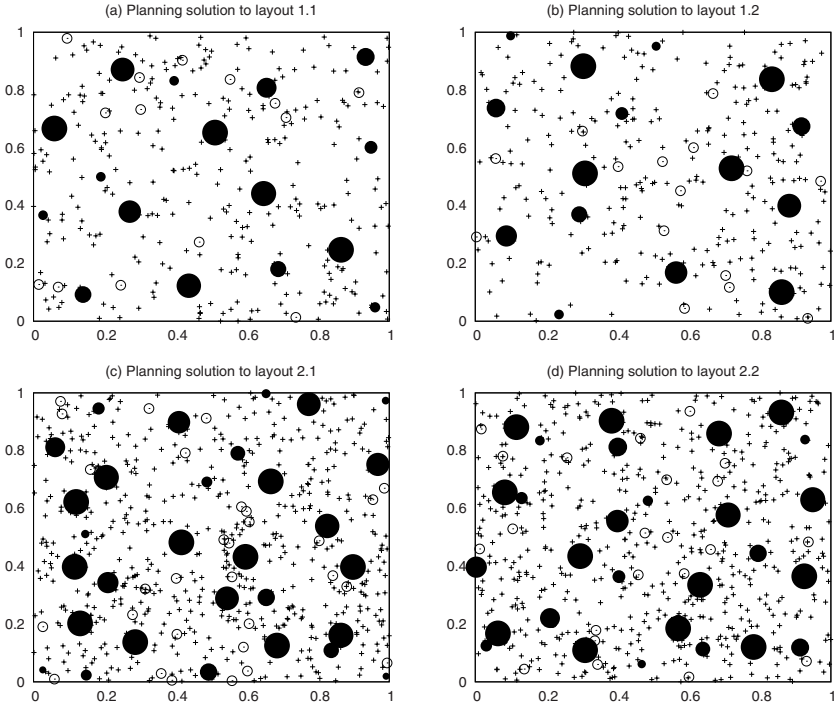


Fig. 2. Network layouts with example optimal solutions. Crosses represent SSs and circles stand for candidate BS sites. Each solid circle means that its BS is activated and its size indicates the load of this BS. Open circles represent inactivated candidate sites. For instance, in (a), the loads of 16 BSs in service vary from 5 to 30 SSs.

each BS is set to 0.2 for the first and to 0.15 for the second network size, such that a BS always has approximately the same number of SSs within range. Two layout instances are generated for each network size denoted by layout 1.1 and 1.2 (of size 1) and layout 2.1 and 2.2 (of size 2).

In all cases, we set the initial population size $|P_0|$ to 200 and the termination stagnation threshold s to 10,000 (evolution is terminated if the best fitness of the population remains unchanged for 10,000 evaluations). Further, we limit the population size to between 100 and 500 when it is varied.

4.2 Results

For each network layout, the results of 100 APS-GA runs are recorded. Figure 2 shows four network layouts with example best solutions. Observe that the loads of the BSs tend to settle on the two extremes, as expected from our Bi-polar Blend crossover and its corresponding mutation and repair operations. We also notice that although some BSs are very lightly loaded, they are indispensable to service the entire network.

Table 1. Results of APS-GA (average data over 100 runs)

	Layout 1.1	Layout 1.2	Layout 2.1	Layout 2.2
Mean of best fitness	16.0	15.9	30.3	29.1
Mean of evaluations	1862	3450	5082	5315
Median of evaluations	1810	3334	4947	4727
95% confidence interval	[1799,1925]	[3176,3723]	[4805,5360]	[4889,5740]
Mean of population size	310	346	234	250

Table 2. Results of FPS-GA (average data over 100 runs)

	Layout 1.1	Layout 1.2	Layout 2.1	Layout 2.2
Mean of best fitness	16.0	16.1	30.2	29.1
Mean of evaluations	2179	3736	5995	5609
Median of evaluations	2170	3459	5732	5250
95% confidence interval	[2157,2201]	[3510,3971]	[5571,6419]	[5199,6019]

Over 100 runs for each layout, the best solutions found by APS-GA are 16 (layout 1.1), 15 (layout 1.2), 30 (layout 2.1), and 28 (layout 2.2). These show that our method is fairly effective since about half of the candidate BSs can be retired and the average load of active BSs can be as high as 70% of the capacity limit. There also can be more than one best solution for each problem instance.

To verify that the adaptive population size scheme has indeed improved the performance of our algorithm, we compare APS-GA to a conventional fixed-population-size GA (FPS-GA) with the same operations and parameter configurations. Since population size fluctuates in APS-GA we average it during an entire evolutionary process over 100 runs for each problem instance. This number is set as the fixed population size for a FPS-GA. Therefore, it is possible and fair to compare these two algorithms.

Tables 1 and 2 show the results of these two algorithms. We collect the mean best fitness achieved at the end of evolution. Recall that evolution terminates when the best fitness of a population does not improve for 10,000 evaluations. Evaluations before stagnation are recorded as the computational cost for a population to reach its best solution. The means, medians, and the 95% confidence intervals of the number of individual evaluations are shown. It can be observed that the two algorithms perform equally well at achieving best solutions. However, APS-GA is noticeable more efficient since it always incurs smaller computational cost.

5 Concluding Remarks and Future Research

A specifically designed adaptive-population-size GA is proposed in this article to solve the WiMAX network planning problem. Computer simulation verifies its effectiveness and efficiency. This work emphasizes that it is critical to thoroughly consider the properties of a problem when applying evolutionary algorithms.

A number of interesting extensions of this work present themselves. First, other features of the solution should be considered, such as resilience. If many BSs are fully loaded, the network might be fragile to changes of SSs. Resilience to demand fluctuation (e.g., fluctuating numbers of SSs in different areas of the 2D map) can also be useful to address user mobility to a degree. Further we can see how our algorithm can adapt in such a scenario, perhaps in real time. Second, an extension of our method to solve the WiMAX network planning problem in MMR mode should be possible. That is, another layer of relay stations are used to extend network coverage. It will be interesting to see how these two levels of mapping affect each other in the framework.

Acknowledgements

W.B. thanks NSERC for support under Discovery Grant RGPIN 283304-07.

References

1. Alp, O., Erkut, E., Drezner, Z.: An efficient genetic algorithm for the p-median problem. Annals of Operations Research 122(1-4), 21–42 (2003)
2. Amaldi, A., Capone, A., Malucelli, F.: Planning UMTS base station location: optimization models with power control and algorithms. IEEE Transactions on Wireless Communications 2(5), 939–952 (2003)
3. Ghosh, D., Gupta, A., Mohapatra, P.: Scheduling in multihop WiMAX networks. ACM Mobile Computing and Communications Review 12(2), 1–11 (2007)
4. Hu, J., Goodman, E.: Wireless access point configuration by genetic programming. In: Proceedings of the IEEE Congerss on Evolutionary Computation (CEC), pp. 1178–1184 (2004)
5. Hu, T., Banzhaf, W.: The role of population size in rate of evolution in genetic programming. In: Vanneschi, L., Gustafson, S., Moraglio, A., De Falco, I., Ebner, M. (eds.) EuroGP 2009. LNCS, vol. 5481, pp. 85–96. Springer, Heidelberg (2009)
6. Levi, R., Shmoys, D.B., Swamy, C.: LP-based approximation algorithms for capacitated facility location. In: Bienstock, D., Nemhauser, G.L. (eds.) IPCO 2004. LNCS, vol. 3064, pp. 206–218. Springer, Heidelberg (2004)
7. Li, B., Qin, Y., Low, C.P., Gwee, C.L.: A survey on Mobile WiMAX. IEEE Communications Magazine 46(12), 70–75 (2007)
8. Lin, P., Ngo, H., Qiao, C., Wang, X., Wang, T., Qian, D.: Minimum cost wireless broadband overlay network planning. In: Proceedings of the IEEE International Symposium on a World of Wireless Mobile and Multimedia Networks (WoWMoM), pp. 228–236 (2006)
9. Meunier, H., Talbi, E.G., Reininger, P.: A multiobjective genetic algorithm for radio network optimization. In: Proceedings of the IEEE Congress on Evolutionary Computation (CEC), pp. 317–324 (2000)
10. Tang, K.S., Man, K.F., Kwong, S.: Wireless communication network design in IC factory. IEEE Transactions on Industrial Electronics 48(2), 452–459 (2001)
11. Yu, Y., Murphy, S., Murphy, L.: A clustering approach to planning base station and relay station locations in IEEE 802.16j multi-hop relay networks. In: Proceedings of the IEEE International Conference on Communications (ICC), pp. 2586–2591 (2008)

Markov Chain Models for Genetic Algorithm Based Topology Control in MANETs

Cem Şafak Şahin[1], Stephen Gundry[1], Elkin Urrea[1],
M. Ümit Uyar[1], Michael Conner[1],
Giorgio Bertoli[2], and Christian Pizzo[2]

[1] The City College and Graduate Center of
the City University of New York, NY, USA
{csahin,eurrea}@gc.cuny.edu, {sgundry00,uyar,conner}@ccny.cuny.edu
[2] US Army Communications-Electronics RD&E Center,
Fort Monmouth, NJ, USA
{Giorgio.Bertoli,Christian.Pizzo}@us.army.mil

Abstract. We analyze the convergence properties of our force based genetic algorithm(FGA) as a decentralized topology control mechanism distributed among software agents. FGA guides autonomous mobile agents over an unknown geographical area to obtain a uniform node distribution. The stochastic behavior of FGA makes it difficult to analyze the effects of various MANET characteristics over its convergence rate. We present ergodic homogeneous Markov chains to analyze the convergence of our FGA with respect to changing communication range of mobile nodes. Simulation experiments indicate that the increased communication range for the mobile nodes does not result in a faster convergence.

Keywords: Bio-inspired Algorithms, Genetic Algorithms, MANET, Markov Chains.

1 Introduction

Genetic algorithms (GAs) have been demonstrated as useful tools in a variety of search and optimization problems. GAs look for the best genes, i.e., the best solution or optimum result in an entire problem set. Markov chains offer an appropriate model to analyze the convergence of GAs [1] and [2]. In our previous studies [3,4] we conducted experiments to show the convergence of our force-based GA (called FGA) for topology control of nodes in mobile ad hoc networks (MANETs). In this paper, we present an ergodic homogeneous Markov chain to model the convergence properties of FGA. Since the population of our FGA, like all GA-based algorithms, only depends on the population of previous generation in a probabilistic manner, Markov chain is an appropriate method to analyze the convergence of our FGA. We study the effects of communication range on the convergence of the FGA, which is used by each mobile node to select the best location, speed and direction among exponentially large number of choices.

One can envision many applications for our GA-based topology control algorithm ranging from transportation systems, mine field clearing, to urban search

C. Di Chio et al. (Eds.): EvoApplications 2010, Part II, LNCS 6025, pp. 41–50, 2010.

and rescue operations after natural disasters where mobile nodes can operate in difficult to access areas. In all such applications, the problem of self-spreading the mobile nodes over a given geographical terrain becomes more challenging since: *(a)* geographical area may dramatically change in a short time period, *(b)* the number of mobile nodes may rapidly and unpredictably increase or decrease due to various reasons such as malfunctions or loss of mobile nodes, *(c)* mobile nodes do not have access neither to navigation map nor to GPS devices, but can only have limited local information from neighbors, and *(d)* mobile nodes are deployed into a terrain from a single entry point which is more realistic approach than random or other types of initial distributions. Our GA-based approach is a good candidate for solving this class of problems since it evolves towards a better solution as the conditions of the environment changes dynamically.

Successful applications of GAs on MANETs include graph optimization [5,6,7], multi-objective optimization [8], routing applications [9], broadcasting strategies [10], and traffic distribution [11]. Their main difference with our approach is that FGA controls the topology in MANETs without a central controller or global network knowledge, but only using with local information from neighboring nodes. *Markov chain*s are widely used to provide a formal structure for analyzing stochastic algorithms [12,13,14,15].

Our paper is organized as follows. Section 2 presents our distributed FGA. The finite Markov chain model of our FGA and convergence analysis are in Section 3. Simulation experiments are presented in Section 4.

2 Our Forced-Based GA

In our earlier work, we introduced a forced-based GA (FGA) [3] [4] inspired by the molecular forced-based distribution in physics [16]. In FGA, each node is applied a force by its near neighbors (i.e, the nodes located within its communication range, R_{com}), which should sum up to zero at the equilibrium. This force can be used as a fitness value to assign the node's speed and location. It is important to highlight that, although inspired by it, FGA is fundamentally different than the deterministic approach of [16]. In our FGA, the nodes are not pre-distributed (up to 90%), the nodes can adjust their speed and direction using local information, and there is no central controller. Our FGA runs as a distributed software agent in each node, making our approach a suitable candidate for MANETs.

In our mobility model [17], each mobile agent can move into one of six hexagonal directions in the area boundaries. A mobile node uses the total force applied to it by the neighboring nodes located in its communication range to decide the next direction and speed. The force value on a node depends on the distance between the node, the location and number of neighboring nodes within its communication range. The force from a closer neighbor is greater than the farther one. The force value can be used as a fitness value of the corresponding mobile node. To calculate a node's fitness value, the absolute normalized force value for each mobile agent is added into its fitness value. A smaller fitness shows a better position for a mobile node since it indicates that the total force applied to it by

its neighbors cancels each other out (or the neighbors apply no force at all since the node may be positioned in an isolated location by itself without any neighbors). The *mean node degree* ($\overline{N}$) is shown to be an effective measure indicating the number of neighbors to construct a fitness function for a given total number of nodes, communication range (we assume, for simplicity, communication range is the same for all mobile nodes), and a geographical area as shown by [17]. $\overline{N}$ is the expected number of node degree to maximize the coverage. Therefore, using $\overline{N}$, we can calculate the total force applied to a mobile node ρ as follows:

$$F(\rho) = \sum_{i=0}^{k} \sum_{j=0}^{k} \overline{N} \cdot (R_{com} - \mid ((x - x_i) + (y - y_j)) \mid) \tag{1}$$

where k is the total number of neighbors, (x, y) is the current coordinate value for the node, ρ is the node ID, and (x_i, y_j) is the location of a neighbor node.

There are three main objectives for achieving the optimum mobile nodes separation in an unknown terrain. The first is to have a fully connected network. This objective is one of most important requirements for MANETs since it provides a multi-hop communication capability between any two nodes in the network. Furthermore, this communication capability between any two mobile agents is maintained while nodes are mobile. The second is to maximize the area occupied by mobile agents while minimizing the intersection between nodes' communication coverage. The third objective is to provide an optimum number of neighbors for each node depending on the network density. The second and third objectives result in better utility usage and less communication overhead due to the use of the minimum required number of mobile nodes.

In summary, a mobile agent gathers information about its neighboring nodes' speed, direction, and location, and then, using the fitness function defined in Eq. (1), proceeds to run its software agent, where the FGA is located, to generate several chromosomes representing candidate solutions for the next generations. These candidates are ordered according to their fitness values from the lowest to the highest. The lowest fitness corresponds to the solution representing the least amount of force applied to a node, and, hence, the best one among the candidate solutions for that generation.

3 Ergodic Homogeneous Finite Markov Chains

GAs use different sets of chromosomes in every population and can be modeled by Markov chains. A simplified behavior of a node running our FGA can be characterized by a finite Markov chain using the node's direction (up, up-right, down-right, down, down-left, or up-left based on the hexagonal lattice), fitness (good or bad), and speed (mobile or immobile) as seen in Fig. 1. In this model, for simplicity, we assumed that mobile agents are capable of changing their directions arbitrarily without stopping. Different values of the node fitness are merged into two distinct values as either good or bad (shown as *1* or *0* in Fig. 1, respectively). $d(i)$ is the number of neighbors for a mobile agent i and $\bar{N}$ is the

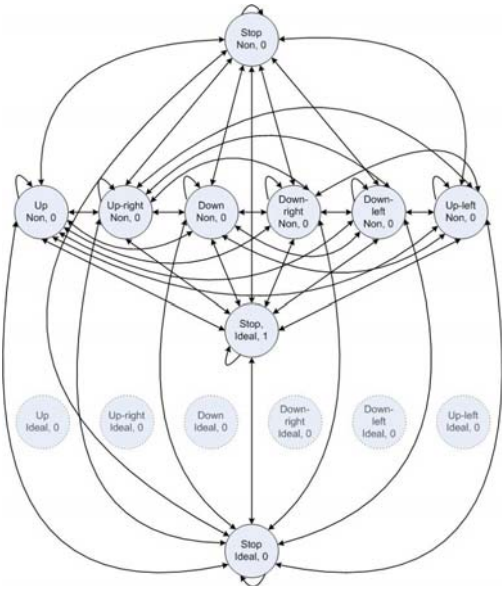


Fig. 1. Markov chain model for our FGA (each state is connected to each of the states in dotted lines, which are not shown for simplicity)

ideal number of neighbors [17] to maximize the area coverage. The state where $\bar{N} - 1 \leq d(i) \leq \bar{N} + 1$ is marked as *ideal* in the Markov chain; otherwise, it is marked as *non-ideal* (shown as Non in Fig. 1).

As can be seen in Fig. 1, the simplified Markov chain model of our GA-based algorithm has 15 states (without this simplification, the model would have in the order of 10^6 states using fitness resolution of 10^4, 10 different speeds, and six directions). If a mobile agent is moving on one of the six directions, its state must be one of the 12 states based on its number of neighbors: six directions with the ideal number of neighbors, and six directions with non-ideal number of neighbors. Speed and fitness are inherently covered by including direction into the state information. The remaining three states are (stop, non, 0) state where the node is immobile due to the non-ideal number of neighbors and zero fitness, (stop, ideal, 0) state where the mobile node is immobile, the fitness is 0 in spite of the ideal number of neighbors, and (stop, ideal, 1) state where the mobile node does not move because of having ideal number of neighbors with a fitness of 1 (the desired final state in our problem). If a mobile node reaches the final state, the mobile agent has the desired number of neighbors at the correct locations using Eq. 1 and stops moving (perhaps until another node comes and disrupts its equilibrium).

A *transition (stochastic) matrix* is a mathematical representation of a Markov model, which specifies the probabilities that the mobile agents move from one state to another in unit time. The sum of the entries in each row of the stochastic matrix must be one because this is the sum of the probabilities of making a

transition from a given state to each of the other states. The probability values of the transition matrix for our simplified Markov chain model cannot be calculated analytically because of the stochastic nature of GAs, and, hence, they must be estimated. We set up a set of experiments in our simulation testbed and collect data concerning the observed state of the mobile nodes. When our FGA decides a mobile node's next speed and direction using the neighborhood information, the mobile node moves from one state to another in our simplified Markov chain shown in Fig. 1. Performing more experiments increases the accuracy of the probability values in the transition matrix since noise in the data is eliminated. After running enough experiments, the transition matrix can be formed using the data showing the observed state of each mobile node at each time unit. The resulting transition matrix from the simulation experiments is a *right* transition matrix with nonnegative elements and entries in each row adding to one.

A homogeneous Markov chain on a finite space X has some initial distribution ν and has a transition matrix (i.e., Markov kernel) that is equivalent for every transition step (i.e., $P_i = P$, where $i = 1, 2, \cdots$). The distribution of states $x \epsilon X$ at times $n \geq 0$ is given by $P^{(n)}(x_0, \cdots, x_n) = \nu(x_0)P_1(x_0, x_1) \cdots P_n(x_{n-1}, x_n)$. An important subset of Markov chains includes those that are *ergodic*, where the Markov chain is *irreducible* and *aperiodic* as defined below. (The proofs for these lemmas are skipped here for brevity, but can be found in [18])

Lemma 1. *A Markov chain for our* FGA *can be called irreducible if and only if* $P(\tau_y < \infty | x_0 = x) > 0$ *for all* $x, y \epsilon X$ *assuming* $P^0(x_0 = x)$ *and* τ_y *is the shortest number of steps from* x *to* y *(i.e., the probability of moving from one state to another is finite).*

Definition 1. *The period (dx) of a state* x *in a Markov chain is defined as the number of steps it takes to return to the same state* x*.*

Lemma 2. *When* $dx = 1$ *in Def. 1 for all states (i.e., each state has a self-loop transition), the Markov chain is aperiodic.*

Lemma 3. *The Markov chain representation of our* FGA *is irreducible and aperiodic and therefore ergodic.*

3.1 Convergence of Ergodic Homogeneous Finite Markov Chain

Some measures used in probability will be extremely useful in proving the convergence of homogeneous Markov chain. For a finite set X with distributions μ and ν on X, the total variation is given by $\|\mu - \nu\| = \sum_n |\mu(x) - \nu(x)|$. Building on this concept is Dobrushin's contraction coefficient [15] which provides a rough measure of orthogonality between the distributions in a Markov kernel. This is defined as $c(P) = \frac{1}{2} \cdot \underbrace{max}_{x,y} |P(x, \cdot) - P(y, \cdot)|$, where c is the contraction coefficient and P is a transition matrix. This measure can be understood as $\frac{1}{2}$ the largest total variation between any two rows in the transition matrix. $c(P) = 1$ when any two distributions of the Markov kernel are disjoint. $c(P) = 0$ when all of

the rows in the transition matrix $P(x, \cdot)$ are equal. Using these concepts, we can make the following statements (proofs are given by Winkler in [15]):

Lemma 4. *Let μ and ν be the probability distributions and let P and Q be Markov kernels: $|\mu P - \nu P| \leq c(P)\,|\mu - \nu|$, $c(PQ) \leq c(P)c(Q) \Rightarrow |\mu P - \nu P| \leq |\mu - \nu|$, $|\mu P - \nu P| \leq 2 \cdot c(P)$.*

Lemma 5. *For each Markov kernel P the sequence $(c(P^n))n \geq 0$ decreases.*

Lemma 6. *If P is primitive then the sequence decreases to 0.*

Theorem 1. *For a primitive Markov kernel P on a finite space with an invariant distribution μ, uniformly in all distributions ν, $\nu P^n \rightarrow \mu$ as $n \rightarrow \infty$.*

In Lemma 4, Winkler shows that the interaction of distributions with an ergodic system (transition matrix) reduces the orthogonality between them (assuming the distributions are not disjoint). Lemmas 5 and 6 state that as the ergodic system iterates through generations, it converges to a stationary distribution. Theorem 1 generalizes this result to any initial distribution. Using Theorem 1 we can state that our FGA will converge to a stationary behavior:

Theorem 2. *Since the Markov kernel for the simplified* FGA *is ergodic, it will converge to a stationary distribution.*

Proof. (*sketch*) It is shown in [18] that the Markov kernel for our FGA is irreducible, aperiodic and ergodic. Therefore, using Theorem 1, it will converge to a stationary distribution. □

To support the statement given by Theorem 2, experimentally we can find an approximation for the final stationary distribution for our FGA model. We can then verify the rate of convergence for various R_{com} values for the nodes as shown in the next section.

4 Simulation Experiments of Convergence for Our Forced-Based GA

We implemented simulation software in Java to study the effectiveness of our distributed FGA for a uniform distribution of knowledge sharing mobile agents [3,17]. In our experimental setup the mobile nodes enter an unknown geographical area without any prior information nor with a global control unit. Without loss of generality, each mobile node has the same limited communication range (R_{com}), and, hence, can only be aware of its neighbors. Each mobile node runs its own FGA application to guide its movement based on its current status of neighboring nodes. To smooth out the noise, each experiment is repeated for 50 times with the same initial values and the same initial node deployments. There are a total of $N = 100$ mobile agents, initially placed at the north-east corner of the geographical area (100x100 hexagonal grids) for more realistic experiments. For $R_{com} = 10, 12$, and 15, mean node degree [17] is $4, 6$, and 9, respectively. The

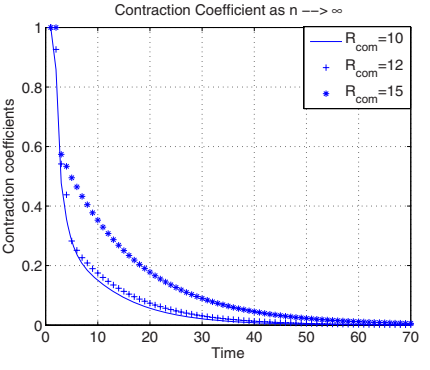


Fig. 2. Contraction Coefficient as $n \to \infty$ for $R_{com} = 10$, 12, and 15

transition matrix with the size of (15x15) is not shown here due to the space limitations.

Fig. 2 shows the Dobrushin's contraction coefficients, which provide a rough measure of orthogonality between initial and final distributions in the Markov kernel (based on Theorem 1), for $R_{com} = 10$, 12, and 15 when n goes to infinity. Markov kernel for our FGA reaches the final state for $R_{com} = 10$, 12, and 15 when $n \approx 45$, 50, and 70, respectively. Fig. 2 also shows that the mobile nodes in a MANET spend more time to reach their final states when their communication range is larger. It is an expected result since a larger communication range emulates a denser network. In a dense network, mobile nodes have more than $\bar{N}$ neighbors. We observe that the mobile nodes in the experiments with large R_{com} values have to move more than those with smaller R_{com} values because they have to travel a greater distance to acquire better fitness values. This excessive movement results in late convergence for experiments with larger R_{com} values.

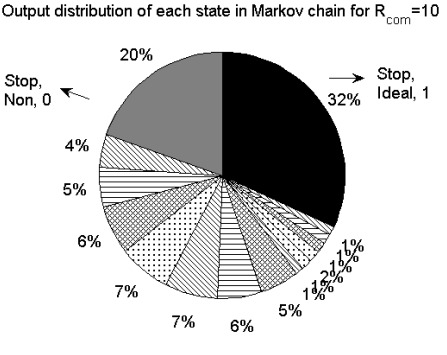


Fig. 3. Output distribution of each state in Markov chain for $R_{com} = 10$

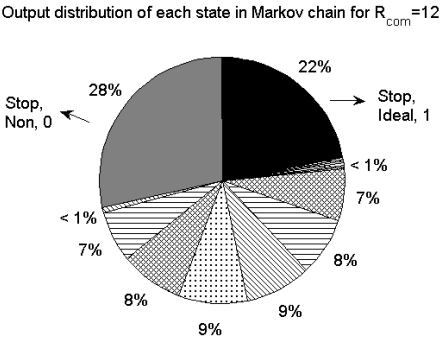


Fig. 4. Output distribution of each state in Markov chain for $R_{com} = 12$

Figs. 3, 4, and 5 present the output distributions for $R_{com} = 10$, 12, and 15, respectively, displaying the convergence characteristics of Markov kernel's states for each experiment as the nodes perform our distributed FGA and time approaches infinity. In Fig. 3, the mobile nodes in the final state have the highest probability of 32% when time approaches infinity (the mobile nodes have the desired number of neighbors at the correct locations with the summed force of almost 0 in Eq. 1). The probability of reaching a stop state with poor fitness and non-ideal number of neighbors is 20%. Mobile agent in this state may have a poor fitness value due to having more neighbors than $\bar{N}$. In Fig 4, the probability for the nodes which are immobile and with an ideal number of neighbors is 22%. However, the state where a node has non-perfect fitness, immobile and with a non-ideal number of neighbors has the highest probability of 28%. When the communication range is increased to 15 (Fig 5), the probability that being in the state with non-perfect fitness, immobile, and non-ideal number of neighbors is the highest (33%). For the larger R_{com} values, a node will have more neighbors within its range and, hence, a lower value of fitness.

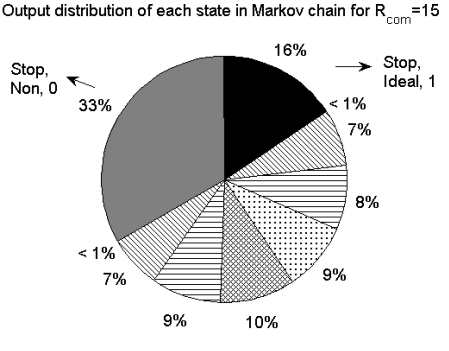


Fig. 5. Output distribution of each state in Markov chain for $R_{com} = 15$

Another important perspective to evaluate the performance of different communication ranges is the distance that the nodes travel. In MANET applications, energy conservation is vital. If the mobile agents have a larger than necessary communication range, it results in a fast energy consumption due to increased movement as seen in Figs. 2, 3, 4, and 5.

5 Conclusion and Future Work

In this paper, we study the effects of communication range on the convergence of our FGA [3,4]. A simplified Markov chain is introduced to analyze the convergence of FGA using Dobrushin's contraction coefficients. The simulation experiments show that the nodes using shorter communication ranges require less movement and converge faster. Larger communication ranges unnecessarily increase the communication among the near neighbors of each node, making convergence decisions harder, and consumes more energy. Future work will include a more detailed convergence analysis of our FGA, including the parameters such as number of mobile nodes and different initial node distributions.

Acknowledgments. This work has been supported by U.S. Army Communications-Electronics RD&E Center. The contents of this document represent the views of the authors and are not necessarily the official views of, or are endorsed by, the U.S. Government, Department of Defense, Department of the Army, or the U.S. Army Communications-Electronics RD&E Center.

This work has been supported by the National Science Foundation grants ECS-0421159 and CNS-0619577.

References

1. Eiben, A.E., Aarts, E., Hee, K.M.: Global convergence of genetic algorithms: A markov chain analysis. In: PPSN I: Proceedings of the 1st Workshop on Parallel Problem Solving from Nature, pp. 4–12. Springer, London (1991)
2. Khosraviani, M., Pour-Mozafari, S., Ebadzadeh, M.M.: Convergence analysis of quantum-inspired genetic algorithms with the population of a single individual. In: GECCO 2008: Proceedings of the 10th annual conference on Genetic and evolutionary computation, pp. 1115–1116. ACM, New York (2008)
3. Sahin, C.S., Urrea, U., Uyar, M.U., Conner, M., Hokelek, I., Bertoli, G., Pizzo, C.: Genetic algorithms for self-spreading nodes in manets. In: GECCO 2008: Proceedings of the 10th annual conference on Genetic and evolutionary computation, pp. 1115–1116. ACM, New York (2008)
4. Sahin, C.S., Urrea, E., Uyar, M.U., Conner, M., Bertoli, G., Pizzo, C.: Design of genetic algorithms for topology control of unmanned vehicles. Special Issue of the Int. J. of Applied Decision Sciences on Decision Support Systems for Unmanned Vehicles (2010) (accepted)
5. Chiang, T., Liu, C., Huang, Y.: A near-optimal multicast scheme for mobile ad hoc networks using a hybrid genetic algorithm. Expert Syst. Appl. 33(3), 734–742 (2007)

6. Abdullah, J., Parish, D.J.: Node connectivity index as mobility metric for ga based qos routing in manet. In: Mobility 2007: Proceedings of the 4th international conference on mobile technology, applications and systems and the 1st international symposium on Computer human interaction in mobile technology, pp. 104–111. ACM, New York (2007)
7. Sapienza, T.J.: Optimizing quality of service of wireless mobile ad-hoc networks using evolutionary computation. In: CSIIRW 2008: Proceedings of the 4th annual workshop on Cyber security and information intelligence research, pp. 1–5. ACM, New York (2008)
8. Kotecha, K., Popat, S.: Multi objective genetic algorithm based adaptive qos routing in manet. In: Proc. of IEEE Congress on Evolutionary Computation, CEC 2007, pp. 1423–1428 (2007)
9. Yuste, A.J., Trujillo, F.D., Trivio, A., Casilari, E.: An adaptive gateway discovery for mobile ad hoc networks. In: MobiWac 2007: Proceedings of the 5th ACM international workshop on Mobility management and wireless access, pp. 159–162. ACM, New York (2007)
10. Alba, E., Dorronsoro, B., Luna, F., Nebro, A.J., Bouvry, P., Hogie, L.: A cellular multi-objective genetic algorithm for optimal broadcasting strategy in metropolitan manets. Comput. Commun. 30(4), 685–697 (2007)
11. Pan, Y., Peng, W., Lu, X.: A genetic algorithm on multi-sensor networks lifetime optimization. Wireless Algorithms, Systems, and Applications 4138(4), 295–306 (2006)
12. Suzuki, J.: A markov chain analysis on simple genetic algorithms. IEEE Transactions on Systems, Man and Cybernetics 25, 655–659 (1995)
13. Rudolph, G.: Convergence analysis of canonical genetic algorithms. IEEE Transactions on Neural Networks 5 (1994)
14. Horn, J.: Finite markov chain analysis of genetic algorithms with niching. In: Proceedings of the Fifth International Conference on Genetic Algorithms, pp. 110–117. Morgan Kaufmann, San Francisco (1993)
15. Winkler, G.: Image Analysis, Random Fields and Markov Chains Monte Carlo Methods. Springer, Heidelberg (2006)
16. Heo, N., Varshney, P.: A distributed self spreading algorithm for mobile wireless sensor networks. IEEE Wireless Communications and Networking (WCNC) 3(1), 1597–1602 (2003)
17. Urrea, E., Sahin, C.S., Uyar, M.U., Conner, M., Hokelek, I., Bertoli, G., Pizzo, C.: Bioinspired topology control for knowledge sharing mobile agents. Mobile Ad Hoc Networks. Special Issue on BioInspired Computing 7(4), 677–689 (2009)
18. Sahin, C.S.: Topology Control of Autonomous Mobile Agents Using Genetic Algorithms in MANETs, Electrical Engineering, Graduate Center of the City University of New York (in progress)

Particle Swarm Optimization for Coverage Maximization and Energy Conservation in Wireless Sensor Networks

Nor Azlina Ab. Aziz[1], Ammar W. Mohemmed[2], and Mengjie Zhang[2]

[1] Faculty of Engineering and Technology, Multimedia University,
Jalan Ayer Keroh, 75450 Melaka, Malaysia
[2] School of Engineering and Computer Science, Victoria University of Wellington,
PO Box 600, Wellington, New Zealand
azlina.aziz@mmu.edu.my, {ammar,mengjie.zhang}@ecs.vuw.ac.nz

Abstract. Two key issues in mobile Wireless Sensors Network (WSN) are coverage and energy conservation. A high coverage rate ensures a high quality of service of the WSN. Energy conservation prolongs the network lifetime. These two issues are correlated, as coverage improvement in mobile WSN requires the sensors to move, which is one of the main factors of energy consumption. Therefore coverage optimization should take into consideration the available energy. Considering the sensors limited energy, this paper proposes a PSO based algorithm for maximizing the coverage subject to a constraint on the maximum distance any sensor can move. The simulation results show that the proposed algorithm achieves good coverage and significantly reduces the energy consumption for sensors repositioning.

Keywords: Constrained problem optimization, coverage, energy conservation, fuzzy, particle swarm optimization, Voronoi diagram, wireless sensor networks.

1 Introduction

A Wireless Sensor Network (WSN) is a group of low-cost, low-power, multifunctional and small size wireless sensor nodes that work together to sense the environment, perform simple data processing and communicate wirelessly over a short distance [1]. By mounting the sensors on mobile platforms such as Robomote [2], they can move, self deploy and self repair adding more to their value [3].

The coverage problem is one of the major concerns in WSNs and it is usually used as a key to evaluate the WSN quality of service (QoS) [4]. Central to the coverage problem is the question on how to guarantee that each of the points in the region of interest (ROI) is covered. For sensors with an actuation capability, the initial coverage can be improved by moving them so that a better coverage is achieved [3, 5-8]. However, even though mobility is an advantage to WSNs, it is a high energy consumption task [2], hence movement planning is another important issue in WSNs. It is desired to achieve optimal coverage and at the same time not to relax the mobility due to the fact that sensors have limited energy supply. Therefore the sensors' maximum distance moved is restricted to conserve energy for other tasks.

C. Di Chio et al. (Eds.): EvoApplications 2010, Part II, LNCS 6025, pp. 51–60, 2010.

Several researchers have addressed this problem. Kwok et al.'s algorithm [5] attempts to improve the coverage, taking into account minimizing the traveled distance. The ROI is divided into a grid. The sensors move to take positions on the grid points, and the movement is restricted by the fuel resources allocated. Howard and Poduri [3] proposed the virtual field concept to WSN. It assumes that the sensor nodes and obstacles have potential fields which exert virtual forces. These forces will cause the nodes to repel each other and the obstacles, resulting in physical movement by the sensors. The movement will continue until either the sensing fields of the sensors no longer overlap or they cannot detect each other. Although this method ensures full coverage and full connectivity, it relies highly on the sensor mobility, which is expensive energy consumption [2]. Zou and Chakrabarty proposed a similar concept, where the sensors move due to their exerted forces [6]. However, the sensors are restricted to make only one limited physical movement.

Chellappan et al. [7] also considered limiting sensors' movements in their sensors deployment strategy. Two objectives are considered. The first objective is to evenly deploy the sensors throughout the ROI, and the second objective is to minimize the total number of movements. Given an initial deployment of limited mobility sensors in a field clustered into multiple regions with assigned weights corresponding to the number of sensors needed, the deployment problem is to determine a movement plan for the sensors that minimizes the variance in the number of sensors among the regions, and simultaneously minimizes the sensor movements. A similar concept is proposed in [8].

In this paper, the coverage problem is considered using evolutionary algorithm. The question which this paper attempts to answer is "How to maximize the WSN coverage through intelligent repositioning of the sensors and at the same time keeping the energy consumed in moving the sensors below a threshold". Clearly this question represents a typical optimization problem of maximizing coverage subject to a maximum distance moved by any sensor. Reducing the distance moved means reducing the energy consumed. The proposed algorithm, $WSNPSO_{con}$ uses Particle Swarm Optimization (PSO) to find the best locations of the sensors according to a penalty based fitness function. The fitness function uses the Voronoi diagram to measure the quality of the coverage. A fuzzy penalty system finds the penalty parameters.

In a related work, Wu et. al. [9] used PSO to optimize coverage in a mobile WSN and reduce the communication energy consumption in cluster based sensor networks by electing the best set of cluster heads. The coverage is evaluated using grid based fitness function. An algorithm known as virtual force directed co-evolutionary PSO (VFCPSO) is introduced in [10]. However, no consideration to minimize energy is taken. In [11], a multiobjective WSN problem is considered, where the objectives are to maximize coverage and minimize energy consumption due to sensor communications. Our algorithm considers the energy consumed in sensor movement.

The rest of this paper is organized as follows. Section 2 introduces the basic concept of PSO. The proposed algorithm is presented in section 3, and followed by the simulation results and discussion in section 4. Finally, section 5 concludes the paper.

2 Particle Swarm Optimization

Particle swarm optimization is a population-based optimization tool inspired by the natural social behavior of certain organisms like bird flocking and fish schooling as developed by Kennedy and Eberhart [12]. This behavior is imitated in PSO where particles (agents) fly over the search domain influenced by their experience and the experience of the surrounding neighbors. The algorithmic flow in PSO starts with a population of particles. Each particle $\boldsymbol{i}$ has a position, $\mathbf{X}_i$ that represents a potential solution for the studied problem and a velocity $\mathbf{V}_i$ that determines the next move. The velocity is influenced by two factors; The previous best found position by the particle $\mathbf{P}_i$, and the best position found so far by the neighbor particles $\mathbf{P}_g$. The search for optimal position (solution) is performed by updating particle velocities and positions according to the following two equations:

$$\mathbf{V}_i = w \times \mathbf{V}_i + c_1 \times rand() \times (\mathbf{P}_i - \mathbf{X}_i) + c_2 \times Rand() \times (\mathbf{P}_g - \mathbf{X}_i) \quad (1)$$

$$\mathbf{X}_i = \mathbf{X}_i + \mathbf{V}_i \quad (2)$$

The inertia weight w, is used to control the effect of the previous velocity in the current velocity and also to control the exploration and exploitation ability of particles. A time decreasing inertia weight encourages high exploration at the beginning and fine tuning at the end of the search [13]. c_1 and c_2 are the learning factors to control the effect of $\mathbf{P}_i$ and $\mathbf{P}_g$. *rand()* and *Rand()* are two independent random numbers in the range of [0.0,1.0]. The $\mathbf{V}_i$ value is clamped to $\pm\mathbf{V}_{max}$ to prevent the particles from exploding and straying too far from the optimal search space.

The quality of the solution is evaluated by a problem-dependent fitness function $f(\mathbf{X}_i)$. If the current solution is better than the fitness of $\mathbf{P}_i$ or $\mathbf{P}_g$, the best value will be replaced by the current solution accordingly. This update process will continue until stopping criterion is met, usually when either the maximum iteration is achieved or the target solution is attained. When the stopping criterion is satisfied, the best particle found so far ($\mathbf{X}_i$* and fitness $\boldsymbol{f}$) is taken as the optimal (or near optimal) solution. The PSO algorithm is shown in Fig. 1.

```
For each particle initialize Xi and Vi
Do{
        For each particle {
          Calculate fitness value;
          Update Pi if the current fitness value is better than Pi;
          Determine Pg: choose the particle position with the best fitness value of all the
        neighbors as the Pg;
        }
        Update Xi*
        For each particle {
                Update Vi according to Eq.1;
                Update Xi according to Eq.2;
        }
} While maximum iteration or ideal fitness is not attained;
```

Fig. 1. PSO algorithm

3 PSO for WSN Coverage Optimization

A PSO based algorithm to solve the WSN coverage problem was proposed by the authors in [14]. The algorithm utilizes the Voronoi diagram to evaluate the fitness function to measure the coverage. Using the Voronoi diagram gave good coverage and better time efficiency than other measurement methods like using the grid. The Voronoi diagram is a partition of sites in such a way that points inside a polygon are closer to the site inside the polygon than any other sites, thus one of the vertices of the polygon is the farthest point of the polygon to the site inside it [15]. However, no consideration is imposed to reduce energy consumption due to the sensor repositioning. Therefore, we are extending the work to consider the energy consumption. The ROI is assumed to be a two-dimensional square area and the WSN is homogeneous; all the sensors have similar sensing radius. In addition, it is assumed that the sensors know their positions and possess locomotion capability, thus they are able to move and change position. The optimization algorithm is executed at a base station after an initial random placement. Then, the final optimal position is transferred to the sensors to take their final positions.

3.1 Particle Encoding

The coding of the particle is straightforward. A particle encodes the positions of the sensors in the ROI which is assumed here to be a two dimensional square area. The position of a sensor j is described by the coordinate (y_j, z_j). Considering M number of sensor nodes, the particle encoding is depicted in Fig.2. Thus, the size of the particle is twice the number of sensors. The final best particle represents the optimum positions of the sensors that lead to the minimum coverage holes

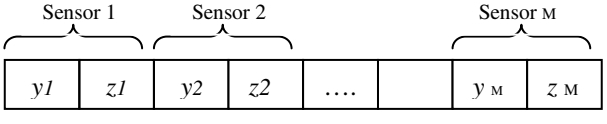


Fig. 2. Particle encoding

3.2 Fitness Function

Maximizing the coverage is done through minimizing the area of the *coverage holes.* A coverage hole is an area not covered by the sensing field of any sensor. The total area of the coverage holes are computed based on the Voronoi diagram. With the sensors acting as the sites, if all Voronoi polygons vertices are covered, then the ROI is fully covered; otherwise coverage holes exist [16]. Therefore, the fitness function is expressed as:

$$minimize : \sum_{point \in interest\_points} coverage\_hole(point) \tag{3}$$

where *coverage_hole* is the area not covered by sensors around a defined set of points called *interest_points*. The set of *interest_points* includes two groups of points: (a) vertices of the Voronoi polygons, obtained from the computed Voronoi diagram, (b) a number of points distributed evenly on the boundary of the polygons. The Voronoi diagram is computed based on the sensors positions encoded by the particle. The boundary points, whose number is selected experimentally, act as pulling forces that prevent the sensors from congregating around a particular point in the ROI. The effect of the boundary points on the final positions of the sensors based on simulation experiments is shown in Fig.3. Without including boundary points, the sensors will congregate around a point as shown in Fig.3(b), while too many points will pull the sensors strongly towards the boundary of the ROI (Fig.3(d)).

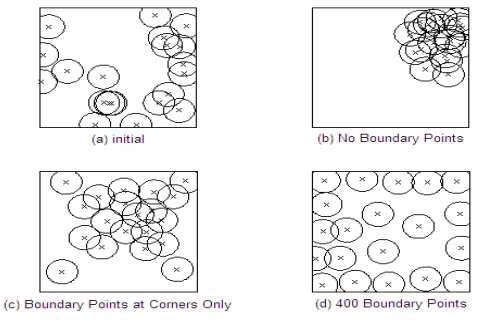


Fig. 3. Effect of boundary points

The area of *coverage_hole* around the interest points is approximated based on the position of the interest point whether it is a vertex or a boundary point as shown in Fig.4. The fitness is the summation of the area of the coverage holes in the ROI. Ideally, the fitness value should equal zero, indicating that there is no coverage holes exist.

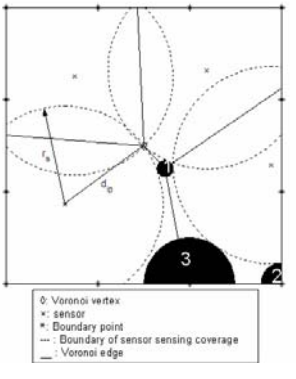


Fig. 4. Hole area estimation

Eq.(3) does not consider limiting the distance moved by the sensors to their final positions. Sensor movement is an energy consumption task, hence Eq.(3) is extended to a constrained fitness function as follows:

$$\begin{aligned} &\textit{minimize:} && \sum_{point \in interest\_points} coverage\_holes(point) \\ &\textit{subject to} && d_{mov} \le D_{\max} \end{aligned} \tag{4}$$

where d_{mov} is the maximum distance moved by any sensor and D_{max} is the maximum distance any sensor is allowed to move. Eq.(4) can be rewritten with the constraint included in a penalty term:

$$\textit{minimize} \quad \sum_{point \in interest\_points} coverage\_holes(point) + \gamma P(d_{mov}) \tag{5}$$

where γ is a positive value penalty parameter and $P(d_{mov})$ is the penalty function. The penalty function penalizes any solution outside the feasible solution. An absolute value penalty function $P(d_{mov})$ is used here;

$$P(d_{mov}) = \max(0, (d_{mov} - D_{\max})) \tag{6}$$

$P(d_{mov})$ is equal to zero as long as the constraint is obeyed, but when the constraint is violated, $P(d_{mov})$ is equal to some positive value. The accuracy of the approximation of the optimal solution found by the penalty method is controlled by γ. A large value of γ will result in a heavier penalty to any breach of the constraint, however a very severe penalty will make it hard to find an optimal solution [17]. On the contrary, a small penalty might be too lenient, thus causing infeasible solution.

In [18] a fuzzy penalty approach is proposed to provide a suitable value of penalty to the objective function based on the condition of the solution. A similar concept is used in this paper. Fig. 5 represents the Fuzzy set used to determine the penalty parameter, γ, based on the value of d_{mov}.

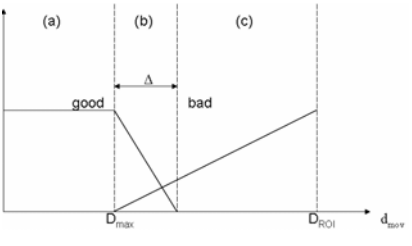


Fig. 5. Fuzzy set for γ based on d_{mov}

D_{ROI} in Fig.5 is the maximum distance the sensors can move without crossing the boundary of the ROI. The γ value is determined based on d_{mov} which is divided into three parts. The solutions from part (a) are good feasible solutions while part (c) solutions are bad solutions that are too far from the feasible area. Part (b) represents interesting infeasible solutions; infeasible but close to the feasible region [19]. This

section reduces the rigidity of the penalty method. An exponential function is chosen to reflect the values of γ.

$$\gamma = \exp^{\alpha} \tag{7}$$

$$\text{where} \quad \alpha = \begin{cases} 0 & \text{if } d_{mov} \le D_{\max} \\ \dfrac{d_{mov} - D_{\max}}{\Delta} & \text{if } D_{\max} < d_{mov} \le D_{\max} + \Delta \\ 2 \times \left(\dfrac{d_{mov} - (D_{\max} + \Delta)}{D_{ROI} - (D_{\max} + \Delta)} \right) + 1 & \text{if } D_{\max} + \Delta < d_{mov} \le D_{ROI} \end{cases}$$

The value of α in Eq. (7) is chosen based on which part does the current solution fell in.

The operational flow diagram of $WSNPSO_{con}$ is shown in Fig.6. The algorithm starts with the sensors randomly deployed. In every iteration, the maximum distance moved by any particle d_{mov} is passed to the fuzzy system to compute a new value of the penalty parameter γ. This value is passed to PSO to be used in the fitness function. This process will continue until either one of the stopping conditions – maximum iteration or 100% coverage with $d_{mov} \le D_{\max}$ – is met.

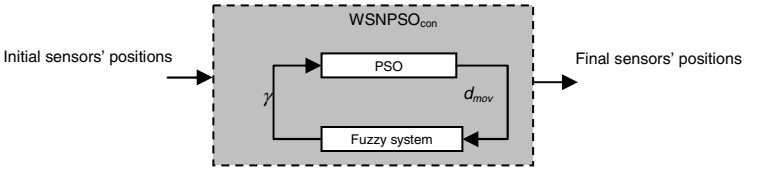


Fig. 6. $WSNPSO_{con}$ operation diagram

4 Results and Discussion

The performance of $WSNPSO_{con}$ is investigated through simulation experiments and compared with *WSNPSO* [14], which does not consider the distance moved. The objective of the experiments is to investigate the performance of the proposed algorithm in increasing coverage and conserving energy, measured using the maximum distance moved. The number of particles is set to 20, the maximum velocity to 4, and linearly decreasing inertia weight is adopted in the [0.5, 2.0] range. The learning factors for c_1 and c_2 are both set to 2.0. The algorithm runs to a maximum number of 1000 iterations. Six tests of different number of sensors and areas are conducted. All the sensors used have sensing range r_s = 7. The maximum moving distance allowed for each of the sensors (D_{max}) is set to 20 for all tests. Table 1 shows the parameters of these tests.

Table 1. Tests parameters

	Size of ROI	Number of sensors	Ideal coverage*	No. of Boundary Points	Description	
					ROI	Density
Test I	50x50	10	≈62%	20	Smaller	Sparse
Test II	50x50	20	100%	40	Smaller	Optimal
Test III	50x50	30	100%	40	Smaller	Dense
Test IV	100x100	60	≈92%	20	Larger	Sparse
Test V	100x100	80	100%	40	Larger	Optimal
Test VI	100x100	100	100%	40	Larger	Dense

* Ideally coverage is 100% if number of sensors: $= \frac{A}{3 \times \sqrt{3} \times r_s^2 / 2}$ where A is area of ROI.

Table 2. Results of simulations

Test I			
	Coverage %	Execution Time (sec)	Max. distance travel
Initial	41.28		
WSNPSO	58.36	14.4187	26.59
$WSNPSO_{con}$	57.96	15.9554	19.29
Test II			
Initial	65.47		
WSNPSO	94.23	28.5554	28.38
$WSNPSO_{con}$	93.34	30.0798	18.90
Test III			
Initial	80.58		
WSNPSO	98.80	44.1984	24.84
$WSNPSO_{con}$	98.64	46.4775	19.18
Test IV			
Initial	56.90		
WSNPSO	77.62	112.8704	43.43
$WSNPSO_{con}$	73.56	114.7657	19.65
Test V			
Initial	68.09		
WSNPSO	88.13	174.9234	38.31
$WSNPSO_{con}$	81.66	176.7735	19.79
Test VI			
Initial	74.70		
WSNPSO	92.70	247.4455	42.51
$WSNPSO_{con}$	84.90	250.4641	19.77

The coverage, the maximum distance travelled among the sensors and the execution time averaged over 20 runs are recorded in Table 2. The table shows that both *WSNPSO* and $WSNPSO_{con}$ significantly improve the initial coverage of the networks. The maximum distance travelled by any sensor when $WSNPSO_{con}$ is used is always below the threshold value of 20. The coverage of the two algorithms is similar to each

other in small ROIs but the difference becomes clearer when the ROI size is larger. This is expected as the constraint in large ROI is restricting the $WSNPSO_{con}$ to search more severely than what it does in small ROI, hence this reduces its coverage. As for density level, both algorithms show that the denser the network the better is the final coverage on the expense of longer execution time. For a better clarity, Fig.7 shows an example of sensors positions before and after using $WSNPSO_{con.}$

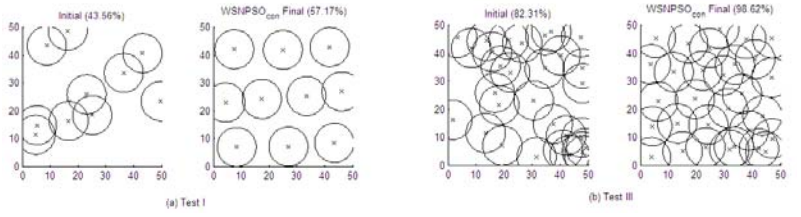


Fig. 7. Initial and final sensors arrangement of $WSNPSO_{con}$

5 Conclusion

This paper presents $WSNPSO_{con}$ as an algorithm based on PSO for optimizing the WSN coverage problem while conserving energy. $WSNPSO_{con}$ sees the two problems as a constrained optimization problem where the coverage is maximized by moving the sensors subject to a maximum distance moved. Simulation results show that the proposed algorithm succeeds in maximizing the coverage and ensures the energy is saved and kept below the threshold value.

References

1. Zhao, J., Wen, Y., Shang, R., Wang, G.: Optimizing Sensor Node Distribution with Genetic Algorithm in Wireless Sensor Network. In: Yin, F.-L., Wang, J., Guo, C. (eds.) ISNN 2004. LNCS, vol. 3174, pp. 242–247. Springer, Heidelberg (2004)
2. Dantu, K., Rahimi, M., Shah, H., Babel, S., Dhariwal, A., Sukhatme, G.: Robomote: Enabling Mobility In Sensor Networks. In: IEEE/ACM 4th International Symposium on Information Processing in Sensor Networks, pp. 404–409 (2005)
3. Howard, A., Poduri, S.: Potential Field Methods for Mobile-Sensor-Network Deployment. In: Bulusu, N., Jha, S. (eds.) Wireless Sensor Networks A System Perspective, pp. 21–33. Artech House, London (2005)
4. Cardei, M., Wu, J.: Coverage in Wireless Sensor Networks. In: Ilyas, M., Mahgoub, I. (eds.) Handbook of Sensor Networks: Compact Wireless and Wired Sensing Systems, pp. 19-1–19-12. CRC Press, USA (2005)
5. Kwok, K.S., Driessen, B.J., Phillips, C.A., Tovey, C.A.: Analyzing the Multiple-target-multiple-agent Scenario Using Optimal Assignment Algorithms. In: Proc. of SPIE, vol. 3209 (1997)
6. Zou, Y., Chakrabarty, K.: Sensor deployment and target localization based on virtual forces. In: Twenty-Second Annual Joint Conference of the IEEE Computer and Communications Societies, vol. 2, pp. 1293–1303. IEEE, USA (2003)

7. Chellapan, S., Gu, W., Bai, X., Xuan, D., Ma, B., Zhang, K.: Deploying Wireless Sensor Networks under Limited Mobility Constraints. IEEE Transactions on Mobile Computing 6(10), 1142–1157 (2007)
8. Wu, J., Yang, S.: SMART: A Scan-based Movement-assisted Sensor Deployment Method in Wireless Sensor Networks. In: Proceedings IEEE 24th Annual Joint Conference of the IEEE Computer and Communications Societies, pp. 2313–2324 (2005)
9. Wu, X., Shu, L., Yang, J., Xu, H., Cho, J., Lee, S.: Swarm Based Sensor Deployment Optimization in Ad hoc Sensor Networks. In: Second International Conference on Embedded Software and Systems, pp. 533–541 (2005)
10. Wang, X., Wang, S., Ma, J.J.: An Improve Co-evolutionary Particle Swarm Optimization for Wireless Sensor Networks with Dynamic Deployment. Sensors 7(3), 354–370 (2007)
11. Wang, X., Ma, J.J., Wang, S., Bi, D.W.: Distributed Particle Swarm Optimization and Simulated Annealing for Energy-efficient Coverage in Wireless Sensor Networks. Sensors 7(5), 628–648 (2007)
12. Kennedy, J., Eberhart, R.C.: Particle swarm optimization. In: Proc. IEEE International Conference on Neural Networks, pp. 1942–1948 (1995)
13. Shi, Y.H., Eberhart, R.C.: A modified particle swarm optimizer. In: IEEE International Conference on Evolutionary Computation, pp. 69–73 (1998)
14. Ab. Aziz, N.A., Mohemmed, A.W., Alias, M.Y.: A Wireless Sensor Network Coverage Optimization Algorithm Based on Particle Swarm Optimization and Voronoi Diagram. In: IEEE International Conference on Networking, Sensing and Control, pp. 602–607 (2009)
15. Aurenhammer, F., Klein, R.: Voronoi diagrams. In: Sack, J., Urrutia, G. (eds.) Handbook of Computational Geometry, pp. 201–290. Elsevier Science Publishing, Amsterdam (2000)
16. Xu, K., Takahara, G., Hassanein, H.: On the Robustness of Grid-Based Deployment in Wireless Sensor Networks. In: Proc. International Wireless Communications and Mobile Computing Conf., pp. 1183–1188 (2006)
17. Smith, A.E., Coit, D.W.: Constraint Handling Techniques - Penalty Functions. In: Baeck, T., Fogel, D., Michalewicz, Z. (eds.) Handbook of Evolutionary Computation, ch. C5.2. Oxford University Press and Institute of Physics Publishing, Bristol (1996)
18. Wu, B., Yu, X., Liu, L.: Fuzzy Penalty Function Approach for Constrained Function Optimization with Evolutionary Algorithms. In: Proceedings of the 8th International Conference on Neural Information Processing, pp. 299–304 (2001)
19. Michalewicz, Z., Schoenauer, M.: Evolutionary Algorithms for Constrained Parameter Optimization Problems. Evolutionary Computation 4(1), 1–32 (1996)

Efficient Load Balancing for a Resilient Packet Ring Using Artificial Bee Colony

Anabela Moreira Bernardino[1], Eugénia Moreira Bernardino[1], Juan Manuel Sánchez-Pérez[2], Juan Antonio Gómez-Pulido[2], and Miguel Angel Vega-Rodríguez[2]

[1] Department of Computer Science, School of Technology and Management, School of Technology and Management, Polytechnic Institute of Leiria, 2411-901 Leiria, Portugal
{anabela.bernardino,eugenia.bernardino}@ipleiria.pt
[2] Department of Technologies of Computers and Communications, Polytechnic School, University of Extremadura, 10071 Cáceres, Spain
{sanperez,jangomez,mavega}@unex.es

Abstract. Resilient Packet Ring (RPR), also known as IEEE 802.17, is a standard designed for optimising the transport of data traffic over optical fiber ring networks. The Weighted Ring Arc-Loading Problem (WRALP) is a NP-complete problem that arises in engineering and planning of the RPR systems. Specifically, for a given set of non-split and uni-directional point-to-point demands (weights), the objective is to find the routing for each demand (i.e., assignment of the demand to either clockwise or counter-clockwise ring) so that the maximum arc load will be minimised. This paper suggests an efficient traffic loading algorithm- Artificial Bee Colony (ABC). We compare our results with the ones obtained by the standard Genetic Algorithm, Tabu Search Algorithm and Particle Swarm Optimisation, used in literature. Simulation results verify the effectiveness of the ABC algorithm.

Keywords: Optical Networks, Optimisation Algorithms, Artificial Bee Colony, Weighted Ring Arc-Loading Problem.

1 Introduction

The standard IEEE 802.17 for the Resilient Packet Ring (RPR) aims to combine the appealing functionalities from Synchronous Optical Network/Synchronous Digital Hierarchy (SONET/SDH) networks with the advantages of Ethernet networks. The key performance objectives of RPR are to achieve high bandwidth utilization, optimum spatial reuse on the dual rings, and fairness [1]. In RPR systems an optimal load balancing is of paramount importance, as it increases the system capacity and improves the overall ring performance. An important optimisation problem arising in this context is the Weighted Ring Loading Problem (WRLP). Given a network and a set of communications requests, a fundamental problem is to design a transmission routing (direct path) for each request such that the high load on the ring arcs is avoided, being an arc an edge endowed with a direction. The load of an arc is defined as the total weight of those requests that are routed through the Arc in its direction

C. Di Chio et al. (Eds.): EvoApplications 2010, Part II, LNCS 6025, pp. 61–70, 2010.

(WRALP). In general each request is associated with a non-negative integer weight. Practically, the weight of a request can be interpreted as a traffic demand or as the size of the data to be transmitted.

The load balancing problems can be classified into two formulations: with demand splitting (WRLP) or without demand splitting (non-split WRLP). Split loading allows the splitting of a demand into two portions to be carried out in both directions, while in a non-split loading each demand must be entirely carried out in either clockwise or counter-clockwise direction. In either split or non-split cases, WRALP ask for a routing scheme such that maximum load on arcs will be minimum. In this paper we study the WRALP without demand splitting.

Research on the no-split WRLP performed by Cosares and Saniee [2] and Dell'Amico et al. [3] studied the problem on SONET rings. Cosares and Saniee [2] proved that the formulation without demand splitting is a NP-complete problem. For the split problem, various approaches are summarised by Schrijver et al. [4] and their algorithms compared in Myung and Kim [5] and Wang [6].

The non-split WRALP considered in the present paper is identical to the one described by Kubat and Smith [7] (non-split WRALP), Cho et al. [8] (non-split WRALP and WRALP) and Yuan and Zhou [9] (WRALP). They try to find feasible solutions in a reduced amount of time (finding approximate solutions). Our pupose is to compare the performance of our algorithm with others in achieving the best known solution. Bernardino et al. [10] had the same purpose, and present four hybrid Particle Swarm Optimisation (PSO) algorithms to solve this problem.

Swarm intelligence (SI) is a research branch that models the population of interacting agents or swarms that are able to self-organise [11]. Artificial Bee Colony (ABC) simulates the intelligent foraging behaviour of a bee colony [12]. In this paper we propose an ABC algorithm to solve the WRALP. We compare the performance of ABC with the standard Genetic Algorithm (GA), the Tabu Search (TS) algorithm and the Local Search–Probability Binary PSO (LS-PBPSO), used in the literature.

The paper is structured as follows. In Section 2 we describe the WRALP; in Section 3 we describe the proposed ABC algorithm; in Section 4 we present the studied examples and we discuss the computational results obtained and, finally, in Section 5 we report about the conclusions.

2 Problem Definition

Let R_n be a n-node bidirectional ring with nodes $\{n_1, n_2, \ldots, n_n\}$ labelled clockwise. Each edge $\{e_k, e_{k+1}\}$ of R_n, $1 \leq k \leq n$ is taken as two arcs with opposite directions, in which the data streams can be transmitted in either direction: $a_k^+ = (e_k, e_{k+1})$, $a_k^- = (e_{k+1}, e_k)$.

A communication request on R_n is an ordered pair (s,t) of distinct nodes, where s is the source and t is the destination. We assume that data can be transmitted clockwise or counter-clockwise on the ring, without splitting. We use $P^+(s,t)$ to denote the directed (s,t) path clockwise around R_n, and $P^-(s,t)$ the directed (s,t) path counter-clockwise around R_n.

Often a request (s,t) is associated with an integer weight $w>=0$; we denote this weighted request by $(s,t\ ;\ w)$. Let $D= \{(s_1,t_1;\ w_1),(s_2,t_2;\ w_2),\ \ldots,$

$(s_m, t_m; w_m)\}$ be a set of integrally weighted requests on R_n. For each request/pair (s_i, t_i) we need to design a directed path P_i of R_n from s_i to t_i. A set $P=\{P_i: i=1, 2, \ldots, m\}$ of such directed paths is called a routing for D.

3 Proposed ABC

ABC was proposed by Karaboga [12] for optimising numerical problems. The minimal model of swarm-intelligent forage selection in a honey bee colony, that ABC algorithm adopts, consists of three kinds of bees: employed bees, onlooker bees, and scout bees [12-18]. In the ABC algorithm, while onlookers and employed bees carry out, in the search space, the exploitation process, the scout bees control the exploration process [18]. Employed bees are responsible for exploiting the nectar sources and giving information to the other waiting bees (onlooker bees) in the hive about the quality of the food source site which they are exploiting. Onlooker bees wait in the hive and decide a food source to exploit depending on the information shared by the employed bees. Scouts randomly search the environment in order to find a new food source depending on an internal motivation or possible external clues or randomly.

The main steps of the ABC algorithm applied to the WRALP are detailed below:

```
Initialise Parameters
Initialise Employed Bee Colony
Evaluate Employed Bees
WHILE number of iterations < Max number of iterations
  Employed Bees Phase:     FOR each Employed Bee in Colony
                             Apply Local Search Procedure
                             Evaluate Employed Bee
                           Compute Probabilities
  Onlooker Bees Phase:     FOR each Employed Bee in Colony
                             Compute number of Onlooker Bees
                             FOR each Onlooker Bee in Colony
                               Apply Neighbourhood Search
                               Evaluate Onlooker Bee
                               IF fitness(Onlooker)<fitness(Employed Bee)
                                 Replace Employed Bee
  Scout Bees Phase:        Create Scout Bees
                           Replace worst Employed Bees with Scout Bees
  Memorise Best Solution (Bee)
```

Initialise Parameters

The following parameters must be defined by the user: Number of Employed Bees (NE); Number of Onlooker Bees (NO>NE); Maximum number of Iterations (MI) and Number of Attempts (NA).

Initialise Employed Bee Colony

In our implementation, the number of employed bees is exactly the number of solutions in the population. In this work, the solutions are represented using binary vectors (Table 1). We assume that weights cannot be split, that is, for some integer $V_i = 1$, $1 \leq i \leq m$, the total amount of data is transmitted along $P^+(s_i, t_i)$; $V_i=0$, the total amount of data is transmitted along $P^-(s_i, t_i)$. The vector $V=(V_1, V_2, \ldots, V_m)$ determines a routing scheme for D.

Table 1. Solution representation

Pair(s,t) Demand	C-clockwise CC-counterclockwise					
1: (1, 2) → 15	15 C					
2: (1, 3) → 3	3 CC					
3: (1, 4) → 6	6 CC					
4: (2, 3) → 15	15 C					
5: (2, 4) → 6	6 CC					
6: (3, 4) → 14	14 C					
	$Pair_1$	$Pair_2$	$Pair_3$	$Pair_4$	$Pair_5$	$Pair_6$
Representation (V)	1	0	0	1	0	1

The initial solutions can be created randomly or in a deterministic form. Initially a deterministic strategy is followed and in second stage the ABC algorithm is used to optimise the solution. The deterministic form is based in a Shortest-Path Algorithm (SPA). SPA is a simple traffic demand assignment rule in which the demand will traverse the smallest number of segments.

Evaluate Employed Bees

For the evaluation of bees it was necessary to define the following fitness function:

$W_i, \ldots, w_m \rightarrow$ demands of the pairs $(s_i,t_i), \ldots, (s_m,t_m)$ (1a)

$V_i, \ldots, V_m = 0 \rightarrow P^-(s_i,t_i);\ 1 \rightarrow P^+(s_i,t_i)$ (1b)

Load on arcs: $L(V, a_k^+) = \sum_{i:\, a_k^+ \in P^+(s_i,t_i)} wi$ $L(V, a_k^-) = \sum_{i:\, a_k^- \in P^-(s_i,t_i)} wi$ (2a)

$\forall k=1,\ldots,n; \quad \forall i=1,\ldots,m$ (2b)

Fitness function: $\max\{\max L(V, a_k^+), \max L(V, a_k^-)\}$ (3)

The fitness function is based on the following constraints: (1) between each node pair (s_i,t_i) there is a demand value >=0. Each positive demand value is routed in either clockwise (C) or counter-clockwise (CC) direction; (2) for an arc the load is the sum of w_i for clockwise or counter-clockwise between nodes e_k and e_{k+1}.

The objective is to minimise the maximum load on the arcs of a ring (3).

Employed Bees Phase

In ABC an employed bee produces a modification on the position (solution) in her memory depending on the local information (visual information) and tests the nectar amount (fitness value) of the new source (new solution). We use the Local Search (LS) procedure to perform this modification. We create two different LS methods that can be chosen by the user. In the LS "Exchange Direction" some pairs of the solution are selected and their directions are exchanged (partial search). This method can be summarised in the following pseudo-code steps:

```
For t=0 to numberNodesRing/4
  P1 = random (number of pairs)            P2 = random (number of pairs)
  N = neighborhoods of ACTUAL-SOLUTION (one neighborhood results of
interchange the direction of P1 and/or P2)
SOLUTION = FindBest (N)
If ACTUAL-SOLUTION is worst than SOLUTION
  ACTUAL-SOLUTION = SOLUTION
```

In the LS "Exchange Max Arc", first it is necessary to establish the arc with higher fitness. A set of neighbours is obtained by interchanging the direction of some pairs that flow by the arc with higher fitness (partial search). This method can be summarised in the following pseudo-code steps:

```
Define MaxArc                y1= random(m/2)              y2= random(m/2)
FOR p=y1 TO y1+y2
  IF p flows by MaxArc
     N = neighborhoods of ACTUAL-SOLUTION (one neighborhood results of
Interchange the direction of p)
SOLUTION = FindBest (N)
If ACTUAL-SOLUTION is worst than SOLUTION
  ACTUAL-SOLUTION = SOLUTION
```

If the nectar amount of the new solution is higher (with a smaller fitness value) than the nectar of the previous one, the bee memorises the new position and forgets the old one. Otherwise the employed bee keeps the position of the previous one in her memory. After all employed bees complete the search process, they share the information related to the nectar of the food sources and their position with the onlooker bees. An artificial onlooker bee chooses a food source depending on the probability value associated with that food source, pr_i. The probabilities are calculated by: $totalFitness = \sum_{n=1}^{NE} fit_n \quad pr_i = \frac{totalFitness - fit_i}{totalFitness}$.

Onlooker Bees Phase

In this phase each onlooker bee selects a source depending on the quality of her solution, produces a new food source in a selected food source site and exploits the better source. An onlooker bee evaluates the nectar information taken from all employed bees and chooses a food source with a probability related to its nectar amount. Our algorithm computes the number of onlooker bees, which will be sent to food sources of employed bees, according to the previously determined probabilities: NO_i = number of onlooker bees sent to food source i. $NO_i = pr_i * NO$

A neighbour is obtained by performing multiple attempts to improve the solution, whose length is specified as NA (number of attempts). The algorithm performs NA attempts to find a new position for the Onlooker Bee. First the algorithm chooses a pair c randomly. In the 50% of the cases changes its direction to the shortest path; otherwise changes its direction to the direction of the best bee of the population. The algorithm repeats this process until at least one exchange with improvement is made or until the NA is reached. The general mechanism of the neighbourhood search is represented in the next pseudo-code:

```
FOR i=1 TO NA DO
  c=random (m);
  if (random(2)=0)
    if (sc- tc) = n/2
      if (random(2)=1)    testSolution[c] = CC
      else testSolution[c] = C
    else if (sc- tc) > n/2 testSolution[c] = C
         else testSolution[c] = CC
  else testSolution[c]= bestSolution[c];
  if(fitnessTest < fitnessOld) break;
if(fitnessTest < fitnessNew) newSolution=testSolution
```

If the nectar amount of the solution is higher than the nectar amount of the previous one, the bee memorises the new position and forgets the old one.

Scout Bees Phase
In this phase the food source of which the nectar is abandoned by the bees is replaced with a new food source by the scouts. In our implementation, this is simulated by producing new solutions and replacing the worst employed bees. This means that the food sources with lower nectar amounts are abandoned. In our implementation the artificial scout bees can be created randomly or using the SPA.

The worst employed bees as many, as the number of scout bees in the population, are respectively compared with the scout solutions. If a scout bee is better than an employed bee, the employed bee is replaced with the scout bee. Otherwise, the employed bee is transferred to the next cycle without any change. In our implementation we consider the number of scout bees equal to 10 % of the number of employed Bees: $NS = 0.1 * NE$.

Memorise best solution
In this step the algorithm memorises the best solution achieved so far. At the end of execution the best solutions are presented.

Termination criterion
The algorithm stops when a maximum number of iterations (MI) is reached.

More Information about ABC can be found in ABC Website [19].

4 Results

We evaluate the utility of the algorithms using the same examples produced by Bernardino et al. [10]. They consider six different ring sizes - 5, 10, 15, 20, 25 and 30 - and four demand cases: (i) complete set of demands between 5 and 100 with uniform distribution; (ii) half of the demands in (i) set to zero; (iii) 75% of the demands in (i) set to zero; and (iv) complete set of demand between 1 and 500 with uniform distribution. The last case was only used for the 30 nodes ring. For convenience, the instances used are labelled C_{ij}, where 1<i<6 represents the ring size and 1<j<4 represents the demand case.

We perform comparisons between all parameters of the ABC using the instance C41 with 50 iterations and creating the artificial scouts randomly.

We studied the influence of the two different LS methods (used to create new employed bees) on the execution time, the average fitness and the number of best solutions found using a growing number of NA (Fig. 1). The “Exchange Direction” obtains a better average fitness, however the “Exchange Max Arc” is less time consuming. We verify that in the same time, independently of the iterations number, the two LS methods produce an identical number of best solutions.

The best results obtained with ABC use NA between 10 and 15 (Fig. 1). A high NA has a significant impact on the execution time (Fig. 1).

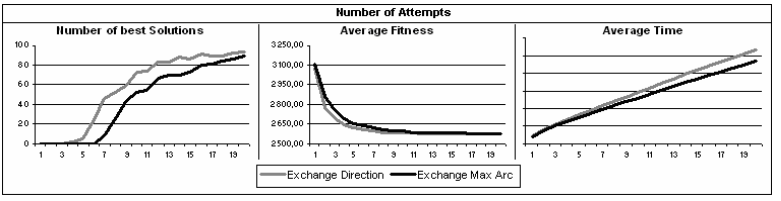


Fig. 1. Number of attempts – number of best solutions/ average fitness/ average time

In our experiments the number of employed bees and the number of onlooker bees were set to {5, 10, 15, 20, 25, 30, 35, 40, 45, 50}. We studied the impact on the execution time, the average fitness and the number of best solutions found. The number of bees has a significant impact on the execution time (Fig. 2 and Fig. 3).

The results show that the best solutions obtained with ABC use NO>NE, being NE between 10 and 20 and NO between 25 and 35. With these values the algorithm can reach in a reasonable amount of time a reasonable number of best solutions. With a higher number of bees the algorithm can reach a better average fitness however it is more time consuming.

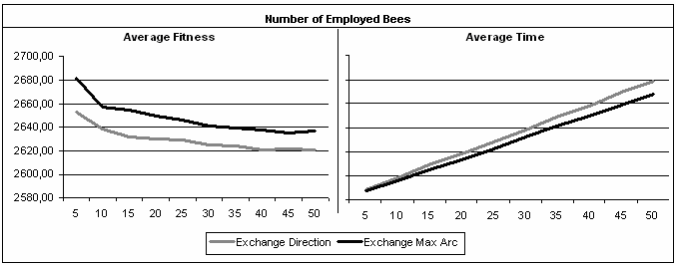


Fig. 2. Number of Employed Bees – Average Fitness/ Execution Time

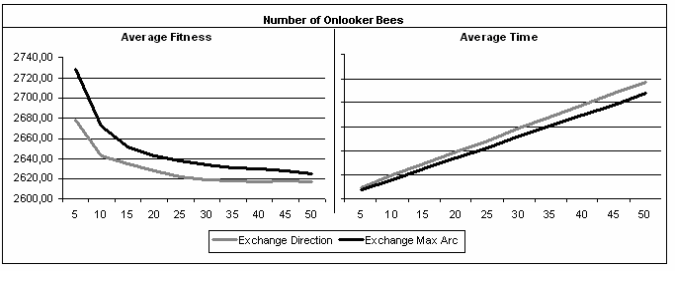


Fig. 3. Number of Onlooker Bees – Average Fitness/ Execution Time

In general, experiments have shown that the proposed parameter setting is very robust to small modifications. Large types of experiments and considerations have been made to define all parameters.

To compare our results we consider the results produced with the standard GA, the TS Algorithm proposed by Bernardino et al. [20], and the LS-PBPSO proposed by Bernardino et al. [10]. The GA is widely used in literature to make comparisons with

other algorithms [21]. The GA adopted uses "one-point" method for recombination, "change direction" method for mutation and "tournament" method for selection. In "change direction", one pair is randomly selected and its direction exchanged. The GA was applied to populations of 200 individuals. Based on preliminary studies we verify that the best solutions obtained by the GA use crossover probability in the intervals [0.6–0.9] and mutation probability in the interval [0.5–0.7]. Suggestions from the literature helped to guide our choice of parameter values for the TS [20] algorithm and LS-PBPSO algorithm [10].

The algorithms have been executed using a processor Intel Quad Core Q9450. The initial solutions of the four algorithms were created using random solutions. For the problem C64 the SPA was used to create the initial populations.

Table 2 presents the best obtained results [12]. The first column represents the instance number (Instance), the second and the third columns demonstrate the number of nodes (Nodes) and the number of pairs (Pairs) and finally the fourth column demonstrates the minimum fitness values obtained (best known solution).

Table 2. Best obtained results

Instance	Nodes	Pairs	Best Fitness	Instance	Nodes	Pairs	Best Fitness
C11	5	10	161	C41	20	190	2581
C12	5	8	116	C42	20	93	1482
C13	5	6	116	C43	20	40	612
C21	10	45	525	C51	25	300	4265
C22	10	23	243	C52	25	150	2323
C23	10	12	141	C53	25	61	912
C31	15	105	1574	C61	30	435	5762
C32	15	50	941	C62	30	201	2696
C33	15	25	563	C63	30	92	1453
				C64	30	435	27779

Table 3. Results – run times and number of iterations

Prob	Number	GA		Tabu Search		LS-PBPSO		ABC	
	Iterations	Time	IT	Time	IT	Time	IT	Time	IT
C11	25	<0.001	2	<0.001	5	<0.001	2	<0.001	2
C12	10	<0.001	2	<0.001	5	<0.001	2	<0.001	2
C13	10	<0.001	1	<0.001	1	<0.001	1	<0.001	1
C21	50	<0.001	25	<0.001	25	<0.001	15	<0.001	10
C22	25	<0.001	5	<0.001	5	<0.001	3	<0.001	3
C23	10	<0.001	3	<0.001	5	<0.001	3	<0.001	3
C31	100	0. 1	40	0. 1	90	0. 1	20	0. 1	15
C32	50	<0.001	15	<0.001	30	<0.001	8	<0.001	5
C33	25	<0.001	5	<0.001	20	<0.001	5	<0.001	3
C41	300	0.15	100	0.2	220	0.1	50	0.1	20
C42	100	0.075	35	0.1	85	0.05	20	0.05	10
C43	50	0.001	10	0.001	25	0.001	5	0.001	3
C51	500	0.8	150	1	260	0.75	80	0.75	50
C52	400	0.15	50	0.2	110	0.1	25	0.1	15
C53	250	0.02	35	0.03	200	0.01	15	0.01	10
C61	1500	2.3	300	3.5	400	2	130	1.75	80
C62	1000	0.6	120	0.8	230	0.4	50	0.3	30
C63	500	0.08	30	0.08	100	0.075	15	0.075	10
C64	500	0.5	5	1.5	250	0.75	40	0.3	5

Table 3 presents the best results obtained with GA, TS, LS-PBPSO and ABC. The first column represents the problem number (Prob), the second column demonstrates the maximum number of iterations used to test each instance and the remaining columns show the results obtained (Time – Run Times, IT - Iterations) by the four algorithms. The results have been computed based on `100` different executions for each test instance using the best combination of parameters found and different seeds. All the algorithms reach the best solutions before the run times and number of iterations presented.

Table 4 presents the average fitness and the average time obtained with GA, TS, LS-PBPSO and ABC using a limited number of iterations for the problems C41, C51 and C61 (harder problems). The first column represents the number of the problem (Prob), the second column demonstrates the number of iterations used to test each instance and the remaining columns show the results obtained (AvgF – Average Fitness, AvgT – Average Time) by the four algorithms. The results have been computed based on `100` different executions for each test instance using the best combination of parameters found and different seeds.

Table 4. Results – Average Time / Average Fitness

Problem	Number of iterations	GA		LS-PBPSO		Tabu		ABC	
		AvgF	AvgT	AvgF	AvgT	AvgF	AvgT	AvgF	AvgT
C41	50	2597,67	0,10	2594,36	0,26	2635,28	0,16	2581,5	0,12
C51	75	4298,23	0,43	4291,52	0,86	4392,70	0,86	4265,85	0,77
C61	100	5848,54	1,34	5837,58	3,10	5963,14	3,71	5762,22	1,86

ABC and LS-PBPSO obtain a better average fitness for larger instances. TS is the slowest algorithm and obtains a smaller number of best solutions. The ABC is the algorithm that presents the best average fitness since in almost executions it reaches the best solution. When using the SPA for creating the initial solutions, the times and number of iterations decreases – problem `C64`. This problem is computationally harder than the `C61`, however the best solution is obtained faster. To improve the solutions we consider more efficient to apply initially a SPA and then the metaheuristic to improve the solutions.

5 Conclusions

In this paper we present an ABC algorithm to solve the non-split WRALP. The performance of' our algorithm is compared with three algorithms: a classical GA, a TS algorithm and a LS-PBPSO algorithm.

Experimental results demonstrate that the proposed ABC algorithm is an effective and competitive approach in composing fairly satisfactory results with respect to solution quality and execution time for the WRALP.

The ABC presents better results for larger problems. Our algorithm provides higher number of best solutions for larger problems. When using SPA for creating the initial solutions or for creating the scout bees the best solution is obtained faster.

In literature the application of ABC for this problem is nonexistent, for that reason this article shows its enforceability in the resolution of this problem.

The continuation of this work will be the search and implementation of new methods for speeding up the optimisation process.

References

1. RPR Alliance: A Summary and Overview of the IEEE 802.17 Resilient Packet Ring Standard (2004)
2. Cosares, S., Saniee, I.: An optimization problem related to balancing loads on SONET rings. Telecommunication Systems 3(2), 165–181 (1994)
3. Dell'Amico, M., Labbé, M., Maffioli, F.: Exact solution of the SONET Ring Loading Problem. Oper. Res. Lett. 25(3), 119–129 (1999)
4. Schrijver, A., Seymour, P., Winkler, P.: The ring loading problem. SIAM Journal of Discrete Mathematics 11, 1–14 (1998)
5. Myung, Y.S., Kim, H.G.: On the ring loading problem with demand splitting. Operations Research Letters 32(2), 167–173 (2004)
6. Wang, B.F.: Linear time algorithms for the ring loading problem with demand splitting. Journal of Algorithms 54(1), 45–57 (2005)
7. Kubat, P., Smith, J.M.: Balancing traffic flows in resilient packet rings. In: Girard, A., et al. (eds.) Performance evaluation and planning methods for the next generation internet, vol. 6, pp. 125–140. Springer, Heidelberg (2005), GERAD 25th Anniversary
8. Cho, K.S., Joo, U.G., Lee, H.S., Kim, B.T., Lee, W.D.: Efficient Load Balancing Algorithms for a Resilient Packet Ring. ETRI Journal 27(1), 110–113 (2005)
9. Yuan, J., Zhou, S.: Polynomial Time Solvability of the Weighted Ring Arc-Loading Problem With Integer Splitting. Journal of Interconnection Networks 5(2), 193–200 (2004)
10. Bernardino, A.M., Bernardino, E.M., Sánchez-Pérez, J.M., Vega-Rodríguez, M.A., Gómez-Pulido, J.A.: Solving the non-split weighted ring arc-loading problem in a Resilient Packet Ring using Particle Swarm Optimization. In: International Conference in Evolutionary Computation (2009)
11. Kennedy, J., Eberhart, R.C., Shi, Y.: Swarm Intelligence. Morgan Kaufmann, San Francisco (2001)
12. Karaboga, D.: An idea based on honey bee swarm for numerical optimization. Technical Report TR06, Erciyes University, Engineering Faculty, Computer Engineering Department (2005)
13. Basturk, B., Karaboga, D.: An artificial bee colony (ABC) algorithm for numeric function optimization. In: IEEE Swarm Intelligence Symposium (2006)
14. Karaboga, D., Basturk, B.: A powerful and efficient algorithm for numerical function optimization: Artificial bee colony (ABC) algorithm. Journal of Global Optimization 39(3), 459–471 (2007)
15. Karaboga, D., Basturk, B.: Artificial Bee Colony (ABC) Optimization Algorithm for Solving Constrained Optimization Problems. In: Melin, P., Castillo, O., Aguilar, L.T., Kacprzyk, J., Pedrycz, W. (eds.) IFSA 2007. LNCS (LNAI), vol. 4529, pp. 789–798. Springer, Heidelberg (2007)
16. Karaboga, D., Basturk, B.: On the performance of artificial bee colony (ABC) algorithm. Applied Soft Computing 8(1), 687–697 (2008)
17. Karaboga, D., Akay, B.: A comparative study of artificial bee colony algorithm. Applied Mathematics and Computation (2009)
18. Karaboga, D., Akay, B.: Artificial Bee Colony (ABC), Harmony Search and Bees Algorithms on Numerical Optimization. In: Innovative Production Machines and Systems Virtual Conference (2009)
19. Artificial Bee Colony Algorithm Website, `http://mf.erciyes.edu.tr/abc/`
20. Bernardino, A.M., Bernardino, E.M., Sánchez-Pérez, J.M., Vega-Rodríguez, M.A., Gómez-Pulido, J.A.: Solving the weighted ring edge-loading problem without demand splitting using a Hybrid Differential Evolution Algorithm. In: The 34th IEEE Conference on Local Computer Networks. IEEE Press, Los Alamitos (2009)
21. Eiben, A.E., Smith, J.E.: Introduction to Evolutionary Computing. Springer, Berlin (2003)

TCP Modification Robust to Packet Reordering in Ant Routing Networks

Malgorzata Gadomska-Kudelska and Andrzej Pacut

Institute of Control and Computation Engineering, Warsaw Univ. of Technology
Nowowiejska 15/19, 00-665 Warsaw, Poland
m.gadomska@elka.pw.edu.pl, A.Pacut@ia.pw.edu.pl

Abstract. In this paper a modification of the TCP protocol is proposed that improves its robustness to packet reordering in networks controlled by ant routing algorithms. In our approach the TCP sender builds models of packet end-to-end delay and every time a DUPACK is received it utilizes these models to calculate the probability that the packet will still arrive at the receiver. Based on this probability the decision is made whether or not to retransmit the packet. The advantages of our approach are proved in a set of simulations.

1 Introduction

The design of TCPs error and congestion control mechanisms assumes that a packet loss is an indication of a network congestion. Therefore, when a loss is detected, the TCP sender reduces the transmission rate by decreasing the congestion window, which constraints the number of packets that can be sent. The TCP uses two strategies for detecting the packet loss. The first one is based on the senders retransmission time-out (RTO) expiration. The second mechanism originates at the receiver, which monitors the sequence numbers of packets it receives and generates the duplicate acknowledgment (DUPACK) for every out-of-order packet. When the sender receives a few DUPACKs (usually 3), the fast retransmit algorithm infers that a packet has been lost and the packet is retransmitted without waiting for a time-out. The basic idea behind the fast retransmit is to improve TCPs throughput by avoiding the senders time-out. This can improve TCPs performance when occasional reordering occurs in the network but it operates under the assumption that out-of-order packets indicate packet loss and therefore a congestion.

However, some networks may experience packet reordering during their normal operation. Examples of such situations may include: multipath routing, multi-hop mobile ad-hoc networks routing and QoS provisioning. In multipath routing algorithms, the packets belonging to the same TCP session can be sent along different routes. Under such conditions, the packets sent earlier can reach their destination after the packets sent later, but along a faster path. Consequently, while the packet reordering is not then a sign of abnormality, yet it is interpreted as such by the TCP. Such an effect can be also observed for wireless

C. Di Chio et al. (Eds.): EvoApplications 2010, Part II, LNCS 6025, pp. 71–80, 2010.

networks and multi-hop mobile ad-hoc networks in particular, where the mobility of nodes causes frequent changes in the network topology. Some mechanisms that control quality-of-service (QoS) by differentiating the traffic are also likely to introduce packet reordering in their normal operation. The above examples show that there is a need to modify the TCP protocol to make it more robust to frequent packet reordering.

In this paper, we propose a modification of the TCP protocol that improves its robustness to packet reordering in networks controlled by ant routing algorithms. Ant routing algorithms are an example of adaptive multipath algorithms in which the path is selected according to a stochastic policy, where the probability of choosing a particular path is proportional to a quality of this path. They do not require supervision, and their distributed form makes them well applicable to the routing problem. Ant routing algorithms are typically considered with the UDP in the transport layer. However, to be useful in a real Internet environment, such routing algorithms should perform well also with the TCP, since most applications use the TCP as a transport layer protocol to ensure a reliable transmission.

1.1 Swarm Intelligence and Ant Routing

Swarm Intelligence is an evolving, collective intelligence of groups of autonomous agents. The agents follow some simple rules, interacting with one another and with their environment. As a result, they are capable of solving complex and distributed problems.

Basing on the Swarm Intelligence paradigm and its derivative, the Ant Colony Optimization scheme [8], there were various ant routing algorithms proposed for both fixed and wireless telecommunication networks.

The first ant routing algorithm for symmetric circuit-switched networks was proposed in 1996 by Schoonderwoerd [13]. In 1998, Dorigo and Di Caro introduced AntNet [6], the first ant routing algorithm designed for asymmetric packet-switched networks. The algorithm implicitly achieves load balancing by distribution of packets over multiple paths. The experiments reported in [6] proved that AntNet outperforms other competitors, such as Q-routing [4] and PQ-routing [5] based on the Reinforcement Learning paradigm, Shortest Path First (SPF) and Open Shortest Path First (OSPF) [9].

Various modifications of AntNet have been developed that address modifications of the routing probabilities and the traffic model in order to achieve a better performance, improvement of the algorithms convergence properties, and exploration techniques [11]. In this work, we use a modification of the ASR algorithm proposed in [15] and analyzed under various network conditions in [10,12].

1.2 The TCP Protocol

When using the TCP in the transport layer, the network dynamics is mainly determined by TCP's error and congestion control mechanisms [14], which employ

the *slow start* and *congestion avoidance* algorithms, or the duplicate acknowledgment (DUPACK) mechanism and the fast retransmit algorithm.

Therefore, it is commonly believed that ant routing algorithms cannot be applied with the TCP in the transport layer as these mechanisms slow down the ant algorithms to such degree that they may become useless for the routing problem. In [10] it is shown that, contrary to this belief, the use of TCP is not prohibitive to multipath routing, including the ant algorithms. While the TCP sets higher demands on the adaptation processes, it is still possible to extend the load range of the network.

Recently, several mechanisms have been proposed to improve the performance of TCP when using multipath routing algorithms. However, most of them help only when the reordering occurs occasionally, since they are based on detecting spurious retransmissions. On the basis of the DUPACKs analysis, statistics of reordering frequencies are being computed [16]. Then the TCP receiver adaptively adjusts the threshold, which decides after how many DUPACKs the packet should be retransmitted [2,16].

Another approach [3] proposes to neglect DUPACKs altogether and rely solely on the timers to detect drops: if the ACK for a packet has not arrived and the time elapsing since the packet was sent exceeds a threshold, then the packet is assumed to be lost. The retransmission threshold should adapt to the changing conditions in the network in such way that the packet is retransmitted only if it has been really lost.

In our approach, every time the TCP sender receives a DUPACK, it calculates the probability that the packet in question may still arrive at the receiver. On the base of this probability, a decision is made whether to retransmit the packet.

2 The TCP Modification

In this section we present the Delay Model Based TCP (DelModAntTCP), which is a modification of the TCP protocol robust to packet reordering. The modification applies to the TCP sender and it utilizes models of the data packets' delay distributions. The method of the delay distribution modeling is described first and then the modified TCP protocol is introduced.

2.1 Packet Delay Modeling

We propose to model the empirical distribution of packet end-to-end delay in a network controlled by adaptive routing algorithms with a mixture of probability distributions. To model the empirical delay distribution of one path, we introduce the gamma-exponential-delta mixture model, denoted by $\Gamma\mathrm{Ex}\delta$, which is a mixture of three probability distributions: namely, the gamma distribution, the exponential distribution, and the single point distribution, all delayed by a constant time. We use the notation

$$f_\Gamma\left(x|\theta_\Gamma\right) = \begin{cases} \frac{\lambda_\Gamma^{\nu_\Gamma}}{\Gamma(\nu_\Gamma)} x^{\nu_\Gamma - 1} e^{-\lambda_\Gamma x} & x \geq 0 \\ 0 & x < 0 \end{cases} \tag{1}$$

for the gamma distribution density with the rate parameter $\lambda_\Gamma > 0$ and the shape parameter $\nu_\Gamma > 0$, where $\Gamma(\nu_\Gamma) = \int_0^\infty z^{\nu_\Gamma - 1} e^{-z} dz$ is the gamma function. Further we use

$$f_{\mathrm{Ex}}(x|\theta_{\mathrm{Ex}}) = \begin{cases} \lambda_{\mathrm{Ex}} e^{-\lambda_{\mathrm{Ex}} x} & x \geq 0 \\ 0 & x < 0 \end{cases} \tag{2}$$

for the exponential distribution density with the rate parameter $\lambda_{\mathrm{Ex}} > 0$ and

$$f_\delta(x) = \delta(x) \tag{3}$$

for the formal description of the single point distribution. We sometimes group the parameters of each distribution, denoting then $\theta_\Gamma = (\lambda_\Gamma, \nu_\Gamma)$ and $\theta_{\mathrm{Ex}} = (\lambda_{\mathrm{Ex}})$.

The empirical distribution of the multi-path end-to-end packet delay from a source to a destination node consists of several peaks, corresponding to different paths the packet can take. Therefore we model the packet delay distribution as a mixture of $\Gamma\mathrm{Ex}\delta$ triplets, $i = 1, \ldots, M,\ t = 1, \ldots, N$, namely

$$f(x_t|\theta) = \sum_{i=1}^{M} \left(\pi_{\Gamma,i}\, f_\Gamma(x_t - s_i|\theta_{\Gamma,i}) + \pi_{\mathrm{Ex},i}\, f_{\mathrm{Ex}}(x_t - s_i|\theta_{\mathrm{Ex},i}) + \pi_{\delta,i}\, f_\delta(x_t - s_i)\right) \tag{4}$$

where M is the number of possible paths from the source to the destination, N is the data sample size, $\theta = \{(\pi_{\Gamma,i}, \pi_{\mathrm{Ex},i}, \pi_{\delta,i}, \theta_{\Gamma,i}, \theta_{\mathrm{Ex},i}, s_i), i = 1, \ldots, M\}$ is the parameter vector and s_i reflects the minimum delay for a given path, which depends on the link delay and bandwidth. The mixing parameters satisfy the following constraints $\pi_{\Gamma,i} \geq 0,\ \pi_{\mathrm{Ex},i} \geq 0,\ \pi_{\delta,i} \geq 0,\ \sum_{i=1}^{M} (\pi_{\Gamma,i} + \pi_{\mathrm{Ex},i} + \pi_{\delta,i}) = 1$.

To introduce the estimation method for the $\Gamma\mathrm{Ex}\delta$ model, note that it is of discrete-continuous type (the density has the delta term), and, moreover, it contains an unknown delay. It is easy to notice that the resulting likelihood function is not differentiable with respect to the delay, hence its basic properties are not fulfilled, and the typical estimation procedures may behave erratically. To overcome this problem, we propose a two-stage estimation procedure:

1. *Elimination of the discrete-type distribution by the estimation of the delay.*
2. *Estimation of the elements of the mixture of continuous distributions.*

In stage 1, we eliminate the discrete part of the distribution (the delta peaks) together with the delays. In this order, we calculate the empirical cumulative distribution function (ECDF) with a given bin width. We set a $\Gamma\mathrm{Ex}\delta$ model delay at the position of every ECDF jump. In the second stage, we use the Expectation Maximization (EM) algorithm to estimate the parameters of the mixture model. The EM algorithm provides an efficient iterative procedure to compute the Maximum Likelihood (ML) estimates in the presence of missing or unobservable data. It iterates two steps: in the expectation step (E-step) the distribution of the unobservable variable is estimated and in the maximization step (M-step) the parameters which maximize the expected log likelihood found on the E-step are calculated.

The parameters of the gamma components are estimated similarly to the way proposed in [1]. The difference is that we use robust parameter estimation [7]. We weigh each data point in such way that the influence of this observation on the value of the estimated distributions parameters decays with the distance from the distributions mean. Consequently, the EM algorithm is more robust to the noise and small insignificant peaks that we do not want to model, as they may turn out to bias parameter estimates of nearby peaks (details in [7]).

For each possible path $i = 1, 2, \ldots, M$, the conditional probability densities for the gamma components estimated in E-step are calculated as

$$p_{\Gamma,i}(x_t, \theta^k) = \frac{\pi^k_{\Gamma,i}\, f_\Gamma(x_t - s_i|\theta^k_{\Gamma,i})}{\sum_{j=1}^{M} \left(\pi^k_{\Gamma,j}\, f_\Gamma(x_t - s_i|\theta^k_{\Gamma,j}) + \pi^k_{\mathrm{Ex},j}\, f_{\mathrm{Ex}}(x_t - s_i|\theta^k_{\mathrm{Ex},j})\right)} \tag{5}$$

and for the exponential components as

$$p_{\mathrm{Ex},i}(x_t, \theta^k) = \frac{\pi^k_{\mathrm{Ex},i}\, f_{\mathrm{Ex}}(x_t - s_i|\theta^k_{\mathrm{Ex},i})}{\sum_{j=1}^{M} \left(\pi^k_{\Gamma,j}\, f_\Gamma(x_t - s_i|\theta^k_{\Gamma,j}) + \pi^k_{\mathrm{Ex},j}\, f_{\mathrm{Ex}}(x_t - s_i|\theta^k_{\mathrm{Ex},j})\right)} \tag{6}$$

The robust parameter estimates are calculated in the M-step as follows:

$$\pi^{k+1}_{\Gamma,i} = \frac{1}{N} \sum_{t=1}^{N} p_{\Gamma,i}(x_t, \theta^k) \tag{7}$$

$$\pi^{k+1}_{\mathrm{Ex},i} = \frac{1}{N} \sum_{t=1}^{N} p_{\mathrm{Ex},i}(x_t, \theta^k) \tag{8}$$

$$\lambda^{k+1}_{\Gamma,i} = \frac{\nu^k_{\Gamma,i} \sum_{t=1}^{N} w_{\Gamma,it}\, p_{\Gamma,i}(x_t, \theta^k)}{\sum_{t=1}^{N} w_{\Gamma,it}\, (x_t - s^k_i) p_{\Gamma,i}(x_t, \theta^k)} \tag{9}$$

$$\lambda^{k+1}_{\mathrm{Ex},i} = \frac{\sum_{t=1}^{N} w_{\mathrm{Ex},it}\, p_{\mathrm{Ex},i}(x_t, \theta^k)}{\sum_{t=1}^{N} w_{\mathrm{Ex},it}\, (x_t - s^k_i) p_{\mathrm{Ex},i}(x_t, \theta^k)} \tag{10}$$

$$\nu^{k+1}_{\Gamma,i} = \nu^k_{\Gamma,i} + a_k G^a_{\nu_{\Gamma,i}}(X, \theta^k) \tag{11}$$

The parameters w_{it} weigh the data points using the Mahalanobis distance $d_{it} = |x_t - \mu_i|/\sigma_i$, for $i = 1, \ldots, M$ (see [7] for details).

2.2 Delay Model Based TCP

The models described in Sec. 2.1 are calculated on-line in every node during the networks operation, on the base of information gathered by ants. At a given time interval, a node starts to collect information about the estimated data packets' delays, the number of hops traveled and the corresponding neighbor node. The information is obtained from every backward ant traveling to a given destination. After collecting a sample of N observations, the EM algorithm is performed as described in Sec. 2.1.

As a result, every node s in the network maintains a delay model to every possible destination node d. Such the delay model consists of several $\Gamma\mathrm{Ex}\delta$ triplets, corresponding to paths from s to d. It is worth noticing that different paths with

the same minimum delay are modeled by one ΓExδ triplet and, from the model point of view, we refer to them as to a single path (Sec. 2.1). Here, we denote such set of paths by i.

On the base of the data sample used to build the model, we assign a weight to each ΓExδ triplet of the mixture. This weight expresses the probability that a particular neighbor node is most likely responsible for generating this part of the delay distribution. Thank to this, it is possible to calculate delay models from every source s to every destination d through every neighbor n.

The weights are calculated as follows. For every neighbor node n and every set of paths i described by a ΓExδ triplet, the average probability is calculated

$$W_{i,n} = \frac{1}{N_n} \sum_{x_t=1}^{N_n} \pi_{\Gamma,i}\, f_\Gamma\left(x_t - s_i | \theta_{\Gamma,i}\right) + \pi_{\mathrm{Ex},i}\, f_{\mathrm{Ex}}\left(x_t - s_i | \theta_{\mathrm{Ex},i}\right) + \pi_{\delta,i}\, f_\delta\left(x_t - s_i\right) \quad (12)$$

where N_n denotes the number of packets from the data sample that traveled through the neighbor n. These probabilities are then used as weights so they are normalized for each i to obtain $P(n|i) = W_{i,n} / \sum_{n=1}^{K} W_{i,n}$, which denotes the probability that the packet traveled via neighbor n, provided that it traveled along a path from set i. K is the number of neighbor nodes.

All paths from set i are modeled by one ΓExδ triplet, which means that we have made an assumption that all paths with the same minimum delay have the same delay distribution. As a result, the delay distribution of packets that traveled along a path from set i is conditionally independent from the neighbor n provided that we know i. Therefore, the joint probability that the delay on that path is shorter than γ and the path leads through neighbor n provided that the path comes from set i is $P(x_t < \gamma, n|i) = P(x_t < \gamma|i)\, P(n|i)$.

Thus, the joint probability that the delay was shorter than γ and the packet traveled through neighbor n can be calculated as

$$P(x_t < \gamma, n) = \sum_{i=1}^{M} P(x_t < \gamma, n|i)\, \pi_i = \sum_{i=1}^{M} P(x_t < \gamma|i)\, \pi_i\, P(n|i) \quad (13)$$

where $P(x_t < \gamma|i)\, \pi_i = \pi_{\Gamma,i}\, p_\Gamma\left(x_t - s_i < \gamma | \theta_{\Gamma,i}\right) + \pi_{\mathrm{Ex},i}\, p_{\mathrm{Ex}}\left(x_t - s_i < \gamma | \theta_{\mathrm{Ex},i}\right) + \pi_{\delta,i}\, p_\delta\left(x_t - s_i < \gamma\right)$ and $\pi_i = \pi_{\Gamma,i} + \pi_{\mathrm{Ex},i} + \pi_{\delta,i}$ is the share of the i-th ΓExδ triplet in the mixture model.

Once the delay models from every source s to every destination d through every neighbor n are known, they can be employed in retransmission decision making in the following way. When a TCP sender receives 3 DUPACKs, it calculates the probability that the missing packet may still arrive at its destination node d provided that it traveled through a neighbor node n, namely

$$P(x_t \geq \gamma|n) = 1 - P(x_t < \gamma|n) = 1 - \frac{P(x_t < \gamma, n)}{P(n)} \quad (14)$$

where the delay x_t is calculated as the difference between the time the DUPACK was generated at the destination node and the time the missing packet was sent from the source node. If the probability $P(x_t \geq \gamma|n)$ is greater than a specified threshold, the missing packet is not retransmitted and this probability will be

recalculated when the next DUPACK arrives. Thank to this approach, it is possible to judge whether the DUPACK was received due to packet reordering.

3 Experimental Results

We performed our experiments using the NS2 network simulator with additional custom made modules. We present the results for the "Simple" network structure (Fig. 1, left). The parameters of the ant routing algorithm ASR were set in such way to obtain non-zero probabilities of choosing every neighbor of Node 0, as we wanted packet reordering to occur quite often. Figure 1 shows the empirical distribution and the $\Gamma\text{Ex}\delta$ mixture model of the packet delay between node 0 and Node 4. Each peak of the distribution density represents the delay distribution along the given path group (Fig. 1, middle). For each path group, we fit three component peaks: the delta peak that corresponds to the delay of packets which did not wait in any queue, the gamma peak to approximate the small and medium delays and the exponential peak to approximate the tail of the delay distribution. It can be seen that the model fits the data very well.

The performance of the DelModAnt TCP was compared to TCP Tahoe performance in two simulation scenarios: with and without packet loss introduced in the network. The figures presented in the next sections show the averages over 20 simulations.

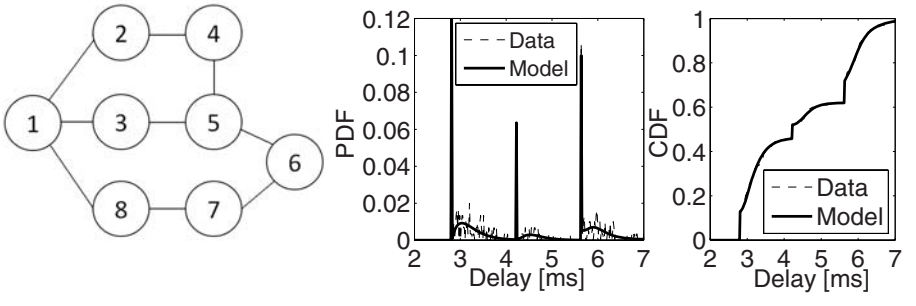


Fig. 1. "Simple" network structure (left), end-to-end delay distribution from Node 1 to Node 4 and the mixture model of this delay distribution (distribution density, middle and cumulative density, right)

3.1 Network without Packet Loss

We tested the differences between the influence of the TCP Tahoe and the DelModAnt TCP on the ant-routing performance. During the experiment, packets are not being lost, hence all duplicate acknowledgments are the result of packet reordering.

When using the TCP Tahoe under high load levels, the ASR algorithm does not manage to find efficient routing policies. It can bee seen that the number of data bytes in the network increases during the simulation and does not settle (Fig. 2,

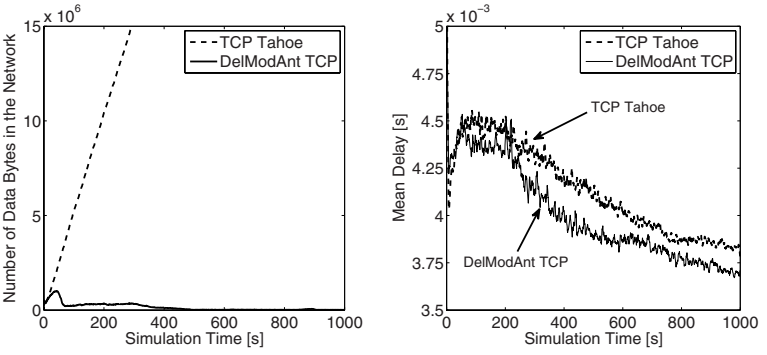


Fig. 2. Influence of the TCP version on the ASR algorithm under high load level. Whereas for TCP Tahoe the simulations diverge (the Number of Data Bytes in the Network grows with Simulation Time), using the DelModAnt TCP makes the network stable.

Table 1. Influence of the TCP version on the Number of Retransmissions. Using the DelModAnt TCP enables reducing the number of retransmissions caused by DUPACKs by about 40% and the overall number of retransmissions by about 30%.

	DelModAnt TCP	TCP Tahoe
Number of Retransmissions	95800	143000
Number of DUPACK Retransmissions	29700	50500

left), although the mean packet delay seems to converge (Fig. 2, right). We calculate the packet delay from the moment the packet is first sent by the TCP agent, so it does not include the time spent in the input buffer of the TCP agent. When using the TCP Tahoe together with the ASR under high load level, many retransmissions occur, which cause the transmission window to decrease. As a result, the queue containing data waiting to be sent increases during the simulation.

Under the same load, the DelModAnt TCP ensured the convergence of the learning process (Fig. 2). When using DelModAnt TCP there are less retransmissions in the network. Consequently, the load level is reduced and the ASR algorithm manages to find efficient routing policies. Our experiments show that the DelModAnt TCP can reduce the number of retransmissions in the network caused by DUPACKS by a significant factor (Tab. 1). Apart from retransmissions caused by DUPACKs, there is also a second source of retransmissions, based on the senders retransmission time-out (RTO) expiration. Therefore, the overall reduction of retransmissions is slightly smaller, but still relevant.

3.2 Network with Packet Loss

In this Section, we test the influence of the TCP version on the ASR algorithm's performance in a network with packet loss. The TCP, unlike the UDP, is a connection oriented protocol that guarantees reliable data transmission.

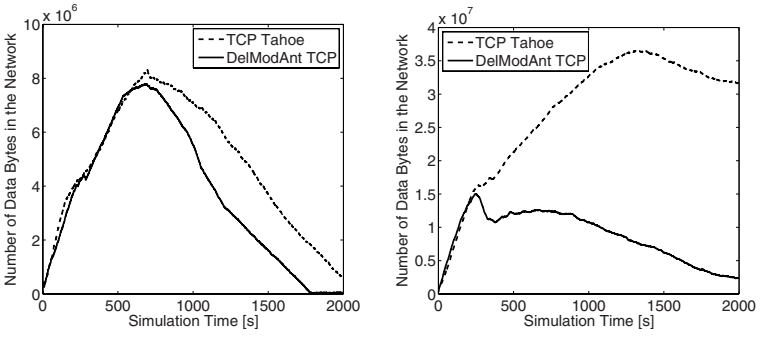


Fig. 3. Influence of the TCP version on the ASR algorithm in a network with packet loss, loss probability: 0.0055 (left) and 0.01 (right). Under lower packet loss ratio using DelModAnt TCP enables faster convergence of the ASR algorithm (left). When the loss ratio increases for TCP Tahoe the simulations diverge (right).

Therefore, if a packet gets lost during transmission it will be retransmitted by the TCP sender. We assumed that each router, independently of other, may loose a packet with a certain probability.

Since every lost packet must be retransmitted, the actual load level increases. In such situation it is especially important to distinguish the reason of a DUPACK being sent in order to decide whether to retransmit a packet.

It can be seen that under the loss probability equal to 0.055 the both TCP versions ensure convergence of the ASR routing algorithm (Fig. 3, left). However, using the DelModAnt TCP results in faster convergence. If the loss probability increases to 0.01, which means that 1% of packets get lost, the simulations for TCP Tahoe diverge (Fig. 3, right). On the base of the delay models, the DelModAnt TCP is able to distinguish a DUPACK received because of packet reordering from a DUPACK received as an effect of packet loss. As a result, using the modified TCP sender decreases the number of needless retransmissions caused by packet reordering. Consequently, the load level in the network is reduced and the ASR algorithm is able to route the packets effectively.

4 Conclusions

In this paper we propose the DelModAnt TCP, which is a modification of the TCP protocol that improves its robustness to packet reordering in networks controlled by ant routing algorithms. The modification applies to the TCP sender and utilizes models of the data packets' delay distributions in order to decide whether a packet has been lost and must be retransmitted or the DUPACK was the result of packet reordering.

Our experiments show that the modified TCP can significantly reduce the number of retransmissions in the network. Moreover, using the DelModAnt TCP extends the range of load levels under which the ant algorithms are able to find efficient routing policies.

It is worth noticing that the use of delay models to improve the performance of TCP is not limited to reducing the number of retransmissions caused by DUPACKS. These models could be also used to compute a modified retransmission timeout, which would depend on the packets path.

Acknowledgments

The scientific work funded as a research project by finance for scientific research in the years 2009-2010.

References

1. Almhana, J., Liu, Z., Choulakian, V., McGorman, R.: A Recursive algorithm for Gamma Mixture Model. In: Proceedings of IEEE ICC 2006, pp. 197–202 (2006)
2. Blanton, E., Allman, M.: On making TCP more robust to packet reordering. ACM Computer Communications Review, 20–30 (2002)
3. Bohacek, S., Hespanha, J.P., Lee, J., Lim, C., Obraczka, K.: TCP-PR: TCP for Persistent Packet Reordering, Tech. Rep., Univ. of California Santa Barbara (2003)
4. Boyan, J.A., Littman, M.L.: Packet Routing in Dynamically Changing Networks: A Reinforcement Learning Approach. Advances in Neural Information Processing Systems 6, 671–678 (1994)
5. Choi, P., Yeung, D.: Predictive q-routing: a memory-based reinforcement learning approach to adaptive traffic control. Advances in Neural Information Processing Systems 8, 945–951 (1996)
6. Di Caro, G., Dorigo, M.: Two ant colony algorithms for best-effort routing in datagram networks. In: Proceedings of the Tenth IASTED International Conference PDCS 1998, pp. 541–546. IASTED/ACTA Press (1998)
7. Dijkstra, M., Roelofsen, H., Vonk, R.J., Jansen, R.C.: Peak quantification in surface-enhanced laser desorption/ionization by using mixture models. Proteomics 6, 5106–5116 (2006)
8. Dorigo, M., Maniezzo, V., Colorni, A.: Positive feedback as a search strategy, Tech. Rep. 91-016, Politecnico di Milano, Dipartimento di Elettronica (1991)
9. Doyle, J.: Routing TCP/IP, Vol. I and II, CCIE Professional Development (1998)
10. Gadomska, M., Pacut, A.: Performance of Ant Routing Algorithms When Using TCP. In: Giacobini, M. (ed.) EvoWorkshops 2007. LNCS, vol. 4448, pp. 1–10. Springer, Heidelberg (2007)
11. Gadomska, M., Pacut, A.: Recent Progress in Ant Algorithms for Fixed Telecommunication Networks: A Review. In: Evolutionary Computation and Global Optimization 2007, Prace Naukowe Politechniki Warszawskiej, Elektronika z, vol. 160, pp. 79–89 (2007)
12. Pacut, A., Gadomska, M., Igielski, A.: Ant-Routing vs. Q-Routing in Telecommunication Networks. In: Proceedings of the 20th ECMS Conference, pp. 67–72 (2006)
13. Schoonderwoerd, R., Holland, O., Bruten, J., Rothkrantz, L.: Ant-based load balancing in telecommunications networks. Adaptive Behavior 5(2), 169–207 (1996)
14. Stevens, W.R.: TCP/IP Illustrated, vol. I, II. Addison-Wesley, Reading (1994)
15. Yong, L., Guang-zhou, Z., Fan-jun, S.: Adaptive swarm-based routing in communication networks. Journal of Zhejiang Univ. Science 5(7), 867–872 (2004)
16. Zhang, N., Karp, B., Floyd, S., Peterson, L.: RR-TCP: A reordering-robust TCP with DSACK, Tech. Rep. TR-02-006, ICSI, Berkeley, CA (2002)

Solving the Physical Impairment Aware Routing and Wavelength Assignment Problem in Optical WDM Networks Using a Tabu Search Based Hyper-Heuristic Approach

Ali Keleş, A. Şima Uyar, and Ayşegül Yayımlı

Computer Engineering Department, Istanbul Techincal University, Istanbul, Turkey
{kelesal,etaner,gencata}@itu.edu.tr

Abstract. In this paper, a tabu search based hyper heuristic is applied to the routing and wavelength assignment problem, considering physical impairments caused by Amplified Spontaneous Emission noise in erbium-doped fiber amplifiers and crosstalk noise at optical cross-connects. The objective is to minimize the total bit error rate of the routed lightpaths over optical wavelength division multiplexing networks. The results of the tabu search based hyper-heuristics are compared with single heuristic approaches. The results show that different heuristics provide the best result for different instances. The tabu search based hyper heuristic, which combines all the heuristics, gives comparable results to the single heuristics while using a modest amount of time. Furthermore, it has the best results for some problem instances.

Keywords: Optical WDM Networks, Routing and Wavelength Assignment, Physical Impairments, Hyper-Heuristics, Tabu Search.

1 Introduction

The aim of the routing and wavelength assignment (RWA) problem is to establish lightpaths between given source and destination nodes in an optical wavelength division multiplexing (WDM) network with the aim of minimizing the use of network resources. A lightpath is an end-to-end optical connection that is established over the physical topology and used by the upper layers (IP, Ethernet, etc.) to transmit data. A fiber link of the physical topology allows multiple channels on different wavelengths to coexist, hence, multiple lightpaths, each operating on a different wavelength can be routed on a single fiber link. A lightpath may span multiple fiber links passing through optical cross-connects at the intermediate nodes to create a connection between two physically non-adjacent end nodes. Finding proper routes and assigning proper wavelengths to each of the lightpaths in a network is known as routing and wavelength assignment.

There are two types of RWA problems: In the dynamic RWA problem, the source and destination pairs for lightpaths are not known a priori. Lightpaths are established as the demands occur. These lightpaths exist permanently or

C. Di Chio et al. (Eds.): EvoApplications 2010, Part II, LNCS 6025, pp. 81–90, 2010.

stay connected for a duration of time. In the static RWA problem, all the source and destination nodes of the lightpaths are known a priori. The problem is to route all the lightpaths while minimizing resource usage in the network. Here, the order of handling the lightpaths affects the resource usage. The static RWA is an NP-hard optimization problem [1].

In literature, many studies for solving both the static and the dynamic RWA problems can be found. In a real world RWA problem, none of the optical devices are produced perfectly, therefore physical impairments (PI) should be considered. Each optical device can add noise to the transmitted power for a lightpath. The transmitted light will be exposed to the noises generated by the optical network devices such as erbium-doped fiber amplifiers, optical cross-connects and the fiber link itself. As the light propagates, its optical signal to noise ratio is reduced. Amplifiers are used to increase this ratio, however, these devices add noise to the signal known as Amplified Spontaneous Emission. Each lightpath is switched at optical cross-connects to go from one input link of a node to an output link of this node. The signal of a lightpath that goes through the optical cross-connects affects the other signals of the lightpaths that have the same wavelength. This causes another noise known as crosstalk noise. These noises and signal degradations are known as PIs.

A Quality of transmission should be provided to the users. PIs affect the optical signal to noise ratio of the connection. As the optical signal to noise ratio of the transmitted power degrades, the quality of transmission provided to the user decreases. A solution to the RWA problem must consider these PIs to manage a acceptable quality of transmission. There are two types of PIs: dynamic and static. In the static PI, the value of the impairment do not depend on the current network status. The effect of PI on the established lightpaths is not changed after a new lightpath is established. However, in the dynamic type, the value of the PI depends on the network status and must be recalculated after a change in the network, such as establishing a new lightpath.

There are different methods to determine whether the considered lightpath is appropriate for quality of transmission or not. One of these is to check if the optical signal to noise ratio of the received signal is above a threshold value or not. Another method is to calculate the total bit error rate (BER) of the established lightpaths. Analytical models and hybrid approaches exist for calculating the BER value. Hybrid approaches require simulations or monitoring to be used with analytical models [2]. In this study, the analytical models in [3] are used.

PI-aware RWA (PI-RWA) can be divided into two subproblems and these subproblems can be solved separately. The routing subproblem can be solved using various heuristic approaches [1]. As the problem parameters change, the performance of these heuristic can vary. In this study, hyper-heuristics (HHs) are applied to the routing part of PI-RWA. HHs are heuristics to choose heuristics [4]. They operate on the search space of heuristics instead of the solution search space of the problem. Through hybridizing different heuristic approaches, HHs can provide acceptable results without problem specific information. No previous studies exist in literature for solving the PI-RWA with HHs. In this study, a

tabu search based HH (TSHH) is proposed for the PI-RWA and its results are compared with single heuristic approaches.

2 Literature Survey and Problem Definition

2.1 Literature Survey

RWA is extensively studied in many papers. However, PI-RWA is a new topic which has attracted a lot of attention in recent years. The studies for the static PI-RWA in literature are very limited. A detailed survey for PI-RWA is given in [2].

In [5], the static RWA problem is solved using Integer Linear Programming. The authors use the impairments as a cost metric for determining k-shortest paths and to select the first available path for a lightpath from this set. Binary Integer Linear Programming is proposed in [6] and the authors compare their results with previous studies. Besides the ILP approach, in [7], authors propose new heuristics and regenerator placement methods to solve the static PI-RWA. After obtaining k-shortest paths for the virtual topology, the static PIs are calculated. If the calculated BER is above a given threshold, a new connection is added to the virtual topology to make the routing feasible. A different wavelength assignment heuristic is studied in [8]. The authors recommend using the wavelengths placed at the center of the transmission window for longer lightpaths, since the four-wave mixing noise is more effective for wavelengths at the the center of the transmission window. A Genetic algorithm approach is applied to PI-RWA in [9].

2.2 Routing and Wavelength Assignment Problem

The objective of the problem is:

$$Minimize \sum_{l=0}^{n} BER_l \tag{1}$$

where BER_l is the bit error rate of lightpath l and is calculated as in Eq. 2.

$$BER = \frac{1}{2} erfc(\frac{Q}{\sqrt{2}}) \approx \frac{e^{\frac{-Q^2}{2}}}{Q\sqrt{2\pi}} \tag{2}$$

In Eq. 2, erfc is a complementary error function as calculated in Eq. 3.

$$erfc(t) = \int_t^{\infty} \frac{e^{-x^2}}{\sqrt{2\pi}} \tag{3}$$

The formulation for the Q Factor is [10]:

$$Q = \frac{I_1 - I_0}{\sigma_1 + \sigma_0} \tag{4}$$

where, I_i is the mean of received bits with value i: 0 or 1, and σ_i is the standard deviation of this distribution [6].

$$\sigma_i = \sigma_{sxi} + \sigma_{sspi} + \sigma_{shi} + \sigma_{th} \tag{5}$$

where σ_{sxi} is the noise variance between signal and crosstalk; σ_{sspi} is the noise variance between the signal and amplified spontaneous emission; σ_{shi} is the noise variance of the shot noise and σ_{th} is the noise variance of the thermal noise.

3 Solution Approaches

For the wavelength assignment part of RWA the First Fit assignment heuristic approach is used. This heuristic finds the first unused wavelength over the physical links that are used to route the considered lightpath and assigns this wavelength to the lightpath. In this study, we focus on the heuristic approaches for routing the lightpaths, while minimizing the BER.

The Shortest Path (SP) heuristic method assigns the shortest path between the given source and destination nodes. Dijkstra's algorithm is used for finding the shortest paths at each step of the SP heuristic based on the physical distances of the links. The K-Shortest Path (KSP) heuristic finds the k-shortest paths between the source and the destination over the available physical topology. Then, it randomly selects one of the k-shortest paths for assigning to a lightpath. The Least Congested Path (LCP) [1] heuristic selects a path from the k-shortest paths. The selection is done according to the congestion order of the paths. The congestion order is the maximum number of unavailable wavelengths among all the links of a path. In SP, KSP and LCP heuristics, the Q-factor which is used to calculate the BER of a lightpath, is not used as a metric. The Lowest BER Path (LBERP) [11] heuristic computes the BER for the k-shortest paths and selects the path with the lowest BER for the considered lightpath. The last heuristic is Minimizing Highest BER Path (MinHBERP) [11]. When the dynamic PI is considered, establishing a lightpath is going to affect the BER of the other lightpaths. This heuristic method selects the path among the k-shortest paths, which has the minimum value for the maximum BER of all the established lightpaths. All the heuristic methods are taken from previous studies in literature, however, it should be noted that they are used with modifications in this study. All the heuristics are constructive heuristics.

3.1 Hyper-Heuristics

Hyper-heuristics (HHs) aim to create a generic, reusable method for solving different problems and problem instances [12]. Unlike meta-heuristics, HHs have no problem specific parameters. This makes HHs a general problem solver which does not require expertise about the problem to obtain a feasible solution. HHs operate on the search space of heuristics, whereas meta-heuristics operate on the search space of solutions [13]. The heuristics used by HHs, are called low

level heuristics. As different low level heuristics can have drawbacks for different problem instances, using them together can help to overcome these drawbacks [4]. Constructive heuristics generate a part of the solution at each step. In literature, constructive heuristics, such as graph coloring heuristics [13], exist. Constructive HHs try to determine an optimal application order of these heuristics to construct a solution. In literature, constructive HHs have been studied commonly for timetabling and scheduling problems [13,12,14]. Simulated Annealing [12], Tabu Search [13] and the Ant Colony approach [15] can be used as a framework for applying constructive HHs. In this study, Tabu Search is chosen.

3.2 Proposed Method

In this study, a tabu search based constructive hyper heuristic (TSHH) is proposed to solve the PI-RWA problem. Tabu search is used to minimize the total BER of all the established lightpaths, as a lower BER is preffered for quality of transmission. The proposed method here is very similar to the one used by Burke et. al. in [13], where a tabu search based HH is used to solve timetabling problems with constructive heuristics. In ourstudy, as low level heuristics, SP, KSP, LCP, LBERP and MinHBERP are used.

In our approach, a solution is represented by an array of heuristics that shows the order in which the heuristics will be applied at each iteration of the solution construction. The constructive heuristic located on the n_{th} entry of the array is used to route the n_{th} lightpath. According to this, the size of the heuristics array equals the number of source-destination (s-d) pairs, i.e., the total number of lightpaths to be set up. All entries of the array are initialized with the SP heuristic.

The Tabu search method randomly changes one of the low level heuristics on the array with another randomly selected one to search on the heuristics space at each iteration. The order of choosing the lightpaths to establish is kept constant. At each iteration of the tabu search, a solution candidate for the PI-RWA is generated and its solution quality is calculated. Starting from the beginning of the heuristics array, each lightpath is routed using the corresponding heuristic on the array. The wavelength assignment is always done using the first fit assignment heuristic. The BER of the lightpath is calculated using analytical PI models. If the routing or the wavelength assignment fails because of insufficient wavelengths, the generated solution is considered infeasible. Infeasible solutions are added to a failed list, so that the corresponding heuristic arrays will not be visited again. The feasible solutions are added to a tabu list so that they will not be revisited for a number of iterations, which is determined by the size of the tabu list. Tabu search keeps the best solution and updates it at each iteration. Finally, after a constant number of iterations, tabu search ends. The quality of a solution is represented with the total BER of all the established lightpaths. Tabu search gives the solution of the PI-RWA that has the lowest BER.

4 Experimental Design

The results of the PI-RWA is obtained for both the NSF network and another telco network with 24 nodes and 43 links [1]. 20 different randomly generated virtual topologies (VT) are routed over these physical networks. The number of lightpath requests is 56 lightpaths for the NSFNET and 72 for the 24-node network. The number of wavelengths is chosen as 16 for both topologies. The distances of all the links are taken to be 100 km [3], as we assume that there are no amplifiers along the physical links. The parameters used for the PIs are given in Table 1.

Table 1. Parameters for Physical Impairment Model [3]

Parameter	Value
Loss of Multiplexer	4 dB
Loss of Demultiplexer	4 dB
Loss of Switch Element Insertion	1 dB
Loss of Waveguide/fiber coupling	1 dB
Loss of Switch	$2log_2$(N*Ls) + 4Lw
Loss of Tap	1 dB
Loss of Fiber	0.2 dB/km
Gain of Input EDFA	22 dB
Gain of Output EDFA	16 dB (for less than 3 input), 18 dB
Switch Crosstalk Ratio	30 dB

The maximum allowed number of iterations for Tabu Search is 300. The sizes of the tabu list and the failed list are 50 and 300 respectively. The value of k is set to 5 for the k-shortest-path. The results are obtained over 10 runs for each 20 randomly generated different VT. All parameter settings are determined experimentally. In this study, the performance of the TSHH is compared with the single heuristic approaches. While the order of the s-d pairs is always the same for all runs and all tabu iterations, TSHH searches by changing the order of the heuristics. Since the SP, LCP, LBERP, and MinHBERP heuristics are all deterministic using the same order of s-d pairs for routing, always produces the same results, therefore, to provide similar variety to the single heuristics, the routing order of the s-d pairs are changed for each run of the heuristic. The number of permutations for creating different orders of s-d pairs is decided by the number of runs for a VT and the iteration count of tabu search. As the quality of the solution changes with the order of establishing the lightpaths, giving 3000 (300 x 10) different permutations to a single heuristic, makes the comparison of obtained results fair with the TSHH results.

5 Results

The total BER of all the established lightpaths of a VT is given as the result of a run of a program for a VT. In the tables, boldface shows the best result obtained for the corresponding VT. The results for 20 VTs for NSFNET is given in Table 2. We see that the heuristic giving the best result changes for different VTs. The performance of LBERP and MinHBERP is generally better than the other heuristics. KSP has the worst performance in all VTs. For VT11, the best performance belongs to TSHH. On the other hand, the results of TSHH for each VT is comparable with the results of others. Since the fitness function is designed as minimizing the BER, the performance of LBERP and MinHBERP is better in many instances. We should mention that if the structure of the fitness function changes, the performance of LBERP and MinHBERP may decrease, but TSHH is expected to find the combination of heuristics that gives acceptable results.

Table 2. Results for 20 VTs for NSFNET

Problem Instances	SP	KSP	LCP	LBERP	MinHBERP	TSHH
VT1	**2.55x10^{-18}**	4.78x10^{-10}	4.56x10^{-12}	2.68x10^{-18}	3.47x10^{-15}	1.11x10^{-12}
VT2	3.16x10^{-14}	3.96x10^{-10}	4.62x10^{-12}	**2.03x10^{-14}**	8.09x10^{-13}	2.31x10^{-12}
VT3	2.84x10^{-14}	5.04x10^{-10}	8.43x10^{-12}	**1.81x10^{-14}**	2.11x10^{-14}	5.86x10^{-11}
VT4	3.68x10^{-14}	3.81x10^{-10}	1.01x10^{-11}	**2.34x10^{-14}**	9.07x10^{-13}	4.37x10^{-12}
VT5	4.18x10^{-14}	5.93x10^{-10}	6.77x10^{-12}	**2.41x10^{-14}**	2.40x10^{-12}	2.27x10^{-12}
VT6	2.06x10^{-12}	8.37x10^{-11}	9.03x10^{-12}	**2.41x10^{-14}**	1.13x10^{-12}	4.72x10^{-12}
VT7	2.75x10^{-14}	6.99x10^{-10}	6.33x10^{-11}	2.38x10^{-14}	**1.72x10^{-14}**	5.59x10^{-12}
VT8	4.11x10^{-14}	9.49x10^{-11}	6.77x10^{-11}	1.15x10^{-12}	**3.41x10^{-14}**	8.08x10^{-13}
VT9	2.14x10^{-14}	6.61x10^{-10}	9.81x10^{-12}	**1.01x10^{-14}**	1.85x10^{-14}	2.32x10^{-14}
VT10	2.35x10^{-12}	5.75x10^{-10}	1.38x10^{-10}	**4.42x10^{-14}**	1.12x10^{-12}	3.26x10^{-12}
VT11	2.90x10^{-14}	3.55x10^{-10}	3.62x10^{-12}	2.81x10^{-14}	3.00x10^{-14}	**2.61x10^{-14}**
VT12	1.17x10^{-12}	3.71x10^{-10}	3.51x10^{-12}	**8.64x10^{-13}**	**8.64x10^{-13}**	4.49x10^{-12}
VT13	6.89x10^{-15}	1.70x10^{-10}	7.67x10^{-12}	**5.42x10^{-15}**	1.15x10^{-14}	8.87x10^{-13}
VT14	**3.23x10^{-14}**	4.05x10^{-10}	3.57x10^{-12}	3.47x10^{-12}	1.17x10^{-12}	1.11x10^{-11}
VT15	3.46x10^{-14}	3.22x10^{-10}	7.57x10^{-12}	3.98x10^{-14}	**2.07x10^{-14}**	2.34x10^{-12}
VT16	**4.13x10^{-15}**	6.66x10^{-10}	2.05x10^{-12}	1.20x10^{-14}	1.52x10^{-14}	4.20x10^{-12}
VT17	9.27x10^{-15}	9.93x10^{-10}	4.07x10^{-12}	**7.15x10^{-15}**	1.71x10^{-14}	1.11x10^{-12}
VT18	2.11x10^{-14}	1.94x10^{-10}	6.04x10^{-12}	**1.42x10^{-14}**	3.42x10^{-14}	1.15x10^{-12}
VT19	1.76x10^{-14}	2.42x10^{-10}	6.04x10^{-12}	1.50x10^{-14}	**1.04x10^{-14}**	3.25x10^{-12}
VT20	1.17x10^{-12}	1.82x10^{-10}	6.29x10^{-11}	2.42x10^{-14}	**2.18x10^{-14}**	1.18x10^{-12}

In Table 3 the results for 20 VTs for the 24-node network are given. We see that the SP heuristic is better than the minHBERP heuristic which is the second successful heuristic for the NSFNET. TSHH gives the best result for VT11. The results of TSHH for VT5, VT8, VT9, VT10, VT15, VT17 and VT18 are almost the same with the best results.

The results for running time of the heuristics can be seen in Table 4. MinHBERP uses more CPU time than the others. SP is the fastest heuristic approach. The running time of TSHH is an average of other heuristics. As TSHH

uses MinHBERP, its running time is worse than SP or LBERP, however, TSHH executes faster than minHBERP as it also uses the other heuristics. The results also show that, as the size of the network increases, the time requirement of TSHH increases at a slower rate than the other single heuristics. To obtain these results, a dual-core 2.13 GHz Pentium PC with 2 GB of RAM is used with Java as a programming language.

The standard error (SE) of the mean values are given in Table 5 for all methods. It can be seen that TSHH has a smaller SE than most the others, showing that it produces consistent results for all instances.

Table 3. Results for 20 VTs for 24-node 43-link Network

Problem Instances	SP	KSP	LCP	LBERP	MinHBERP	TSHH
VT1	$2.26\text{x}10^{-8}$	$1.50\text{x}10^{-6}$	$1.92\text{x}10^{-6}$	$2.94\text{x}10^{-8}$	$\mathbf{2.18\text{x}10^{-8}}$	$1.40\text{x}10^{-7}$
VT2	$7.53\text{x}10^{-9}$	$5.79\text{x}10^{-7}$	$1.63\text{x}10^{-7}$	$\mathbf{1.34\text{x}10^{-10}}$	$6.24\text{x}10^{-9}$	$8.15\text{x}10^{-9}$
VT3	$\mathbf{3.41\text{x}10^{-9}}$	$1.39\text{x}10^{-6}$	$4.01\text{x}10^{-7}$	$7.48\text{x}10^{-9}$	$3.48\text{x}10^{-9}$	$2.85\text{x}10^{-8}$
VT4	$2.01\text{x}10^{-9}$	$1.85\text{x}10^{-6}$	$1.14\text{x}10^{-7}$	$\mathbf{5.25\text{x}10^{-10}}$	$1.90\text{x}10^{-9}$	$5.53\text{x}10^{-8}$
VT5	$5.54\text{x}10^{-9}$	$3.24\text{x}10^{-7}$	$1.66\text{x}10^{-7}$	$\mathbf{1.45\text{x}10^{-9}}$	$5.20\text{x}10^{-9}$	$5.04\text{x}10^{-9}$
VT6	$1.80\text{x}10^{-9}$	$5.32\text{x}10^{-7}$	$1.70\text{x}10^{-7}$	$\mathbf{2.79\text{x}10^{-10}}$	$2.91\text{x}10^{-9}$	$4.54\text{x}10^{-8}$
VT7	$6.68\text{x}10^{-9}$	$2.19\text{x}10^{-6}$	$5.31\text{x}10^{-7}$	$\mathbf{5.77\text{x}10^{-9}}$	$1.35\text{x}10^{-8}$	$2.15\text{x}10^{-8}$
VT8	$4.54\text{x}10^{-8}$	$4.83\text{x}10^{-7}$	$3.45\text{x}10^{-7}$	$\mathbf{3.24\text{x}10^{-8}}$	$5.05\text{x}10^{-8}$	$9.85\text{x}10^{-8}$
VT9	$4.21\text{x}10^{-8}$	$4.10\text{x}10^{-7}$	$7.29\text{x}10^{-8}$	$\mathbf{1.45\text{x}10^{-8}}$	$2.29\text{x}10^{-8}$	$4.78\text{x}10^{-8}$
VT10	$\mathbf{1.65\text{x}10^{-9}}$	$5.16\text{x}10^{-7}$	$5.76\text{x}10^{-8}$	$1.72\text{x}10^{-9}$	$1.96\text{x}10^{-9}$	$4.23\text{x}10^{-9}$
VT11	$5.77\text{x}10^{-9}$	$1.26\text{x}10^{-6}$	$1.88\text{x}10^{-7}$	$5.88\text{x}10^{-9}$	$5.98\text{x}10^{-9}$	$\mathbf{5.68\text{x}10^{-9}}$
VT12	$\mathbf{7.29\text{x}10^{-9}}$	$1.07\text{x}10^{-6}$	$5.10\text{x}10^{-8}$	$9.02\text{x}10^{-9}$	$7.96\text{x}10^{-9}$	$4.19\text{x}10^{-8}$
VT13	$6.29\text{x}10^{-8}$	$4.60\text{x}10^{-7}$	$7.63\text{x}10^{-7}$	$8.31\text{x}10^{-8}$	$\mathbf{5.59\text{x}10^{-8}}$	$1.48\text{x}10^{-7}$
VT14	$\mathbf{6.42\text{x}10^{-9}}$	$4.95\text{x}10^{-7}$	$2.36\text{x}10^{-7}$	$8.53\text{x}10^{-9}$	$7.85\text{x}10^{-9}$	$1.53\text{x}10^{-8}$
VT15	$7.07\text{x}10^{-8}$	$1.30\text{x}10^{-6}$	$6.18\text{x}10^{-7}$	$\mathbf{1.93\text{x}10^{-8}}$	$5.35\text{x}10^{-8}$	$2.40\text{x}10^{-8}$
VT16	$2.76\text{x}10^{-8}$	$6.34\text{x}10^{-7}$	$1.70\text{x}10^{-7}$	$\mathbf{5.84\text{x}10^{-9}}$	$3.49\text{x}10^{-8}$	$1.09\text{x}10^{-7}$
VT17	$1.40\text{x}10^{-8}$	$4.17\text{x}10^{-7}$	$5.12\text{x}10^{-8}$	$\mathbf{3.17\text{x}10^{-9}}$	$5.00\text{x}10^{-9}$	$9.24\text{x}10^{-9}$
VT18	$\mathbf{6.19\text{x}10^{-9}}$	$6.32\text{x}10^{-7}$	$6.90\text{x}10^{-7}$	$2.11\text{x}10^{-8}$	$8.53\text{x}10^{-9}$	$9.04\text{x}10^{-9}$
VT19	$2.27\text{x}10^{-7}$	$4.26\text{x}10^{-7}$	$1.83\text{x}10^{-7}$	$\mathbf{6.76\text{x}10^{-9}}$	$1.30\text{x}10^{-7}$	$9.58\text{x}10^{-8}$
VT20	$9.33\text{x}10^{-9}$	$7.78\text{x}10^{-7}$	$2.19\text{x}10^{-8}$	$4.35\text{x}10^{-9}$	$\mathbf{2.84\text{x}10^{-9}}$	$3.21\text{x}10^{-8}$

Table 4. Running Time of the programs in seconds

Problem Instances	SP	KSP	LCP	LBERP	MinHBERP	TSHH
NSFNET	0.42	0.94	0.76	3.15	30.68	32.54
24-node Network	4.24	13.8	12.74	58.47	285.14	72.78

Table 5. Standard Errors of Results for NSFNET and 24-node Network

Network	SP	KSP	LCP	LBERP	MinHBERP	TSHH
NSF	$2.17\text{x}10^{-12}$	$1.10\text{x}10^{-9}$	$10\text{x}10^{-10}$	$2.06\text{x}10^{-12}$	$1.30\text{x}10^{-10}$	$1.76\text{x}10^{-12}$
24-node	$1.75\text{x}10^{-8}$	$2.30\text{x}10^{-7}$	$1.64\text{x}10^{-7}$	$1.54\text{x}10^{-8}$	$2.78\text{x}10^{-8}$	$1.42\text{x}10^{-8}$

6 Conclusion and Future Work

RWA problem is extensively studied in literature. However, the PI aware RWA has started receiving more attention only in the recent years. In this study, TSHH is applied to the PI-RWA for the first time and the results are compared with single heuristic approaches. The heuristics considered for TSHH are for the routing part of the RWA. The wavelength assignment is always done using the first fit assignment heuristic. PIs due to amplified spontaneous emission and cross-talk noise are calculated with analytical models. The objective is to minimize the total BER of all the established lightpaths.

Constructive heuristics are used as low level heuristics of the TSHH. As it can be seen from the results, TSHH gives comparable results with the other single heuristic methods. Furthermore, TSHH gives the best results for some of the VTs. The results also show that, as the size of the network increases, the time requirement of TSHH increases at a slower rate than the other single heuristics. We see that the performance of heuristics changes with the problem instances. HHs can be a general solver without a priori knowledge of problem specific data. This work is a preliminary study, which is part of a larger project. In the project, the Virtual Topology Design problem, which includes the PI-RWA as a subproblem, will be solved. Also, other HHs, such as Ant Colony based HH [15], will be explored as future work.

References

1. Mukherjee, B.: Optical WDM Networks. Springer, New York (2006)
2. Azodolmolky, S., Klinkowski, M., Marin, E., Careglio, D., Pareta, J., Tomkos, I.: A survey on physical layer impairments aware routing and wavelength assignment algorithms in optical networks. Comput. Netw. 53(7), 926–944 (2009)
3. Ramamurthy, B., Datta, D., Feng, H., Heritage, J.P., Mukherjee, B.: Impact of transmission impairments on the teletraffic performance of wavelength-routed optical networks. J. Lightwave Technol. 17(10), 1713 (1999)
4. Burke, E., Kendall, G., Newall, J., Hart, E., Ross, P., Schulenburg, S.: Hyper-heuristics: An emerging direction in modern search technology. In: Handbook of Metaheuristics, ch. 16, pp. 457–474. Springer, Heidelberg (2003)
5. Tomkos, I., Vogiatzis, D., Mas, C., Zacharopoulos, I., Tzanakaki, A., Varvarigos, E.: Performance engineering of metropolitan area optical networks through impairment constraint routing. IEEE Communications Magazine 42(8), 40–47 (2004)
6. Marino, P., Azodolmolky, S., Aparicio-Pardo, R., Garcia-Manrubia, B., Pointurier, Y., Angelou, M., Sole-Pareta, J., Garcia-Haro, J., Tomkos, I.: Offline impairment aware rwa algorithms for cross-layer planning of optical networks. J. Lightwave Technol. 27(12), 1763–1775 (2009)
7. Manousakis, K., Christodoulopoulos, K., Kamitsas, E., Tomkos, I., Varvarigos, E.: Offline impairment-aware routing and wavelength assignment algorithms in translucent wdm optical networks. J. Lightwave Technol. 27(12), 1866–1877 (2009)
8. Adhya, A., Datta, D.: Design methodology for wdm backbone networks using fwm-aware heuristic algorithm. Optical Switching and Networking 6(1), 10–19 (2009)

9. Waldman, H., Pavani, G.S.: Using genetic algorithms in constrained routing and wavelength assignment. In: 8th IFIP Working Conference on Optical Network Design and Modelling - ONDM 2004, vol. 1, pp. 565–584 (2004)
10. Agrawal, G.P.: Fiber-Optic Communication Systems, 3rd edn. Wiley, New York (2002)
11. Pointurier, Y., Brandt-Pearce, M., Subramaniam, S., Xu, B.: Cross-layer adaptive routing and wavelength assignment in all-optical networks. IEEE Journal on Selected Areas in Communications 26(6 suppl.), 32–44 (2008)
12. Bai, R., Burke, E., Kendall, G., Mccollum, B.: A simulated annealing hyperheuristic for university course timetabling. In: Proceedings of 6th International Conference on the Practice and Theory of Automated Timetabling, pp. 345–350 (2006)
13. Burke, E., McCollum, B., Meisels, A., Petrovic, S., Qu, R.: A graph-based hyperheuristic for educational timetabling problems. European Journal of Operational Research 176(1), 177–192 (2007)
14. Qu, R., Burke, E., McCollum, B.: Adaptive automated construction of hybrid heuristics for exam timetabling and graph colouring problems. European Journal of Operational Research 198(2), 392–404 (2009)
15. Burke, E., Kendall, G., Silva, D., O'Brien, R., Soubeiga, E.: An ant algorithm hyperheuristic for the project presentation scheduling problem. In: Congress on Evolutionary Computation, pp. 2263–2270. IEEE, Los Alamitos (2005)

A Generalized, Location-Based Model of Connections in Ad-Hoc Networks Improving the Performance of Ant Routing

Michal Kudelski[1,2] and Andrzej Pacut[1,2]

[1] Institute of Control and Computation Engineering, Warsaw Univ. of Technology
Nowowiejska 15/19, 00-665 Warsaw, Poland
m.kudelski@elka.pw.edu.pl, A.Pacut@ia.pw.edu.pl
[2] NASK
Wawozowa 18, 02-796 Warsaw, Poland

Abstract. We formalize and analyze a generalized model of connections in ad-hoc networks. The proposed model defines connections on the locations level rather than on the nodes level. We show how the generalized model improves the stability of connections if the ant learning mechanism is applied to solve the routing problem. Stable connections reduce the overhead generated by ant routing mechanism and improve its performance in terms of end-to-end delay and delivery ratio. Moreover, connections on the locations level reflect physical connections in ad-hoc networks very well and agree with biological inspirations of ant algorithms.

1 Introduction

It is commonly known that an ad-hoc network makes a very demanding environment [8]. Ad-hoc networks are dynamic, fast changing systems of unknown and changing structure and parameters. One of the basic issues is the problem of routing. Finding the best routes from a source node to a destination node under dynamically changing topology is a typical example of a complex task in ad-hoc networks.

Adaptive learning structures seem to be a very adequate solution in ad-hoc networks. There are many motivations for using learning approaches in such systems. The need of high adaptation abilities is probably the strongest one. A non-deterministic and complex behavior of ad-hoc networks makes it difficult or impossible to use classical methods. Ant routing may be a prominent example of a learning technique employed to solve a difficult problem in ad-hoc networks.

One significant challenge for a learning mechanism applied in an ad-hoc network is the stability of the model used for learning. Considering an ant routing mechanism, the model of connections between nodes in the network is required to solve the problem. Commonly, the model is defined on the nodes level: each routing path is defined as a sequence of nodes. Thus, a connection loss between two individual nodes often implies a significant modification of the routing policy. The learning mechanism needs to adapt the policy to the new structure of

C. Di Chio et al. (Eds.): EvoApplications 2010, Part II, LNCS 6025, pp. 91–100, 2010.

connections. Consequently, the model of connections can be unstable and may cause the operation of ant learning mechanism to deteriorate.

The aim of this paper is to show how a generalized model of connections defined on the locations level may improve the stability of connections seen by the ant learning mechanism in an ad-hoc network. We use Geographical Localization of Knowledge [7,9] to introduce and formalize the generalized model of connections in ad-hoc networks. We show that the connections between geographical locations are much more robust to dynamic topology changes than the connections between nodes. The more stable model of connections improves the operation of ant routing mechanism and, as a result, the overall performance of the network.

The paper is organized as follows. In Section 2, we describe the related works. Section 3 introduces the generalized model of connections defined on the locations level. In Section 4, we briefly describe the AntHocGeo algorithm which operates on the generalized model of connections. Experimental results are presented and discussed in Section 5. Section 6 summarizes the paper.

2 Related Work

In recent years, a large number of ad-hoc routing algorithms have been proposed (see [1,4,14] for reviews). At the same time, different learning mechanisms were invented to solve various problems in ad-hoc networks [8]. Amongst them, a group of ant routing algorithms may be identified [6].

The idea of a generalized model of connections in ad-hoc networks has not yet been widely studied in the literature. Some hierarchical routing algorithms introduce multiple levels of connections, yet such models have not been used together with learning mechanisms.

For instance, a multilevel logical clustering scheme is proposed in the *Hierarchical State Routing* (*HSR*) protocol [5]. A hierarchy of links is created, where *virtual links* on higher levels are realized by lower level nodes. Thus, virtual links provide a generalized model of connections.

Authors of [10] propose a scheme of defining routes on the locations level which is similar to our model of connections. In their *GRID* mechanism, the network is partitioned into logical grids and routing is then performed in a grid-by-grid manner. However, there are two significant differences. First, and most important, no learning is performed in the model (a purely reactive routing scheme is used instead). Second, the implementation of the model is different: a leader of each grid is elected instead of our idea of distributing the knowledge between the nodes within a given location.

3 Generalized Model of Connections

We base our model on the concept of *geographical cells* [7]. We use geographical cells together with location information to construct a higher level model of connections in the space of geographical areas rather than in the space of individual nodes.

In general, a physical system may be described by diverse models [12]. They may be created in different time scales, have different accuracy, model different components of the system, etc. In other words, diverse models of the same system may have different time and space resolution. The latter may be referred to as a *model granularity*, reflecting the level of details recognized in the model. In [12], author presents and analyzes neural models of different time scales and granularity. Following the idea, we want to *zoom out* the classical model of connections in ad-hoc networks from the level of nodes to the level of locations, thus changing the granularity of the model and making it more general.

We illustrate the process of model generalization on Fig. 1.

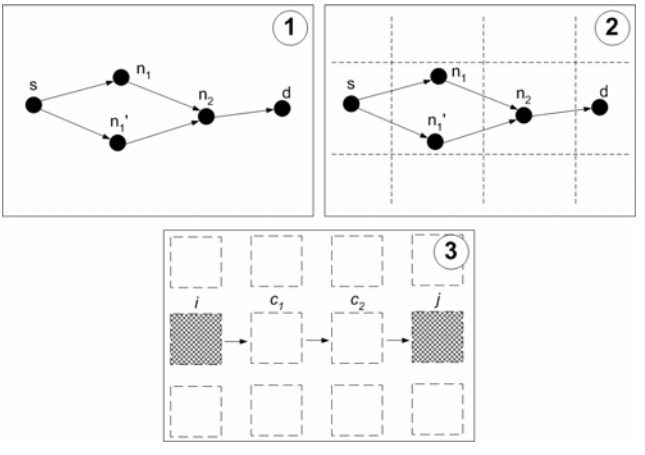


Fig. 1. Illustration of a model generalization

Fig. 1(1) demonstrates a simple ad-hoc network and its connections on the nodes level. For simplicity, we assume a static situation, i.e. nodes keep their positions. There are two paths between a source node s and a destination node d: $\{s, n_1, n_2, d\}$ and $\left\{s, n_1^{'}, n_2, d\right\}$. Fig. 1(2) introduces *geographical cells*. Nodes n_1 and $n_1^{'}$ are located in the same neighborhood (in a sense of the communication range), thus they belong to the same cell. Fig. 1(3) defines paths on the locations level, aggregating two separate node-level paths into a single *higher level* path $\{i, c_1, c_2, j\}$ consisting of geographical cells. The resulting connections model does not recognize individual nodes in a geographical cell — it rather focuses on the connectivity between cells. Thus, we may say that it is on a higher level of granularity. We formalize the model using ad-hoc network graphs.

Ad-hoc network graph - nodes level. Considering the classical approach on the nodes level, a mobile ad-hoc network may be defined as a dynamic multi-hop graph $G = (N, L)$, where N is a finite set of mobile nodes and L is a set of edges

representing wireless links [2]. A link $(i,j) \in L$ exists if and only if the distance between two mobile nodes is less or equal than a fixed radius r:

$$(i,j) \in L \Leftrightarrow d(i,j) \leq r \qquad i,j \in N \ , \ i \neq j \ , \tag{1}$$

where d is an euclidean distance function. The value of r represents the transmission range of wireless communication devices.

The *neighborhood of a node* x is defined by the set of nodes in N that are inside a circle with center at x and radius r, and is denoted by

$$Nr(x) = \left\{ n_j | d(x,n_j) \leq r \ , \ x \neq n_j \ , \ \forall_{n_j \in N} \ \right\} \tag{2}$$

A path from a source node s to a destination node d is a sequence of nodes, which we denote by:

$$P_{s,d} = [s, n_1, n_2, \ldots, n_k, d] \ , \tag{3}$$

where

$$\begin{array}{l} s, d, n_1, \ldots, n_k \in N \\ (s,n_1), (n_k,d), (n_y, n_{y+1}) \in L \ \text{ for } \ 1 \leq y \leq k-1. \end{array} \tag{4}$$

Due to mobility of the nodes, the set of paths between the nodes together with the distances are changing over time. New links can be established and the existing links can vanish.

Ad-hoc network graph - locations level. On the locations level, we may model an ad-hoc network as a dynamic multihop graph $G = (C, L^C)$, where C is a finite (and predefined) set of geographical cells and L^C is a set of edges representing connections between geographical cells.

In order to define links on the geographical cells level, we first define a membership function $c(x) = c$. The function indicates that the node $x \in N$ is currently located in the cell $c \in C$. We also need to define a function determining a connectivity between two cells, namely

$$l(i,j) = \begin{cases} 1 & \exists_{n_1,n_2 \in N} \left[(d(n_1,n_2) \leq r) \wedge (c(n_1) = i) \wedge (c(n_2) = j) \right] \\ 0 & \text{otherwise} \end{cases}$$

A link $(i,j) \in L^C$ exists if and only if there is a direct communication between a node from the cell i and a node from the cell j, namely:

$$(i,j) \in L^C \Leftrightarrow l(i,j) = 1 \qquad i,j \in C \ , \ i \neq j \tag{5}$$

We define the *neighborhood of a cell* c as the set of cells in C that have a direct communication with c, namely

$$Nr(c) = \left\{ c_j | l(c,c_j) = 1 \ , \ c \neq c_j \ , \ \forall_{c_j \in C} \ \right\} \tag{6}$$

The *cell path* from a cell i to a cell j is a sequence of locations (represented by geographical cells), namely

$$C_{i,j} = [i, c_1, c_2, \ldots, c_k, j] \ , \tag{7}$$

where

$$\begin{array}{l} i, j, c_1, \ldots, c_k \in C \\ (i, c_1), (c_k, j), (c_y, c_{y+1}) \in L^C \ \text{ for } \ 1 \leq y \leq k-1. \end{array} \tag{8}$$

It is always possible to map higher level cell paths into lower level node paths, assuming that the connectivity exists. A higher level path consisting of k geographical cells $C_{c_1,c_k} = [c_1, c_2, \ldots, c_k]$ may be translated into lower lever path consisting of k' nodes, namely:

$$P_{n_1,n'_k} = [n_1, n_2, \ldots, n_{k'}] \quad , \tag{9}$$

where

$$c(n_1) = c_1 \quad , \quad c(n_{k'}) = c_k \tag{10}$$

and

$$\begin{array}{r} \forall_{y=1,\ldots,k'-1} \left[\ (c(n_y) = c_z) \wedge (c(n_{y+1}) = c_{z'}) \ \right] \Rightarrow [(z' = z) \vee (z' = z+1)] \\ c_z, c_{z'} \in C_{c_1,c_k} \quad , \quad z, z' \in \{1, \ldots, k\}. \end{array} \tag{11}$$

It is worth noticing that the proposed model does not cover direct connections between the neighboring nodes, e.g. the nodes from the same cell. It is assumed that such communication takes place independently of the model.

The model of connections in ad-hoc networks presented in this section is on the higher granularity level than the classical approaches. It takes into account the connectivity between *locations* in the network rather than low-level connections between individual nodes. One connection between neighboring locations may represent numerous node-to-node connections. Therefore, we claim that the higher level cell connections are more stable than the low-level connections on the nodes level. Consequently, we may expect that the generalized model will be more stable than the basic one, in a sense of its higher robustness to dynamic topology changes. We verify this claim in the later experiments (Sec. 5).

Moreover, higher level connections on the locations level reflect physical connections very well, as the latter are typically composed of successive hops made through successive geographical locations. The model also agrees with the biological inspiration of ant algorithms: real ants communicate with each other by leaving the pheromone on the ground, thus the information about paths is indeed defined on the locations level and does not depend on the ants mobility.

4 AntHocGeo Algorithm

The generalized model of connections was applied in *AntHocGeo*: an ant routing mechanism for ad-hoc networks.

AntHocGeo, proposed and discussed in detail in [7], is a modification of AntHocNet [3] which implements the concept of distributed geographical localization of knowledge [9].

Previous experiments performed with AntHocGeo proved that the concept of geographical localization of knowledge may improve the performance of the underlying ant routing mechanism. In this paper, we show that one of the most important sources of AntHocGeo's success is the generalized model of connections in the network.

5 Experimental Results

All experiments were performed using ns2 simulation environment [11]. Each result discussed below was obtained by averaging 20 simulation runs.

5.1 Connections Stability

In this section we show that the generalized learning model based on geographical cells is more ,,stable" than the model defined on the nodes level. Namely, the connections between cells are much more robust to dynamic topology changes than the connections between nodes.

In this order, we analyze the number of lost connections in a network during the whole simulation (Fig. 2). The network consists of nodes moving according to the Random Waypoint mobility model. Nodes speed is varied within the range of 5 m/s and 40 m/s. We compare AntHocNet and AntHocGeo routing algorithms. The simulation area (1000m x 1000m) is divided into 25 square cells when using AntHocGeo.

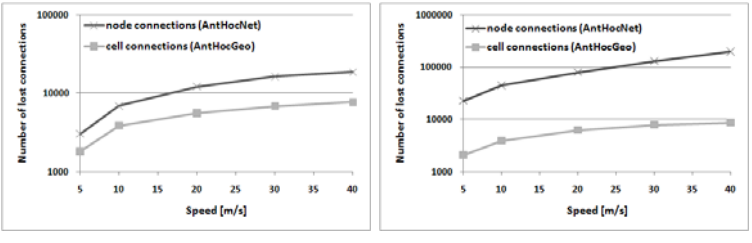


Fig. 2. Total number of lost connections during a simulation vs. node speed; 30 nodes in the network (left) and 100 nodes in the network (right)

We present two types of lost connections. The first one is a *node connection* which is a direct connection between two neighboring nodes which is used by AntHocNet in one of the currently utilized routing paths (on the nodes level). Such connection is considered lost when the nodes move outside of their direct communication range. The second type is a *cell connection* being a higher level connection between two geographical cells that is used by AntHocGeo in a routing path on the locations level. The connection between two cells is considered lost when none of the nodes from one cell is able to communicate directly with a node from the other cell. In both cases, we do not consider connections that are not exploited by the current routing policy.

The number of routing connections lost on the nodes level is much bigger than the number of connections lost between cells. This can be simply explained: a single connection loss between two neighboring nodes from different cells does

not imply the connection loss between those cells. There is always a chance that there are other nodes in the neighborhood that provide a communication between the cells. Consequently, there are much more connection losses on the nodes level then on the higher level of geographical cells. Finally, the connections on the cells level are much more stable, i.e. they are much more robust to dynamic changes of the topology in ad-hoc networks.

The obtained results show that not only the connections on the locations level are more stable than the connections on the nodes level, but the difference increases with the increase of the networks dynamics. The higher is the rate of topology changes, the higher is the frequency of lost connections on both levels. However, the number of connection losses on the cells level grows notably slower. This illustrates how the proposed approach may further improve the adaptation abilities of the underlying ant routing mechanism.

The results obtained for 100 nodes in the network (Fig. 2, right) may indicate that the number of lost direct node-to-node connections grows dramatically with the increased number of nodes in the network. While we observe even *tenfold* growth of the number of node-level connections in AntHocNet algorithm, *the number of connection losses on the geographical cells level remained stable* — the number of lost cell connections may grow by only 15% when the number of nodes increased from 30 to 100.

Encouraged by this observation, we perform another experiment (Fig. 3) in which we increase the number of nodes in the network and we observe the connection losses. The nodes speed is fixed to 20 m/s.

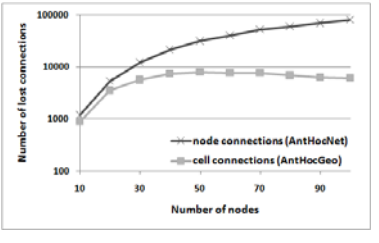


Fig. 3. Total number of lost connections during a simulation vs. nodes number; nodes speed: 20 m/s

The outcome of the simulation confirms our earlier expectations. The number of lost connections on the nodes level grows rapidly with the increased connectivity in the network, whereas the number of lost connections on the locations level remains stable.

The results may explain why the proposed approach can be beneficial in high connectivity scenarios: the generalized model used by AntHocGeo reduces the complexity of the structure of connections in the network while providing more robust connections on the locations level.

5.2 The Overhead

The increased robustness of the underlying learning model has a positive impact on the size and structure of the overhead generated by the ant routing mechanism.

We present the outcome of two experiments (Fig. 4) in which we increased either the nodes speed or the number of nodes in the network. We observe the number of bytes transmitted by the routing mechanism and focus only on one type of routing packets, namely the failure notification packets. These packets are directly connected with lost connections in the network [7]. We compare the results of AntHocNet and AntHocGeo algorithms.

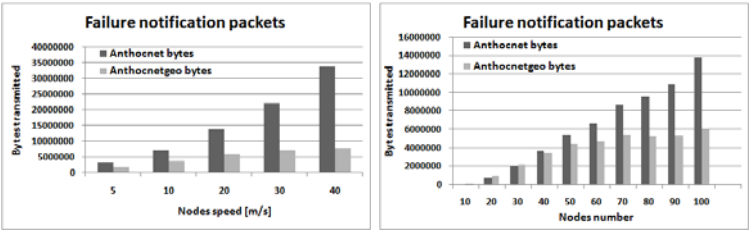


Fig. 4. Volume of failure notifications vs. nodes speed; 100 nodes in the network (left). Volume of failure notifications vs. nodes number; nodes speed: 20 m/s (right).

As the rate of topology changes grows (Fig. 4, left), the number of failure notification messages generated by AntHocNet increases significantly . At the same time, the number of failure messages generated by AntHocGeo grows moderately. The similar effect may be observed when increasing the number of the nodes in the network (Fig. 4, right). While AntHocNet generates more and more failure messages, the number of failure notifications produced by AntHocGeo remains stable for the number of nodes exceeding 50.

This results are a straightforward effect of increasing the stability of connections seen by the learning mechanism, observed and discussed in Sec. 5.1. The decreased number of failure notifications implies the reduction of the total routing overhead and consequently, improve the overall performance of the routing mechanism.

5.3 Overall Performance

Finally, we demonstrate the benefits of applying the generalized model of connections together with the ant routing mechanism (Fig. 5). We perform two experiments: in the first one we increase the rate of topology changes and in the other we increase the number of nodes in the network.

We compare AntHocNet and AntHocGeo algorithms. In addition, we show the results obtained by AODV [13] as a reference. We observe the average end-to-end delay and the delivery ratio.

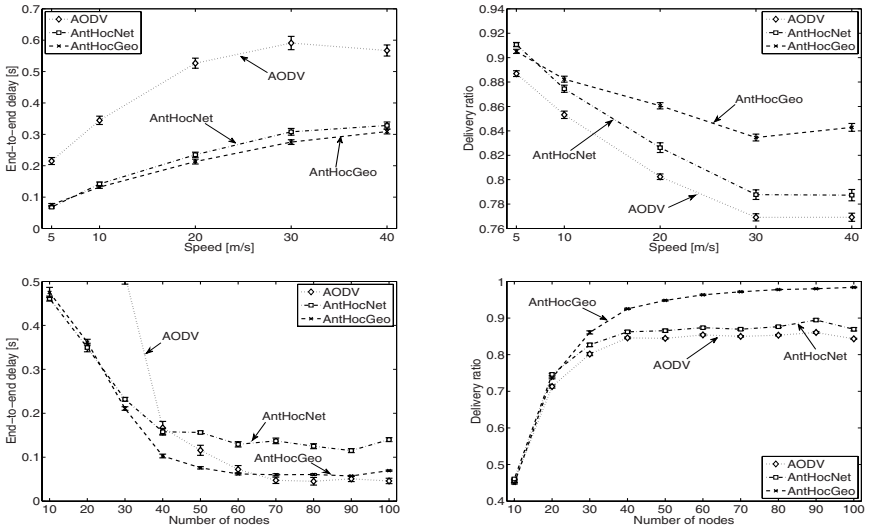


Fig. 5. The average packet delay (upper left) and the delivery ratio (upper right) vs. node speed. The average packet delay (lower left) and the delivery ratio (lower right) vs. node density. Random Waypoint mobility.

The results show that AntHocGeo outperforms the competitors both in terms of delays and delivery ratios. The benefits of employing the generalized model of connections are particularly notable under increased connectivity in the network and under high dynamics of network's topology changes.

6 Conclusions

In this paper, we introduced and analyzed the generalized model of connections in ad-hoc networks. We defined connections on a geographical locations level and demonstrated higher robustness of this approach.

The robustness of connections is crucial as far as the effectiveness of ant routing is concerned. A connection loss often implies a significant modification of the routing policy and the learning mechanism needs to adapt its policy to the new structure of connections. Furthermore, stable connections allow to reduce the overhead generated by the ant routing mechanism.

Our experiments showed that the robust model of connections improves the performance of ant routing in terms of delays and delivery ratio. The benefits of applying our approach are particularly visible under the increased connectivity in the network and under high dynamics of the network topology changes.

Acknowledgments

The scientific work funded as a research project by finance for scientific research in the years 2009-2010.

References

1. Abolhasan, M., Wysocki, T., Dutkiewicz, E.: A review of routing protocols for mobile ad hoc networks. Ad Hoc Networks 2(1), 1–22 (2004)
2. El-Nabi, T.H.A., Ahmed, A.: Modeling and simulation of a routing protocol for ad-hoc networks combining queuing network analysis and ant colony algorithms. PhD thesis, Universitt Duisburg-Essen (2005)
3. Di Caro, G., Ducatelle, F., Gambardella, L.M.: Anthocnet: An adaptive nature-inspired algorithm for routing in mobile ad hoc networks. European Transactions on Telecommunications 16, 443–455 (2005)
4. Hong, X., Xu, K., Gerla, M., Angeles, L.C.A.: Scalable routing protocols for mobile ad hoc networks. IEEE network 16(4), 11–21 (2002)
5. Iwata, A., Chiang, C.c., Pei, G., Gerla, M., Chen, T.w.: Scalable routing strategies for ad hoc wireless networks. IEEE Journal on Selected Areas in Communications 17, 1369–1379 (1999)
6. Kalaavathi, B., Madhavi, S., VijayaRagavan, S., Duraiswamy, K.: Review of ant based routing protocols for manet. In: International Conference on Computing, Communication and Networking, ICCCn 2008, December 2008, pp. 1–9 (2008)
7. Kudelski, M., Pacut, A.: Geographical cells: Location-aware adaptive routing scheme for ad-hoc networks. In: EUROCON 2007. The International Conference on "Computer as a Tool", September 2007, pp. 649–656 (2007)
8. Kudelski, M., Pacut, A.: Learning methods in ad-hoc networks: a review. In: Evolutionary Computation and Global Optimization, Prace Naukowe Politechniki Warszawskiej, Elektronika z, vol. 160, pp. 153–163 (2007)
9. Kudelski, M., Pacut, A.: Ant routing with distributed geographical localization of knowledge in ad-hoc networks. In: Giacobini, M., et al. (eds.) EvoWorkshops 2009. LNCS, vol. 5484, pp. 111–116. Springer, Heidelberg (2009)
10. Liao, W.-H., Sheu, J.-P., Tseng, Y.-C.: Grid: A fully location-aware routing protocol for mobile ad hoc networks. Telecommunication Systems 18(1-3), 37–60 (2001)
11. ns2. The network simulator, http://www.isi.edu/nsnam/ns
12. Pacut, A.: Stochastic modeling at diverse scales: from Poisson to network neurons. Oficyna Wydawnicza PW, Warsaw (2000)
13. Perkins, C., Royer, E.: Ad-hoc on-demand distance vector routing. In: Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications, pp. 90–100 (1997)
14. Royer, E.M., Toh, C.-K.: A review of current routing protocols for ad hoc mobile wireless networks (1999)

Using Code Bloat to Obfuscate Evolved Network Traffic

Patrick LaRoche, Nur Zincir-Heywood, and Malcolm I. Heywood

Faculty of Computer Science, Dalhousie University
Halifax, Nova Scotia, Canada
{plaroche,zincir,mheywood}@cs.dal.ca

Abstract. In this work, we investigate the ability of genetic programming techniques to evolve valid network patterns, while avoiding detectability by obfuscating the intent of the traffic. In order to validate our system's capabilities, we choose to evolve a port scan attack while running the packets through an Intrusion Detection System (IDS). In turn, the evolutionary process uses feedback such that it minimizes the alarms raised while port scanning across a network range. Results build off of previous work allow us to further analyze and understand what the role of introns, code bloat, play in the systems ability to reduce the detectability of it malicious behaviour.

1 Introduction

Vulnerability testing, a mechanism in which one attempts to discover weaknesses in a security mechanism, is done both on the part of the attackers and the protectors. Fuzzing, a form of vulnerability testing where one explores the boundaries and variations of a systems designed input parameters, allows the "fuzzer" to discover the weaknesses of the targeted system. In this case, the aim of the fuzzer is to analyze the vulnerability and attempt to remedy the situation before it becomes exploited. Such an exploration is inherently limited by the breadth and depth of the expert's knowledge in the system. In this work, we apply paged based Linear Genetic Programming (GP) [1,2] techniques in order to discover new variants of known vulnerabilities. By harnessing the natural exploratory nature of GP techniques, our work aims to start with base domain knowledge, and allow the GP to explore outside the realm of what an expert might conceive. In the work described here, we examine how GP learns to take advantage of introns (code bloat) to hide its desired network behaviour, reducing detectability. Also of importance, we look at the competition during evolution between attacking and remaining undetectable. By employing GP techniques, we are allowing for the discovery of mechanisms in which one can obfuscate the real intent and avoid detection.

Previous work has shown the validity of evolutionary techniques by exploring mimicry attacks at the system call level against host based IDSs [3]. The "mimicry' aspect is that the actual attack consists of not only the core sequence,

C. Di Chio et al. (Eds.): EvoApplications 2010, Part II, LNCS 6025, pp. 101–110, 2010.

but also functionality surrounding it, which is used to obfuscate that actual attack. In [4], authors implemented a preliminary work towards port scans.

In this work, we explore attacks at the network level that require the generation of network packets. As a validation of our technique, we have chosen to mimic a port scan attack against a block of hosts and ports, which are protected by the Snort IDS [5]. We focus on analyzing how GP achieves the attack, not the framework of the attack itself, which was analyzed in [4] . To this end, the proposed system evolves proper TCP/IP packets, including header values, such that it not only succeeds in creating port scan attacks, but also remain stealthy and evades a well know IDS, Snort [5].

The rest of this paper is organized as follows; Section 2 and 3 describe background information on both the networking and evolutionary methods, respectively. Section 4 details the goals and the design of the experiments and presents their results. Finally, conclusions are drawn and future work is discussed in Section 5.

2 Background

This paper explores the use of code bloat for vulnerability testing. To achieve this, we employe GP in order to evolve malicious network behaviour by evolving TCP/IP network packets. The resulting network traffic is then used to perform fuzzing at the transport level. This allows us to investigate the behaviour of the individual solutions, comparing to known attack behaviours.

2.1 Port Scans

The prevalence of port scanning, a method many employ to ascertain their next victims [6] continues to be on the rise [7]. Due to this increase, and the direct correlation with network attacks [8], many detection mechanisms have been developed. The basic detection mechanism for port scanning logs the number of packets X sent to Y host ports in time T. This establishes a threshold that has a temporal dependency, counting the number of "events" in a specific time frame. Snort, which is an open source IDS, implements such a mechanism.

One can classify port scans into 4 generic models, in all cases the attacker is attempting to detect whether a port is open, closed, or managed. In our work we allow the GP to evolve solutions that fit into any one of these models. A high level summary of these models is given below:

1. Basic - The attacker scans a series of ports sequentially, attempting a full TCP connection.
2. Random - The attacker behaves in the same manner as the basic attack, except randomizing the port numbers scanned.
3. Stealth SYN- Instead of trying to establish a full TCP connection, the attacker sends a TCP packet with the SYN flag set. This attack is more difficult to detect due to not initiating a full TCP connection, it simply starts one.

4. NULL, FIN, or "Xmas' scans- Similar to the stealth SYN scan, this attack does not initiate a TCP connection, sends packets with any combination of flags set, looking for RST packet to be returned (or not).

2.2 TCP/IP Packets

The intent of a port scan is to determine whether a port is open, closed or filtered. As such, we have selected the following TCP/IP features in order to control source and destination addresses and ports, as well as what control bits are set (also known as flags). All remaining values are calculated and implemented in a manner such that the packet fits requirements of the corresponding RFCs, 791 and 793. We use; IP Addresses, Port # (both source and destination) and Control Bits.

It is the above feature set, that GP evolves in creating packets. This provides the appropriate tool set in order to create useful packets for port scanning, but does not exclude the ability to create non-sensical packets as well. We recognize that some combinations of these features would result in a packet that would not make it to its destination, for this work we still consider this a valid packet.

2.3 Code Bloat

Code bloat, the rapid growth of Genetic Programs, without relationship to fitness, is well documented in literature [9,10]. Much work has been done on solving this growth issue [11]. Typically code bloat, in the form of introns or repeated sequences of operations that do not contribute to the overall solution, are considered to be extraneous and trimmed out either during the evolutionary process, or after the final generation, as they result in a solution that performs the desired task, but in an inefficient manner. However, in the realm of malicious network traffic, one could argue that the ability to run code that slows down the processing of the solution individual, might in fact not be a negative, but a positive quality. In our work, we focus on packets with temporal malicious behaviour, so introns that lead to additional computational time could perhaps be a benefit for avoiding detection by obfuscating the actual intent.

3 Evolutionary Model

Using the knowledge gained from past works [12,3] , authors have developed a system that allows the GP to create valid TCP/IP packets while responding to feedback from Snort [4]. As such, a multi-objective evolutionary model is defined to evolve individual solutions that create valid network packets, send said packets such that a port scan is completed, and avoid detection. These objectives are defined as follows:

- Solutions determine which ports are available on a range of hosts
- Solutions raise a minimal number of Snort Alarms

These two competing objectives are then summed into a scalar value. Thus the fitness evaluation not only assists us to rank the evolved attacks relative to the core attack but also provides feedback in terms of Snort's reaction.

3.1 Instruction Set

In selecting an instruction set, we must provide the mechanisms in which our system will be successful in recreating the core attack, but also be able to do so in a way that remains undetected. In our work, the attack consists of a series of packets. As described in Section 2, we are concerned with specific fields within a TCP/IP packet, as such we define an instruction set that allows our model to create valid TCP/IP packets, table 1 lists this set.

Table 1. Instruction Set for GP

Function	Parameter 1	Parameter 2
set_flags	Decimal 0-15	N/A
set_ips	Source IP	Destination IP
set_ports	Source Port	Destination Port
send_packet	N/A	N/A

An example of creating a packet would be to call each function call once, in any order, with the last function being "send_packet", lines 1-4 in Fig 3.1. Each function call is considered a "tick", thus we do not give the model an explicit "NOP" (No OPeration) or "wait" function in which to use up time. However, it is plausible that a series of functions not involving a "send_packet" could exist to use up "ticks" (obfuscation), effectively adding a temporal component. An example of a "code bloat" scenario, Fig. 1, lines 5 and 6 show how the second call overrides the first before sending the packet in line 7. If the intent of the individual was to simply send the packet, then the first system call would be trimmed from the individual. However, in our case, we wish to harness these introns such that they allow the individual to either use up time (by using up processor cycles without sending a packet) or send multiple packets to non target hosts, shown in lines 7 - 13. This is a good example of how code bloat obfuscates the intent of the attacker (for this example, assume 134.129.1.1 is the target, i.e the victim).

3.2 Fitness Evaluation

In order to assign a fitness metric to our individuals' performance, we need to define what a successful port scan is, as well as incorporating feedback from the detector in order to minimize alarms raised. For the purpose of this work, we consider the scan to be successful if a specific destination "host:port" pair is found to be open during the scanning process. The specific details of which port and which host varies on the experiment, as described in Section 4.

Figure 1. Example Sequence

```
 1: set_flags(0)
 2: set_ips(192.168.120, 134.29.1.1)
 3: set_ports(80,22)
 4: send_packet()
 5: set_ports(1024,20)
 6: set_ports(22,31)
 7: send_packet()
 8: set_ips(192.168.120, 134.29.1.8)
 9: send_packet()
10: set_ips(192.168.120, 301.1.2.1)
11: send_packet()
12: set_ips(192.168.120, 134.28.1.2)
13: send_packet()
```

In order to provide useful feedback to the fitness evaluation we have developed a metric that defines how close IP addresses are to each other. Our assumption when assigning this metric is that one considers the proximity of two IP addresses based on what class network they are on, and how similar those networks are to each other. A IPv4 address consists of 4 octets expressed in decimal with period as the delimiter between them. This describes a 32 bit address space, allowing for $4,294,967,296(2^{32})$ possible network address. For our testing we evaluate the "closeness" by looking from the left most octet to the right and measuring the difference between each octet, favouring closeness in the left most octet (which represents the network address) versus the right most octet (which represents host address). Eq. 1 describes this, where the distance is between two IP addresses, $A.B.C.D$ and $a.b.c.d$. The result of this distance metric is that GP is encouraged to find IP addresses that have open ports and are on the same network. We then use this metric in computing the fitness of an individual, that is to say that an individual is rewarded for finding targets 'close' to actual targets. This encourages exploration of similar networks, allowing for eventual discovery of targets.

$$DistanceIP = |((A-a)*4)| + |((B-b)*3)| + |((C-c)*2)| + |(D-d)| \quad (1)$$

The second part of our fitness function, the feedback from the detector - Snort, is provided in real time as the solutions are evolving. The feedback is offered in the form of an alarm count, where the worst case scenario is the number of alarms greater than or equal to the number of packets sent during the scan. Similarly, the best case is zero alarms being triggered during the execution of an individual solution.

Thus, fitness function is then described as the combination of two metrics (Eq. 1 and 2), with each section being described as a percentage. In this case, the first objective is maximizing the percentage of packets sent that correspond

to a targeted open "host:port" pair, whereas the second objective is the inverse of the percentage of alarms raised by all packets sent, as shown in Eq. 2.

$$Fit = \frac{\#TargetIP:PortHit}{TotalTargetIP:Port} + \left(1 - \frac{\#Alarms}{TotalPacketsSent}\right) \quad (2)$$

Where *"# Target IP: Port Hit"* describes the number of target *"host:port"* pairs detected and *"Total Target IP: Port"* is the total number of open *"host:port"* pairs that were available for detection during the training process. *"# Alarms"* is the number of alarms triggered by Snort during the execution of the individual, and *"Total Packets Sent"* represents the total number of complete packets sent. Note that the size of the individual is not the same as the number of packets sent, as it requires at least four function calls to send one packet.

3.3 Evolutionary Model

In order to achieve our goals we have implemented a page based linear GP. The basis for such a representation is a (virtual) register machine [1,13,2]. Such a machine consists of a predefined set of general purpose registers, support for the execution of an instruction set and the ability to supply input features describing the state of the environment at any given time step. Thus, individuals in the population of possible solutions are defined using a fixed length representation, with each individual representing a linearly sequential set of instruction calls. Selection of individuals between each generation is performed using a steady state tournament over 4 individuals. Search operators are applied to all in the tournament and the resulting two best performers replace the worst two. The search operators employed in this work are:

- Crossover, single point: A single page at a location (chosen with uniform probability) in one parent is interchanged with the page located at the same position in other parent.
- Swap Selector: Two instructions are selected in an individual and swapped, individual length remains fixed.
- Instruction-wise Mutation: For each instruction, test for application of mutation operator. If mutation is to be applied, another instruction from the complete instruction set is chosen with uniform probability and used as replacement.

Table 2. Parameters for Evolutionary Model

Parameter	Value	Parameter	Value
Population	1000	Mutation	0.5 with linear decay
Page Count	100	Swap	0.5
Page Size	6	Crossover	0.9
Tournament Size	4	Stop Criteria	100 000 Tournaments

In this work, we have employed instruction-wise mutation. With the linear annealing schedule, the probability of mutation linearly decreases over the generation count, reducing it to zero for the last iteration. Table 2 lists the specific evolutionary model parameters used during our experiments.

4 Experiments and Results

In these experiments, we explore the relationship between the available search space and the size of the target list, where the search space consists of all possible IP addresses while the target list varies in size, building off of the previous work's success. The experiments consist of three separate target lists, only used as a validation to the fitness of an individual, versus the entire IPv4 range. The first experiment consists of 10 targets, the second 100 and the final experiment being 1000 hosts. All three experiments are run 30 times using a different initial seed. Thus, we run 90 experiments in total.

Experiments are performed such that there is a machine originating the attacks (running the GP) and a machine acting as a monitor for traffic generated by the attack machine. Snort is is deployed on the monitor, seeing all the traffic. Snort was configured such that all packets received are monitored and inspected for malicious behaviour. As such this testbed mimics the attacker and the targeted machines in a controlled manner, allowing us to examine alarms triggered solely by our work. It should be noted here that we use the default Snort configuration, and set pre-processor specific variables for the most robust port scan detection possible. Specifically, the flow-portscan pre-processor was enabled, and the sfportscan preprocessor was set to the highest scan level available.

To assign a metric value to the performance of our system we rank our fitness out of 100 (perfect). A 'perfect' solution would be a solution that sent a packet to each target machine without triggering any alarms. Also of importance, in order to realize the usage of code bloat towards obfuscating network traffic. This is measured by comparing the number of function calls (total size) versus the number of packets sent to the network, thus showing what percentage of an individual results in a packet being sent. We list the best solution as well as the average of each target list in Table 3.

Also of interest is the tradeoff between the two competing metrics we use in our fitness evaluation, namely the metric of being able to find targets

Table 3. Results

	10 Targets	100 Targets	1000 Targets
Best Solution's Fitness	91.57	87.39	85.47
Best Solution's Length	84	42	30
Best Solution's Packets	65	38	29
Mean Population Solution Fitness	89.68	86.73	86.13
Mean Population Solution Length	110.00	51.20	36.16
Mean Population Packets Send	55.56	44.25	33.29

(open ports on hosts) versus staying stealthy. Moreover, how this develops over the evolutionary process is also important to understand the effect of code bloat in our experiments. Fig 2 presents the two components (finding targets, remaining stealthy) of the fitness for all experiments. By examining the fitness evaluation of each objective (component of the fitness function) separately, we can see how they evolve over each generation. Thus, Fig. 2 shows that the exploratory target objective evolves slowly at a linear rate, meanwhile the stealthy objective evolves exponentially. This difference in learning rates has led to two further experiments where we modified the fitness function and re-ran the "100 Target" experiments again. In this case, the fitness function was altered such that in one set of experiments once the IDS fitness component reached 90% we started tracking the target fitness (results shown as as Alt #2) . 'Alt #1' shows the other additional experiment in which the Target fitness was weighted more (2 to 3) then the IDS fitness to encourage more improvement on the seemingly more difficult objective. Average results of each experiment at a given generation count are shown in Fig 2.

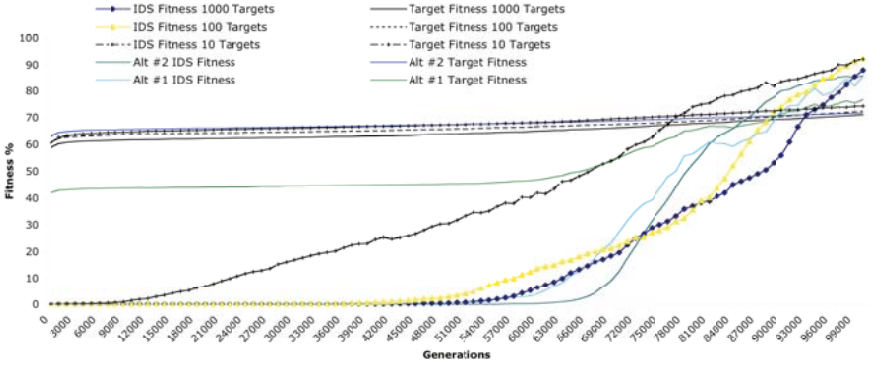


Fig. 2. Fitness over Generation Count

In both additional experiments we see an improvement in the rate in which the target fitness is learnt, with the alternative version #1 showing an actual non linear improvement after approximately 60000 generations. Further work in this area needs to be performed, but some promise is shown.

4.1 Analysis

The system implemented in this work needed to concur two problems concurrently, scan a number of unknown targets to find a small subset that had available services while also remaining stealthy in its activity. The system used feedback from the IDS (Snort) as well as distance metrics to determine how close it was to finding the available targets. In analyzing our results the system does indeed learn to achieve this by achieving the best solution fitness values of 91.57%,

87.39% and 85.47% in the three sets of experiments, Table 3. In looking at the size of the solutions as well as the breakdown of the fitness metrics, we see that the system is learning which problem is easier to solve: (i) finding all the targets, or (ii) avoiding detection. In all three sets of experiments, we can see that the avoiding of detection is the simpler problem to solve, Fig 2. The apparent difficulty of the problem can also be seen in the exponential increase in fitness with regard to detection avoidance versus the slow linear growth of the fitness associated with finding the targets. We explored this further by implementing two additional sets of experiments where we modified the relationship between the two objectives, first by artificially capping the IDS fitness at 90% until the Target Fitness improves, then by weighting the Target Fitness more then the IDS fitness. In both experiments, we show that we can further improve our results to best solution fitness values of 92.8% and 93.2% respectively, Fig 2. As stated earlier, the underlying motivation of our work is to examine the GP's ability to obfuscate it's core network intent using code bloat. To this end, we observe that the GP focuses on sending many packets, to many sources, while using a smaller amount of genes to change other packet header values. This is also an appropriate method for avoiding detection, as the GP is sending many packets, to many sources, raising no flags in Snort (as they are not being sent all to a specific IP range or port addresses, Snort does not flag this as an attack).

5 Conclusion and Future Work

We have demonstrated the ability for GP to generate network traffic, while adding complexity to the problem by increasing the search space to real-life scenarios. To achieve this, we modified the fitness evaluation of [4] by adding a distance metric between targeted IP addresses, as well as allowing the GP to explore the complete IP address space. We performed 90 experiments with the new fitness function, where the malicious user would locate a small, medium, and large number of targets during the scanning process while remaining stealthy in its activity. Then, the ability for GP is explored to learn how it obfuscates the network traffic through the use of code bloat. The resulting solutions did not use code bloat in methods we had expected. Instead, solutions were focusing on maximizing packets sent by an individual to a large set of targets, probably due to the larger search space required in these experiments. However, the solutions remain stealthy in their activities. On one hand it is not surprising that finding 10 target hosts out of the full IPv4 range is difficult, but on the other hand we did not expect the stealthy component to be relatively easier to solve. In future work, we plan to further explore this apparent disparity in order to improve our system while also applying it to further network activities.

Acknowledgment

This research is sponsored by the granting agencies NSERC and MITACS as well as the industrial partners TARA Inc. and SwissComm Innovations SA.

References

1. Heywood, M.I., Zincir-Heywood, A.N.: Dynamic page based crossover in linear genetic programming. IEEE Transactions on Systems, Man, and Cybernetics: Part B - Cybernetics 32(3), 380–388 (2002)
2. Nordin, P.: A compiling genetic programming system that directly manipulates the machine code. In: Kinnear Jr., K.E. (ed.) Advances in Genetic Programming, pp. 311–331. MIT Press, Cambridge (1994)
3. Kayacik, H.G., Heywood, M.I., Zincir-Heywood, A.N.: Evolving buffer overflow attacks with detector feedback. In: Giacobini, M. (ed.) EvoWorkshops 2007. LNCS, vol. 4448, pp. 11–20. Springer, Heidelberg (2007)
4. LaRoche, P., Zincir-Heywood, N., Heywood, M.: Evolving tcp/ip packets: A case study of port scans. In: Second IEEE Symposium on Computational Intelligence for Security and Defense Applications (2009)
5. Snort. org: Snort ids (March 2009)
6. Ashfaq, A.B., Robert, M.J., Mumtaz, A., Ali, M.Q., Sajjad, A., Khayam, S.A.: A comparative evaluation of anomaly detectors under portscan attacks. In: Lippmann, R., Kirda, E., Trachtenberg, A. (eds.) RAID 2008. LNCS, vol. 5230, pp. 351–371. Springer, Heidelberg (2008)
7. Symantec: Global Internet Security Threat Report trends for July-December 2007, vol. XIII, Symantec (2008)
8. Panjwani, S., Tan, S., Jarrin, K.M.: An experimental evaluation to determine if port scans are precursors to an attack. In: DSN 2005: Proceedings of the 2005 International Conference on Dependable Systems and Networks, pp. 602–611. IEEE Computer Society, Washington (2005)
9. Soule, T., Professor, M., Foster, J.A., Foster, J.A., Alves-foss, J., Frenzel, J.F., Frincke, D., Jacobsen, R.T., Shreeve, J.M.: Code growth in genetic programming (1998)
10. Koza, J.R.: Genetic programming as a means for programming computers by natural selection. Stat. Comput. 4, 191–198 (1994)
11. Nordin, P., Francone, F., Banzhaf, W.: Explicitly defined introns and destructive crossover in genetic programming (1995)
12. Kayacik, H.G., Heywood, M., Zincir-Heywood, N.: On evolving buffer overflow attacks using genetic programming. In: GECCO 2006: Proceedings of the 8th annual conference on Genetic and evolutionary computation, pp. 1667–1674. ACM, New York (2006)
13. Huelsbergen, L.: Toward simulated evolution of machine language iteration. In: Koza, J.R., Goldberg, D.E., Fogel, D.B., Riolo, R.L. (eds.) Genetic Programming 1996: Proceedings of the First Annual Conference, Stanford University, CA, USA, pp. 315–320. MIT Press, Cambridge (1996)

ABC Supported Handoff Decision Scheme Based on Population Migration

Xingwei Wang[1], Hui Cheng[2], Peiyu Qin[1], Min Huang[1], and Lei Guo[1]

[1] College of Information Science and Engineering, Northeastern University, Shenyang 110004, China
[2] Department of Computer Science, University of Leicester, Leicester, United Kingdom
wangxw@mail.neu.edu.cn

Abstract. In this paper, a handoff decision scheme is proposed with always best connected (ABC) supported. It comprehensively considers the status of the access network, the application QoS requirement, the preference of the user over the coding system of the access network, the preference of the user over the access network provider, the current speed of the terminal, and the residual electric quantity of the terminal. Based on the population migration algorithm (PMA) and gaming, it aims to find the Pareto-optimal solution under Nash equilibrium between the user utility and the network service provider utility. The simulation experiments demonstrate that the proposed scheme is effective.

Keywords: access network, handoff decision, always best connected, population migration algorithm.

1 Introduction

Next generation Internet (NGI) is becoming an integrated network environment which consists of multiple heterogeneous sub-networks. It is possible that a user will face multiple heterogeneous and diversified access networks which are all available at the same time. The user can be always best connected (ABC) [1] with the NGI in both the session initialization and on-going period. In such environment, there are two types of handoff, i.e., vertical handoff and horizontal handoff [2]. For vertical handoff, its decision is much more complicated than horizontal handoff since how to select an optimal access network from multiple heterogeneous ones is really tough. The concept of "best" itself is fuzzy because there are quite a few issues to be considered. Currently, under the commercialized operational environment of the networks, ABC cannot be achieved by the efforts at the user side only. Instead, it needs to consider the benefits at both the user side and the network service provider side. Another problem is that the application QoS requirements are very fuzzy and cannot be accurately quantized. The handoff decision needs to support the processing of fuzzy information. Similarly, to support ABC should not result in frequent handoff and should avoid the ping-pong effect. Therefore, the problem to make a handoff decision with ABC supported is very complicated and it can be summarized as follows. There are N mobile terminals waiting for handoff and they are covered by M access networks simultaneously. By considering the application QoS requirements, the cost the user is willing to pay, the user preference, the status of the

C. Di Chio et al. (Eds.): EvoApplications 2010, Part II, LNCS 6025, pp. 111–120, 2010.

mobile terminal, and the status of the access networks, an optimal solution is expected which assigns N terminals to M access networks and aims to achieve the Pareto-optimal under Nash equilibrium between the utilities of both the user and the network service provider. This problem is not only a fuzzy multi-objective optimization decision problem but also NP-Complete, selecting the best solution from M^N candidates. Thus, effective heuristics or optimization algorithms are necessary to solve it.

Quite a few researches have been done on the decision schemes of selecting access networks under vertical handoff scenarios [2]. However, they do not consider comprehensively the handoff decision problem with the ABC supported. In this paper, the knowledge of fuzzy mathematics and microeconomics is exploited to describe the application types, QoS requirements, the access networks and mobile terminals. Then a handoff decision scheme is proposed with the ABC supported. By considering multiple factors, it tries to find an optimal handoff solution of assigning N terminals to M access networks based on population migration algorithm (PMA) [3]. With the help of gaming, Pareto optimum [4] under Nash equilibrium [4] between the user utility and the network provider utility is achieved or approached for the found solution.

2 Model Description

2.1 Application Type, QoS Requirement and Its Fuzzy Degree

Based on the DiffServ [5], assuming that there are K different application types in NGI, and the set of application types is $ATS = \{AT_1, \cdots, AT_K\}$. In this paper, six QoS parameters are considered, namely, bandwidth BW, delay DL, delay jitter JT, bit error rate BE, packet loss rate PL and security level SL. Since the users cannot accurately describe the desired QoS requirements for the applications and the network cannot maintain the rigid single QoS value, in this paper the intervals are used to describe the QoS parameters and the fuzzy degrees are used to describe how fuzzy are the values of the QoS parameters. The mapping between the application types and the values of the QoS parameters is shown in Formula (1). For the application type AT_i, the fuzzy degrees of the six QoS parameters, i.e., FB_i, FD_i, FJ_i, FE_i, FL_i and FS_i, are defined, with FB_i shown in Formula (2) and others similar to it, $i = 1, \cdots, K$.

$$AT_i \rightarrow <[BW_i^l, BW_i^h], [DL_i^l, DL_i^h], [JT_i^l, JT_i^h], [BE_i^l, BE_i^h], [PL_i^l, PL_i^h], [SL_i^l, SL_i^h]> \quad (1)$$

$$FB_i = \frac{BW_i^h - BW_i^l}{BW_i^h + BW_i^l} \quad (2)$$

2.2 Access Network Model and Terminal Model

For an access network, the following parameters are used to describe it: j is the access network number, $1 \le j \le M$, where M is the total number of the access networks covering the terminals waiting for handoff; $PI_j \in PIS$ is the identifier of the access network provider, where $PIS = \{PI_1, PI_2, \cdots, PI_{|PIS|}\}$ is the set of the identifiers of all the access network providers; $TI_j \in TIS$ is the identifier of the access network type,

where $TIS = \{TI_1, TI_2, \cdots, TI_{|TIS|}\}$ is the set of the identifiers of all the access network types; CA_j and MV_j are the coverage area and the highest mobility speed of the access network type TI_j, and TI_j has one-to-one mapping with CA_j and MV_j; $CS_j \subseteq CIS$ is the set of coding systems supported by the access network j, where $CIS = \{CD_1, CD_2, \cdots, CD_{|CIS|}\}$ is the set of coding systems supported by all the access networks; $NAS_j \subseteq ATS$ is the set of application types supported by the access network j; FR_j is the frequency range supported by the access network j; TB_j is the total bandwidth provided by the access network j; AB_j and $AB_j^{\min}$ are the current available bandwidth provided the access network j and its lower bound, when AB_j is lower than $AB_j^{\min}$, the performance of the access network j decreases dramatically and in this case the terminal handoff request should not be accepted, this could help prevent the handoff from the ping-pong effect; TP_j is the lowest signal strength emitted by the access network j, it is the lowest signal strength emitted within the coverage area of the access network; ct_j is the cost of one unit bandwidth per unit time of the access network j; pr_{ij} is the sale price of one unit bandwidth per unit time of the access network j for the application with type AT_i, it is determined by the basic price pb_j and tuning price pf_{ij}, as shown in Formula (3). To encourage the rational use of the bandwidth, pb_j is determined by the access network provider and is divided into the low-rate, the flat-rate and the high-rate pricing intervals [6]. Let η_j denote the load rate of the access network j, and η_j is calculated in Formula (4).

$$pr_{ij} = pb_j + pf_{ij} \qquad (3) \qquad\qquad \eta_j = 1 - \frac{AB_j}{TB_j} \qquad (4)$$

When $\eta_j \le \eta_j^0$, the access network j is at the low load and falls into the low-rate pricing interval with pb_j determined by Formula (5). When $\eta_j \ge \eta_j^1$, it is at the high load and falls into the high-rate pricing interval with pb_j determined through the auction [7]. In other cases, it is at the middle load and falls into the flat-rate pricing interval with pb_j determined by Formula (6).

$$pb_j = \begin{cases} pb_j^{\min} & \eta_j < \eta_j^{\min} \\ \dfrac{A}{1 + \alpha \cdot \eta_j^{-\beta}} & \eta_j^{\min} \le \eta_j \le \eta_j^0 \end{cases} \qquad (5)$$

$$pb_j = \begin{cases} pb_j^{\max} & \eta_j^{\max} < \eta_j < \eta_j^1 \\ B \cdot (2 - e^{-\delta \cdot (\eta_j - \eta_j^0)^2}) & \eta_j^0 \le \eta_j \le \eta_j^{\max} \end{cases} \qquad (6)$$

Here, η_j^0 and η_j^1 are predetermined empirical values, $0 < \eta_j^0 < \eta_j^1 < 1$. $pb_j^{\min}$, pb_j^0 and $pb_j^{\max}$ are separately the starting price of the low-rate pricing interval, the starting price of the flat-rate pricing interval and the highest price of the flat-rate

pricing interval, $\eta_j^{\min}$, η_j^0 and $\eta_j^{\max}$ are their corresponding load rates, $pb_j^{\min} \le pb_j^0 \le pb_j^{\max}$. When $\eta_j^{\min} \le \eta_j \le \eta_j^0$, pb_j is similar as semi-Cauchy distribution [8]. When $\eta_j^0 \le \eta_j \le \eta_j^{\max}$, pb_j is similar as semi-Normal distribution [8]. Therefore, A , α , B and δ can be derived out.

pf_{ij} is determined by AT_i to reflect the effect on the sale price of the QoS requirement parameters except the bandwidth.

$$AT_i \rightarrow pf_{ij} \tag{7}$$

The following parameters are used to describe the terminal: t is the terminal number, $1 \le t \le N$, where N denotes the total number of terminals waiting for handoff; $TAS_t \subseteq ATS$ is the set of application types supported by the terminal; $MCS_t \subseteq CIS$ is the set of the coding systems supported by the terminal; WF_t is the working frequency of the terminal; RS_t is the lowest signal strength received by the terminal, representing the lower bound of the signal strength sensitive to the terminal; CV_t is the current moving speed of the terminal; RC_t and C_{tc} are the current residual electric quantity of the terminal and its threshold; HP_{ti} is the unit time unit bandwidth price that the terminal user is willing to pay for the application with type AT_i ; $PC_t = \{PC_{t1}, \cdots, PC_{tq}\}$ and $PP_t = \{PP_{t1}, \cdots, PP_{tm}\}$ are the preference sequences of the terminal user over the access network coding systems and access network providers separately, $PC_{tp} \in CIS$, $1 \le q \le |CIS|$, $1 \le p \le q$, $PP_{tn} \in PIS$, $1 \le m \le |PIS|$, $1 \le n \le m$,and the sequence is from high to low according to the preference degree.

2.3 Satisfaction Degree and Suitability Degree

Denote the actual QoS provided by the access network j for the application with type AT_i conducted by the user on terminal t as $< bw_{t_i j}, dl_{t_i j}, jt_{t_i j}, be_{t_i j}, pl_{t_i j}, sl_{t_i j} >$, the user evaluations on them are separately defined as $EB_{t_i j}$, $ED_{t_i j}$, $EJ_{t_i j}$, $EE_{t_i j}$, $EL_{t_i j}$ and $ES_{t_i j}$, with $EB_{t_i j}$ shown in Formula (8) and $ES_{t_i j}$ similar to it, also with $ED_{t_i j}$ in Formula (9) and $EJ_{t_i j}$, $EE_{t_i j}$ and $EL_{t_i j}$ similar to it. Accordingly, the user satisfaction degrees over them and the overall QoS are separately defined as $SB_{t_i j}$, $SD_{t_i j}$, $SJ_{t_i j}$, $SE_{t_i j}$, $SL_{t_i j}$, $SS_{t_i j}$ and $SQ_{t_i j}$, with $SQ_{t_i j}$ shown in Formula (10), with $SB_{t_i j}$ shown in Formula (11) and the other fives similar to it. If the sale price of one unit bandwidth per unit time of the access network j for the application with type AT_i is pr_{ij} , the price satisfaction degree of the user over the access network j is defined as Formula (12). The preference satisfaction degrees over the coding system and the provider of the access network of the terminal user are defined as Formula (13) and (14).

$$EB_{t_i j}=\begin{cases}0 & bw_{t_i j}<BW_i^l\\ \varepsilon & bw_{t_i j}=BW_i^l\\ \left(\dfrac{bw_{t_i j}-BW_i^l}{BW_i^h-BW_i^l}\right)^k & BW_i^l<bw_{t_i j}<BW_i^h\\ 1 & bw_{t_i j}\geq BW_i^h\end{cases} \quad (8)$$

$$ED_{t_i j}=\begin{cases}0 & dl_{t_i j}>DL_t^h\\ \varepsilon & dl_{t_i j}=DL_t^h\\ 1-e^{-\left(\frac{DL_t^h-dl_{t_i j}}{DL_t^h-DL_t^l}\right)^k} & DL_t^l<dl_{t_i j}<DL_t^h\\ 1 & dl_{t_i j}\leq DL_t^l\end{cases} \quad (9)$$

$$SQ_{t_i j}=\omega_1\cdot SB_{t_i j}+\omega_2\cdot SD_{t_i j}+\omega_3\cdot SJ_{t_i j}+\omega_4\cdot SE_{t_i j}+\omega_5\cdot SL_{t_i j}+\omega_6\cdot SS_{t_i j} \quad (10)$$

$$SB_{t_i j}=\frac{\alpha_1\cdot EB_{t_i j}}{\beta_1\cdot FB_i+1} \quad (11) \qquad SP_{t_i j}=\begin{cases}1 & pr_{ij}\leq HP_{ti}\\ 0 & \text{otherwise}\end{cases} \quad (12)$$

$$ST_{tj}=\begin{cases}\dfrac{1}{x^2} & CI_{tj}\in PC_t\\ 0 & \text{otherwise}\end{cases} \quad (13) \qquad SR_{tj}=\begin{cases}\dfrac{1}{y^2} & PI_{tj}\in PP_t\\ 0 & \text{otherwise}\end{cases} \quad (14)$$

Here, k is a constant, ε is a pure decimal far less than 1, α_{1-6} and β_{1-6} are for dimensional adjusting; ω_{1-6} are weights to reflect the relative importance of the six parameters to the application QoS requirement, and their sum is 1. Clearly, the higher value of the user satisfaction degree to the QoS is expected. $CI_{tj}\in CIS$ is the identifier of the coding system used by the terminal t when it makes the handoff into the access network j, and x is the sequence number of CI_{tj} in PC_t, $1\leq x\leq |PC_t|$. $PI_{tj}\in PIS$ is the provider identifier of the access network j into which the terminal t makes a handoff, and y is the sequence number of PI_{tj} in PP_t, $1\leq y\leq |PP_t|$.

If the current moving speed of the terminal is high, the access network with larger coverage area should be preferred since it will reduce the times of handoff operations. If the residual electric quantity of the terminal is low, the access network with smaller coverage area should be preferred since the terminal working in it consumes relatively low transmission and receiving power. The suitability degrees of the access network j to the current moving speed and the residual electric quantity of terminal t are defined as Formula (15) and (16) respectively.

$$SM_{tj}=\begin{cases}\dfrac{1}{z^2} & CV_t\leq MV_j\\ 0 & \text{otherwise}\end{cases} \quad (15) \qquad SW_{tj}=\begin{cases}\dfrac{1}{d^2} & RC_t\leq C_{tc}\\ 1 & \text{otherwise}\end{cases} \quad (16)$$

Here, z and d are the sequence number of TI_j when all the access network types are sorted by the size of their coverage area in the order from big to small and from small to big respectively, $1\leq z,d\leq |TIS|$.

2.4 Gaming Analysis and Utility Calculation

The players are the access network and the terminal. The access network j has two strategies a_1 and a_2. a_1 represents that the access network agrees to admit the terminal t while a_2 represents the opposite. The terminal t has two strategies b_1 and b_2. b_1

represents that the terminal is willing to be admitted by the access network j while b_2 represents the opposite. The gain matrices of the access network j and the terminal t are NP and TP, defined as follows.

$$NP=\begin{bmatrix} pr_{ij}-ct_j & pr_{ij}-ct_j \\ -\mu\cdot(pr_{ij}-ct_j) & -(pr_{ij}-ct_j) \end{bmatrix} \quad (17) \qquad TP=\begin{bmatrix} HP_{ti}-pr_{ij} & -\mu\cdot(HP_{ti}-pr_{ij}) \\ HP_{ti}-pr_{ij} & -(HP_{ti}-pr_{ij}) \end{bmatrix} \quad (18)$$

In both NP and TP, the rows correspond to a_1 and a_2, and the columns correspond to b_1 and b_2. The reason that np_{21} and np_{22} are negative is that if the access network refuses the handoff request of the terminal, it will lose the gain which is supposed to be obtained. μ is a penalty factor [9] denoting that if the access network refuses the terminal user who is willing to make a handoff into it, the negative effect will be brought to the user when it makes the decision about the future handoff. μ is greater than 1. The reason that tp_{12} and tp_{22} are negative is the same as in NP. If the in-equation in Formula (19) is satisfied, $\{a_{c^*},b_{d^*}\}$ is the strategy pair to achieve Nash equilibrium among the gains of the respective parties [4], $c^*,d^*,c,d=1,2$. The tuning factor Ω is defined in Formula (20) to reflect the effect of Nash equilibrium on utility.

$$\begin{cases} np_{c^*d^*}\geq np_{c^*d} \\ tp_{c^*d^*}\geq tp_{cd^*} \end{cases} \quad (19) \qquad \Omega=\begin{cases} 1 & \text{Nash equilibirum} \\ <1 & \text{otherwise} \end{cases} \quad (20)$$

$\Lambda=[\lambda_1 \quad \lambda_2 \quad \lambda_3 \quad \lambda_4 \quad \lambda_5 \quad \lambda_6]$ reflects the relative importance of the following aspects when selecting the access network: the application QoS requirement, the sale price of the access network for the unit bandwidth per unit time and the cost that the user is willing to pay, the preference over the coding system of the access network, the preference over the access network provider, the current moving speed of the terminal, and the current residual electric quantity of the terminal. λ_{1-6} can be determined empirically or by analytic hierarchy process (AHP) [10] and their sum is 1. $G_{t_ij}=[SQ_{t_ij} \quad SP_{t_ij} \quad ST_{tj} \quad SR_{tj} \quad SM_{tj} \quad SW_{tj}]^T$ is introduced. The terminal utility and the access network utility is defined as follows. Clearly, the large utility is expected.

$$uu_{t_ij}=\begin{cases} \Omega\cdot\Lambda\cdot G_{t_ij}\cdot\dfrac{(HP_{ti}-pr_{ij})}{HP_{ti}} & t \text{ enters } j \\ 0 & \text{otherwise} \end{cases} \quad (21) \qquad nu_{t_ij}=\begin{cases} \Omega\cdot\Lambda\cdot G_{t_ij}\cdot\dfrac{(pr_{ij}-ct_j)}{pr_{ij}} & t \text{ enters } j \\ 0 & \text{otherwise} \end{cases} \quad (22)$$

2.5 Mathematical Model

If there are N terminals covered by M access networks simultaneously, the handoff solution of assigning N terminals to M access networks should satisfy the following objectives to make the user utility and the access network provider utility achieve or approach the Pareto optimum under Nash equilibrium.

$$\text{maximize}\{uu_{t_ij}\} \quad (23) \qquad \text{maximize}\{nu_{t_ij}\} \quad (24) \qquad \text{maximize}\{\sum_{t=1}^{N}\sum_{j=1}^{M}uu_{t_ij}\} \quad (25)$$

$$\text{maximize}\{\sum_{t=1}^{N}\sum_{j=1}^{M} nu_{t_i j}\} \quad (26) \qquad \text{maximize}\{\sum_{t=1}^{N}\sum_{j=1}^{M}(uu_{t_i j} + nu_{t_i j})\} \quad (27)$$

3 Algorithm Description

PMA is a global optimization algorithm by simulating the population migration procedure in human society [3]. It is mainly consisted of the followings: people migrate in local area; the preferential region attracts the immigrants; people immigrate into the preferential region until the population pressure reaches a certain upper limit; people move out of the preferential region and look for new opportunities.

3.1 Solution Encoding and Its Attraction Force Function

For each terminal t, its available access network set AN_t is constructed by putting any access network j with $TAS_t \subseteq NAS_j$ and $WF_t \subseteq FR_j$ and $CS_j \cap MCS_t \neq \varphi$ and $CV_t \leq MV_j$ and $TP_j \geq RS_t$ and $(HP_{ti} \geq pr_{ij})$ and $(AB_j - BW_i^h) \geq AB_j^{\min}$ satisfied into $AN_t, t = 1,\cdots,N$. Then, for each AN_t, construct its corresponding available access network sequence AS_t. Here, AS_t has the same elements as in AN_t with the only difference that its elements are ordered with the ordering number starting from 0.

The position of a person represents one solution. Each solution is encoded in N-dimension vector. For an individual person q, his current position P_q is recorded as $< p_{q_1},\cdots,p_{q_N} >$, which represents one solution of assigning N terminals to M access networks. Here, p_{q_t} is the sequential number in AS_t which corresponds to the number of the access network assigned to the terminal t, $t = 1,\cdots,N$.

The attraction force function of P_q is denoted as $FT(P_q)$ and defined as follows.

$$FT(P_q) = \sum_{t=1}^{N}\left(\frac{1}{uu_{t_i \overleftarrow{p_{q_t}}}} + \frac{1}{nu_{t_i \overleftarrow{p_{q_t}}}}\right) \quad (28)$$

Here, $\overleftarrow{p_{q_t}}$ represents the (p_{q_t})th element in AS_t, which is the number of the access network assigned to the terminal t. Clearly, the less the $FT(P_q)$, the larger and the more balanced the user utility and the access network provider utility, and the utilities of the two parties approach or achieve Pareto optimum under Nash equilibrium more closely, thereby the corresponding handoff solution is better.

3.2 Algorithm Procedure

It is described as follows: (1) input N terminals waiting for handoff and M access networks covering them simultaneously; (2) construct the sequence of the available access networks AS_t for each terminal t, $t = 1,\cdots,N$; (3) set the iteration times IT and

the population size PS, denote the optimal solution set as ES and $ES = \varphi$, and denote the iteration times counter as it and $it = 0$; (4) generate PS initial positions of the population; (5) regard P_q as the centre of the qth region, set $P_q \pm \delta_q$ as the upper and lower boundaries of the qth region with $\delta_q =< \delta_{q_1},...,\delta_{q_N} >$ defined as Formula (29); (6) calculate $FT(P_q)$ by Formula (28), $q = 1,2,\cdots,PS$, record the minimum attraction force function value as FT^* and put the corresponding P_q into ES, the P_q corresponding to FT^* is recorded as P_g^b, and if there are multiple ones corresponding to the same FT^*, one will be selected randomly; (7) set the migration times in the local region as MT; (8) set the migration times counter $mt = 0$; (9) the people move within their local regions, that is, everyone's position P_q is changed uniformly and randomly by Formula (30); (10) calculate $FT(P_q)$ by Formula (28), $q = 1,2,\cdots,PS$, record the minimum attraction force function value as FT^*, if $FT^* < FT(P_g^b)$, replace all the elements in ES with the P_q corresponding to FT^* and the P_q corresponding to FT^* is recorded as P_g^b (if there are multiple ones corresponding to the same FT^*, select one randomly), if $FT^* = FT(P_g^b)$, all the P_q corresponding to FT^* are put into ES and select one element from ES randomly as P_g^b; (11) $mt = mt + 1$; (12) if $mt < MT$, go to (9); (13) population immigrate to the preferential region, that is, uniformly and randomly regenerate positions for all the people within the preferential region centered at P_g^b and with the upper and lower boundaries $P_q \pm \delta_q$, use these new positions to replace the current positions; (14) same as (10); (15) reduce the preferential region, that is, $\delta_q = (1-\Delta)\cdot\delta_q$, $q = 1,2,\cdots,PS$, Δ is the reduction coefficient, $0 < \Delta < 1$; (16) if $\delta_{q_t}^{\max} > \alpha$, go to (13), α is the population pressure threshold value, $\delta_{q_t}^{\max}$ is the maximum value of δ_{q_t}, $t = 1,\cdots,N$; (17) move people out of the preferential region, that is, randomly regenerate the positions for all the people; (18) same as (10); (19) $it = it + 1$; (20) if $it < IT$, go to (8), otherwise output the solution corresponding to P_g^b as the optimal handoff decision, the end.

$$\delta_{q_t} = \frac{|AS_t|}{2\cdot PS} \quad (29) \qquad p_{q_t} = \lfloor |2\cdot\delta_{q_t}\cdot rd + (p_{q_t} - \delta_{q_t})| \rfloor \bmod |AS_t| \quad (30)$$

Here, $|AS_t|$ represents the number of elements in AS_t, rd is a random number, $q = 1,2,\cdots,PS$, $t = 1,\cdots,N$. The procedure of local population migration is simulated by (8)-(12); the procedure of people immigrating to the preferential region, that is, population centre following economic centre, is simulated by (13)-(16); the procedure of people moving away from the preferential region is simulated by (17).

4 Simulation Implementation and Performance Evaluation

The PMA based handoff decision scheme is implemented through simulation in network simulator 2 (NS2). Fig. 1 illustrates a hexagon cellular topology where three different types of access networks overlap. Based on this topology, the proposed handoff decision scheme, a greedy strategy based handoff decision scheme [11], and a multiservice vertical handoff decision scheme [12] (Just for simplicity, these three schemes are marked as Scheme 1, Scheme 2 and Scheme 3 separately below) are run 500 times. Under various number of terminals waiting for handoff, the mean values of the following evaluation metrics are compared: the user utility (UU), the access network provider utility (NU), the comprehensive utility (CU, CU=UU+NU), the ratio of the Pareto-optimal solutions under Nash equilibrium (PN), the QoS satisfaction degree (SQ), the price satisfaction degree (SP), the preference satisfaction degree over the coding system of the access network (ST), the preference satisfaction degree over the access network provider (SR), the suitability degree to the moving speed of the terminal (SM) , and the suitability degree to the residual electric quantity of the terminal (SB). The results are shown in Table 1.

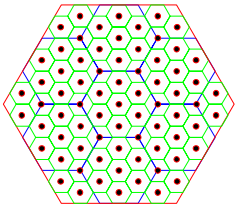


Fig. 1. Hexagon cellular topology

Table 1. Performance Comparison

P:G:M	number of terminals for handoff		
	5	10	50
UU	1.25:0.94:1.00	1.30:0.94:1.00	1.36:0.84:1.00
NU	1.30:0.93:1.00	1.34:0.94:1.00	1.75:1.02:1.00
CU	1.27:0.94:1.00	1.32:0.94:1.00	1.49:0.91:1.00
PN	1.24:1.00:1.00	1.31:1.00:1.00	1.44:1.07:1.00
SQ	0.59:0.40:1.00	0.58:0.38:1.00	0.44:0.31:1.00
SP	2.11:1.71:1.00	2.18:1.66:1.00	2.15:1.59:1.00
ST	1.44:1.07:1.00	1.43:0.96:1.00	1.62:0.95:1.00
SR	1.64:1.26:1.00	1.54:1.09:1.00	1.67:1.05:1.00
SM	1.87:1.27:1.00	1.91:1.21:1.00	2.26:1.36:1.00
SB	1.82:1.10:1.00	1.94:1.14:1.00	2.12:1.03:1.00

Scheme 1 is always better than Scheme 2 and Scheme 3 for the above metrics except SQ. The reason is that Scheme 1 aims to satisfy these metrics when searching the optimal solution. However, in terms of the SQ, Scheme 1 is worse than Scheme 3 but better than Scheme 2. The reason is that Scheme 3 considers only SQ when it makes the handoff decision while Scheme 2 does not consider it and Scheme 1 comprehensively considers various factors including the QoS.

5 Conclusion

In this paper, an ABC supported handoff decision scheme is proposed based on PMA. It is evaluated through simulation implementation in NS2. Compared to the existing schemes, the proposed one is effective. At present we are designing and evaluating the ABC supported handoff decision schemes based on particle swarm optimization and others. These efforts will help evaluate the performance of our proposed model under

different algorithms for further optimization in the model. And also we are focusing on the further applications of the proposed models and algorithms to evaluate their practicalities and thus improve their runtime performances.

Acknowledgments. This work is supported by the National Natural Science Foundation of China under Grant No.60673159, No.70671020 and No.60802023; the National High-Tech Research and Development Plan of China under Grant No.2007AA041201; the Specialized Research Fund for the Doctoral Program of Higher Education under Grant No.20070145017 and No.20070145096.

References

1. Gustafsson, E., Jonsson, A.: Always best connected. IEEE Wireless Communications 10(1), 49–55 (2003)
2. Kassar, M., Kervella, B., Pujolle, G.: An overview of vertical handover decision strategies in heterogeneous wireless networks. Computer Communications 31(10), 2607–2620 (2008)
3. Zhou, Y., Mao, Z.: A New Search Algorithm for Global Optimization: Population Migration Algorithm. Journal of South China University of Technology 31(3), 1–5 (2003) (in Chinese)
4. Song, C.X.: Modern Western Economics (Microeconomics). Fudan University Press, Shanghai (2004) (in Chinese)
5. Blake, S.: An architecture for differentiated services. IETF 2475 (1998)
6. Andrew, O.: Paris metro pricing: The minimalist differentiated services solution. In: 7th International Workshop on Quality of Service, pp. 159–161. IEEE Press, Piscataway (1999)
7. Maille, P., Tuffin, B.: Pricing the Internet with multibid auctions. IEEE/ACM Transactions on Networking 14(5), 992–1004 (2006)
8. Yang, L.B., Gao, Y.Y.: Theory and Applications for Fuzzy Mathematics, 3rd edn. South China University of Technology Press, Guangzhou (2001) (in Chinese)
9. Cao, X.R., Shen, H.X., Milito, R., Wirth, P.: Internet pricing with a game theoretical approach: concepts and examples. IEEE/ACM Transactions on Networking 10(2), 208–216 (2002)
10. Saaty, T.L.: The Analytic Hierarchy Process. McGraw-Hill Company, New York (1980)
11. Wang, X.D.: Computer Algorithm Design and Analysis, 2nd edn. Publishing House of Electronics Industry, Beijing (2005) (in Chinese)
12. Fang, Z., Janise, M.: Multiservice vertical handoff decision algorithms. EURASIP Journal on Wireless Communications and Networking, 1–13 (2006)

A Hyper-Heuristic Approach for the Unit Commitment Problem

Argun Berberoğlu[1] and A. Şima Uyar[2]

[1] Istanbul Technical University, Informatics Institute
[2] Istanbul Technical University, Department of Computer Engineering,
34490, Maslak, Turkey
{aberberoglu,etaner}@itu.edu.tr

Abstract. This paper introduces a hyper-heuristic approach for the Unit Commitment Problem (UCP). Tests are performed using benchmark data from literature and real-world data from the Turkish interconnected power network. The proposed hyper-heuristic and several methods applied previously to the UCP, are compared. Results show that the hyper-heuristic method achieves good results in all test sets. Furthermore, it is also a robust method for increased problem sizes without the need for parameter tuning. Based on the promising results, research will continue for further improvements.

Keywords: Unit commitment problem, hyper-heuristics, evolutionary algorithms, optimization.

1 Introduction

The Unit Commitment Problem (UCP) is a constrained optimization problem, whose aim is to determine start-up and shut-down schedules for a group of power generators over a given time period, so that the power generation costs are minimized while providing a forecasted amount of power for each hour.

An optimal scheduling of the generators allows for great economic savings with regard to cost and energy use. Therefore, the UCP has become an important research area and many approaches have been used to solve the problem. Lagrangian relaxation [1], dynamic-programming [2] simulated annealing [3], tabu search [4], branch and bound [5], priority lists [6], greedy algorithms [7] and evolutionary algorithms [8, 9,10,11,12,18] are among these approaches.

This paper proposes a hyper-heuristic method to solve the UCP. Hyper-heuristics [13] operate on a set of heuristics to choose one of them at each step of the search. This selection depends on either problem independent measures, such as the quality change of the solution when the selected heuristic is used, or on some probability distribution. Since hyper-heuristics do not require any domain knowledge to operate, they are easy to use over a wide range of applications. Also they require much less parameter tuning efforts than most approaches, e.g. evolutionary algorithms. The recent success of hyper-heuristic approaches in the domain of scheduling and timetabling problems, e.g. as in [19] has been a motivation for this study.

C. Di Chio et al. (Eds.): EvoApplications 2010, Part II, LNCS 6025, pp. 121–130, 2010.

2 The Unit Commitment Problem

The objective of the UCP is to minimize the power generation costs over a given time period. Power generation costs consist of fuel costs and start-up costs. Penalty factors are also considered when assessing the quality of a solution. These penalty values are the demand penalty, which occurs when the predefined hourly demand is not fulfilled by the candidate solution, and the up/down penalty, which is added to the objective function, when an up/down constraint is violated for at least one generator. The following parameters are used in the UCP formulation.

Table 1. Parameters used in the definition of the UCP

Parameter	Explanation
$P_i(t)$	generated power by unit *i* at time *t*
$F_i(p)$	cost of producing *p* MW power by unit *i*
PD(t)	power demand at time *t*
PR(t)	power reserve at time *t*
$CS_i(t)$	start-up cost of unit *i* at time *t*
$x_i(t)$	duration for which unit *i* has stayed online/offline since hour *t*
$v_i(t)$	status of unit *i* at time *t* (on-off)

Fuel cost depends on the power produced by each online generator for a given time slot. A major concern in the solution of the UCP is to meet the predetermined power demand for each time slot while keeping the generated power of each unit within its minimum and maximum values. For N generating units at time t, the objective function is given below.

$$\min \quad F_{total}(t) = \sum_{i=1}^{N} F_i(P_i(t)) \tag{1}$$

Subject to constraints:

$$\sum_{i=1}^{N} P_i(t) = PD(t) \tag{2}$$

$$P_i^{\min} \le P_i(t) \le P_i^{\max} \tag{3}$$

The second cost factor is the start-up cost which occurs when a generator changes its status from offline to online. This cost needs to be calculated according to the following formulation. The start-up cost does not only depend on the generator type, but it is also affected by the amount of time a generator has stayed offline.

$$CS_i(t) = \begin{cases} CS_{hot} & if \quad x_i(t) \le t_{coldstart} \\ CS_{cold} & otherwise \end{cases} \tag{4}$$

Another constraint in this problem defines the minimum up-time of a unit before it is turned off after becoming online and the minimum down-time before the unit is turned on after becoming offline. If this constraint is violated for any generator, a penalty cost, named as up/down penalty, is added to the power generation cost of the corresponding unit. The formulation in Eq.5 is used to calculate the up/down penalty.

$$\begin{array}{ll} if\ v_i(t)=1 & x_i(t-1) \geq t_{down} \\ else & x_i(t-1) \geq t_{up} \end{array} \tag{5}$$

Based on the fuel and start-up costs and demand and up/down penalty values, the UCP objective function for N units and T hours is defined as given below.

$$\min \quad F_{total} = \sum_{t=1}^{T} \sum_{i=1}^{N} \left[F_i(P_i(t)).v_i(t) + CS_i(t) \right] \tag{6}$$

Subject to constraints:

$$\sum_{i=1}^{N} P_i(t).v_i(t) = PD(t) \tag{7}$$

$$v_i(t).P_i^{\min} \leq P_i(t) \leq v_i(t)P_i^{\max} \tag{8}$$

$$\sum_{i=1}^{N} P_i^{\max}(t).v_i(t) \geq PD(t) + PR(t) \tag{9}$$

$$\begin{array}{ll} if\ v_i(t)=1 & x_i(t-1) \geq t_{down} \\ else & x_i(t-1) \geq t_{up} \end{array} \tag{10}$$

The fuel cost of generating p MW power for the i-th unit is calculated using the following equation. Fuel cost for a power generation unit i depends on three parameters, a_{0i}, a_{1i} and $a_{2i,}$ which are predetermined for each generator. The lambda iteration technique [14] uses this formulation to find the minimum cost for dispatching the amount of power to be generated by online units.

$$F_i(p) = a_{0i} + a_{1i}.p + a_{2i}.p^2 \tag{11}$$

3 Hyper-Heuristics

Heuristics, such as evolutionary algorithms, tabu search, greedy search, simulated annealing are used to solve complex optimization problems, but one of the major obstacles during the process is to adapt a heuristic to the corresponding problem, since it requires great test effort and experience about the problem. Therefore, it is

nearly impossible to develop a heuristic which is capable of solving a wide range of optimization problems. To overcome this, hyper-heuristic methods are introduced.

Hyper-heuristic methods perform their operations on heuristics rather than on candidate solutions [13]. They are used to select a heuristic from a set of heuristics at each step of the search without any prior knowledge of the problem instance, so that they can be successfully applied to a broad range of optimization problems.

A hyper-heuristic consists of two mechanisms. These are heuristic selection and move acceptance mechanisms [13]. Heuristic selection is responsible for selecting a heuristic according to feedback from the previous runs or according to a probability distribution, and the acceptance mechanism decides whether the new candidate solution survives into the next generation [13]. Heuristics may be divided into two groups as mutational heuristics and hill-climbers. Hill-climbers are used as local search techniques to increase the quality of the solution, whereas mutational heuristics are used to increase the diversity and to search in different areas of the solution space.

In a traditional hyper-heuristic framework, one of the low level heuristics is selected and applied to the candidate solution. After that, its fitness value is calculated and the acceptance mechanism decides whether to accept the solution or not.

4 Related Work on the UCP

In literature, there are many successful evolutionary algorithm applications to solve the UCP. These may be grouped into four different approaches.

In the first approach, a binary chromosome is used as the candidate solution to represent the on/off schedule of the power generation units. Genetic operators work on this chromosome. An iterative technique, such as lambda iteration, is used to determine the power generation amounts of the online units for each time slot, e.g. as in [8, 10, 12, 14, 18].

In the second approach, the chromosome consists of integers or floating point numbers. It represents the on/off schedule of the units. Integers can be either negative or positive to show the duration of the off or on states of the unit. Lambda iteration is used to determine the generated power by each online unit, e.g. as in [16, 17].

The third approach uses Lagrangian relaxation to solve the UCP, while evolutionary algorithms are used to update the Lagrangian multipliers, e.g. as in [1].

In the last approach, a floating point chromosome is used, where each gene shows the output power of the corresponding power unit. Evolutionary algorithms work only on online generators, which are not working at maximum capacity, and they improve the already dispatched power, e.g. as in [15].

5 Proposed Approach

In this paper, a hyper-heuristic technique is used to solve the UCP. A binary representation is used with a length of N*T, where N is equal to the number of units and T is equal to the number of time slots. The decision variables take on values as either 0 or 1 to indicate that the corresponding generator is off or on for this time slot. As the heuristic selection mechanism, the *permutation descent* method is used. In this

method, a random permutation of all heuristics is created. At each iteration, the next heuristic in the permutation is applied to the candidate solution. Improving heuristics are applied repeatedly, until no more improvement is seen. As move acceptance criteria, only improving moves are accepted. The heuristic selection and solution acceptance mechanisms used in this study are chosen from those tested in [13] based on the results of initial experimentation.

Seven heuristics are used during the search. These are mutation with a probability of 1/L, where L is the solution length, mutation with a probability of 2/L, swap-window [8], window-mutation [8], swap-mutation [8], swap window hill-climbing [8], and Davis bit hill-climbing [20] operators. The last three operators have hill-climbing capabilities. In [8] a genetic algorithm is proposed to solve the UCP. The heuristics used in this study are used in [8] as genetic mutation operators during the iterations. In this approach, each mutation operator has an adaptive application probability which takes on values within predetermined intervals. After initialization, these probabilities are adapted based on the current convergence status of the search.

```
create randomly an initial solution
calculate fitness value of the initial solution
create a random permutation of all 7 heuristics
while not end of iterations do
  Select the first heuristic in the permutation
  repeat
    repeat
      Apply the selected heuristic to the solution
      If (heuristic does not contain hill climbing)
        Apply Davis bit hill climbing
      Calculate fitness value
      If (fitness value is better)
        Accept the new solution
    until (the fitness value is no more improved)
    Select the next heuristic
  until (last heuristic in the permutation is applied)
end while
```

Fig. 1. Pseudo-code for the Proposed Hyper-Heuristic Approach

In the first step of the proposed hyper-heuristic approach, an initial solution is created randomly. After that, one of the seven heuristics is selected according to a predetermined permutation. If the selected heuristic does not contain hill-climbing capability, then Davis bit hill-climbing operator is applied to the solution after the selected heuristic completes its task [13]. In the next step, the fitness value of the resulting solution is calculated. If there is an improvement, the solution is accepted and the same operator is applied to this solution again; otherwise, the previous solution is not replaced with the new one and the next heuristic in the permutation is applied to the previous solution.

6 Experiments

The hyper-heuristic approach, explained in section 5, is tested on several benchmark problems taken from literature and also on a real-world data set obtained from the

Turkish interconnected power system. In the first part of the experiments, the performance of the hyper-heuristic is compared with the performance of the genetic algorithm proposed in [8]. The aim of this part is to show the benefits of incorporating the genetic operators used in [8] within the framework of a hyper-heuristic. In the next part, three benchmark problems are used and the results obtained using the hyper-heuristic are compared with results reported in literature. These benchmarks are referred to as System 1, System 2 and System 3 in the rest of the paper. System 1, taken from [10], consists of 10 units. System 2 and System 3 data are generated by repeating the data of System 1 two and four times respectively, as also done in [8, 10]. As a result, System 2 has 20 generators and System 3 has 40 generators. For all test systems, the time horizon is 24 hours. The effect of problem size on the performance of the methods is investigated through using data sets of increasing sizes. In the last part, real-world data is used to determine the efficiency of the hyper-heuristic approach with respect to others.

6.1 Parameter Settings

In System 1, and in the Turkish Interconnected Power System, as used in [18], the best, the average and the worst case values are reported over 20 runs of the program. For System 2 and System 3, as used in [1, 16], these values are reported over 15 runs of the program. In the fitness calculation, the demand penalty and the up/down penalty coefficients are set to a very high value as 100000, so that infeasible solutions can not have a better fitness than feasible ones. The number of allowed iterations for the hyper-heuristic per run is chosen as 1000, 5000, 10000, for System 1, System 2 and System 3 respectively. These values are determined empirically.

6.2 Experimental Results

In this section, several optimization methods are compared using System 1, System 2 and System 3 and the Turkish interconnected power network data sets. All results in the following tables are taken from literature. For some methods, the average and worst data are not available in literature; therefore, they are not given in this paper. The listed methods are abbreviated as follows:

LR1 and LR2 are the Lagrangian Relaxation methods as used in [8, 10]
GA1 is a standard genetic algorithm as used in [10]
GA2 is a genetic algorithm with special operators as used in [8]
GRA1 and GRA2 are the Greedy Randomized Search methods as used in [7]
MA and SMA are memetic algorithms as used in [10]
BDE1 and BDE2 are binary differential evolution methods as used in [12,18]
ES is an Evolutionary Strategies algorithm as used in [18]
SSGA is Steady-State Genetic Algorithm as used in [18]
HH is the hyper-heuristic approach proposed in this study.

In the first part of the experiments, the GA2 algorithm as used in [8], and the proposed HH algorithm are compared on three data sets. In GA2, premature convergence is not desirable, since it makes the search inefficient. Therefore, the search process is

monitored, and premature convergence is prevented by adjusting crossover and mutation probability rates. In the HH method, same genetic operators are used; however, the system does not need to be monitored. That is, HH does not require parameter tuning and adaptation. It simply selects a heuristic according to a permutation, which is created randomly in the beginning, and this heuristic is applied to the solution repeatedly, until it does not improve the solution candidate.

For System 1, both methods obtain very similar results, but for System 2 and System 3 HH obtains better results than GA2. In larger data sets, the difference between the performances of GA2 and HH becomes more significant.

Table 2. Results for System 1

Algorithm	Best	Average	Worst
LR1	565825	n/a	n/a
GRA1	565825	-	-
GA2	565825	-	570032
BDE2	565827	565965	566650
HH	565827	566121	567028
MA	565827	566453	566861
ES	565827	569199	571312
GA1	565866	567329	571366
BDE1	566166	-	-
SMA	566686	566787	567822
LR2	567663	n/a	n/a

In the second set of experiments, HH is run on three benchmark data sets and comparative results are reported in tables 2-4. If we look at these tables, we can see that most of the methods show inconsistency in their performances with different data sets. For instance, LR1 and GRA1 obtain the best result in System 1, but they achieve very poor results in the other two systems. Similarly, GRA2 obtains good results in System 1, but if the data set gets larger, its performance decreases significantly. However, the HH approach achieves consistent results over all data sets. In System 1, it obtains a very good result, which is slightly higher than the overall best result. The difference percentage between LR1 and HH is only 0.0003%. In System 2 and System 3, HH finds the overall best results.

Methods, including hybridization techniques, such as repair operators, specialized reproduction operators or hill-climbers perform significantly better on larger problems. From these results, it can be easily noticed, that SMA, MA, GA2 and HH obtain good results in larger data sets because of the usage of hybrid operators during the search. At each iteration, a hill-climbing operator is applied in HH; therefore, HH is scalable and can be adapted to different problem instances with varying data sizes as well. Additionally, HH does not contain sophisticated methods during heuristic selection and move acceptance steps. Applying such methods to HH can improve current results.

Table 3. Results for System 2

Algorithm	Best	Average	Worst
HH	1126231	1127553	1128831
GA2	1126243	-	1132059
GRA2	1126805	-	-
MA	1127254	1128824	1130916
GRA1	1128160	-	-
SMA	1128192	1128213	1128403
GA1	1128876	1130160	1131565
LR2	1129633	n/a	n/a
LR1	1130660	n/a	n/a

Table 4. Results for System 3

Algorithm	Best	Average	Worst
HH	2249099	2251528	2253099
SMA	2249589	2249589	2249589
LR2	2250223	n/a	n/a
GA2	2251911	-	2259706
GA1	2252909	2262585	2269282
MA	2252937	2262477	2270361
GRA2	2255416	-	-
LR1	2258503	n/a	n/a
GRA1	2259340	-	-

In the last experiment, real-world data is used to compare the performances of four methods with results reported in literature. In this system, there are only eight units and eight hours. The HH is allowed to execute 1000 iterations per run with the same parameter settings as explained in subsection 6.1. According to the results in Table 5, HH and BDE2 methods obtain the best result. Table 6 shows the 95% confidence intervals for the cost values calculated by HH for the test systems. First row gives the means for each test system and the second row shows the 95% confidence intervals of the means.

Table 5. Results for the Turkish Interconnected Power System

Algorithm	Best	Average	Worst
HH	530346	530346	530346
BDE2	530346	530346	530346
ES	530392	530392	530392
SSGA	530392	530392	530392
BDE1	532142	-	-

Table 6. Mean and the 95% confidence interval of the cost calculated by HH

	System 1	System 2	System 3	Turkish Power System
Mean	566121	1127553	2251528	530346
Confidence interval	[565958.15 , 566283.85]	[1127151.55 , 1127954.45]	[2251014.8 , 2252041.2]	[530346 , 530346]

7 Conclusion

In this study, a robust and scalable solution is proposed for the UCP through the application of a hyper-heuristic approach. Four test sets are used to compare the performance of HH with other optimization methods. Firstly, HH is applied to three test sets, which are taken from literature, and the change in the performance of the algorithm is investigated with respect to other methods. In the last test, real-world data is used to compare the efficiency of the algorithms.

As can be seen from the results, HH shows very good performances in these four test sets. For System 2, System 3 and the Turkish Interconnected Power Network System, it obtains the overall best result; in System 1 it finds the second best result, but the difference between the overall best result and the result of HH is very small. Unlike some methods, such as LR1, GA1, GRA1, GRA2, which obtain inconsistent results with varying data sizes, HH succeeds in getting good and consistent results in all four data sets. The use of hill-climbers has an important role on this performance.

Additionally, the performance of HH and GA2 are compared. Despite using the same genetic operators, HH becomes more successful in larger data sets. Moreover, HH does not require parameter tuning or adaptation before and during the search. However, in GA2, the search needs to be monitored to prevent convergence and to find good results. Therefore, HH is a robust solution approach for large data sets.

As the first application of HH to the UCP, the results are very impressive. The performance of this approach can be further enhanced by applying more sophisticated heuristic operators and hill-climbers. Besides, as future work, the initial solution can be improved by using a heuristic such as the priority list method [6]. Advanced techniques with learning mechanisms, which make decisions based on feedback from previous iterations, can also be used for heuristic selection and move acceptance.

References

1. Cheng, C., Liu, C.W., Liu, C.C.: Unit Commitment by Lagrangian Relaxation and Genetic Algorithms. IEEE Transactions on Power Systems 15(2), 707–714 (2000)
2. Lowery, P.G.: Generating Unit Commitment by Dynamic Programming. IEEE Tranasactions on Power Apparatus and Systems PAS-85(5), 422–426 (1966)
3. Zhuang, F., Galiana, F.D.: Unit Commitment by Simulated Annealing. IEEE Transactions on Power Systems 5(1), 311–318 (1990)
4. Mantawy, A.H., Abdel-Magid, Y.L., Selim, S.Z.: Unit Commitment by Tabu Search. IEEE Proceedings – Generation, Transmission and Distribution 145(1), 56–64 (1998)
5. Chen, C.L., Wang, S.C.: Branch-and-Bound Scheduling for Thermal Generating Units. IEEE Transactions on Energy Conversion 8(2), 184–189 (1993)

6. Burns, R.M., Gibson, C.A.: Optimization of Priority Lists for a Unit Commitment Program. In: Proceedings of IEEE/PES Summer Meeting, pp. 1873–1879 (1975)
7. Viana, A., de Sousa, J.P., Matos, M.: Using Grasp to Solve the Unit Commitment Problem. Annals of Operations Research 120, 117–132 (2003)
8. Kazarlis, S.A., Bakirtzis, A.G., Petridis, V.: A Genetic Algorithm Solution to the Unit Commitment Problem. IEEE Transactions on Power Systems 11(1), 83–92 (1996)
9. Rudolf, A., Bayrleithner, R.: A Genetic Algorithm for Solving the Unit Commitment Problem of a Hydro-Thermal Power System. IEEE Transactions on Power Systems 14(4), 1460–1468 (1999)
10. Valenzula, J., Smith, A.E.: A Seeded Memetic Algorithm for Large Unit Commitment Problems. Journal of Heuristics 8, 173–195 (2002)
11. Dudek, G.: Unit Commitment by Genetic Algorithm with Specialized Search Operators. Electric Power Systems Research 72(3), 299–308 (2004)
12. Keles, A., Etaner-Uyar, A.S., Turkay, B.: A Differential Evolution Approach for the Unit Commitment Problem. In: Proceedings of ELECO 2007: 5th International Conference on Electrical and Electronics Engineering, pp. 132–136 (2007)
13. Ozcan, E., Bilgin, B., Korkmaz, E.E.: A Comprehensive Analysis of Hyper-heuristics. Intelligent Data Analysis 12(1), 3–23 (2008)
14. Saramourtsis, J., Damousis, A., Bakirtzis, A.G., Dokopoulos, P.: Genetic Algorithm Solution to the Economic Dispatch Problem. IEE Proceedings-C 141(4), 377–382 (1996)
15. Duo, H., Sasaki, H., Nagata, T., Fujita, H.: A Solution for Unit Commitment Using Lagrangian Relaxation Combined with Evolutionary Programming. Electric Power Systems Research 51, 71–77 (1999)
16. Damousis, I.G., Bakirtzis, A.G., Dokopoulos, P.S.: A Solution for Unit Commitment Problem Using Integer-Coded Genetic Algorithm. IEEE Transactions on Power Systems 19(2), 1165–1172 (2004)
17. Dang, C., Li, M.: A Floating-Point Genetic Algorithm for Solving the Unit Commitment Problem. European Journal of Operational Research 181, 1370–1395 (2007)
18. Uyar, Ş.A., Türkay, B.: Evolutionary Algorithms for the Unit Commitment Problem. Turkish Journal of Electrical Engineering 16(3), 239–255 (2008)
19. Bilgin, B., Ozcan, E., Korkmaz, E.E.: An Experimental Study on Hyper-Heuristics and Final Exam Scheduling. In: Proceedings of the International Conference on the Practice and Theory of Automated Timetabling, pp. 123–140 (2006)
20. Davis, L.: Bit Climbing Representational Bias and Test Suite Design. In: Proceedings of the 4th International Conference on Genetic Algorithms, pp. 18–23 (1991)

Application of Genetic Programming Classification in an Industrial Process Resulting in Greenhouse Gas Emission Reductions

Marco Lotz[1] and Sara Silva[2,3]

[1] Tropical Research Institute, Portugal
[2] INESC-ID Lisboa, Portugal
[3] CISUC, University of Coimbra, Portugal
sara@kdbio.inesc-id.pt

Abstract. This paper compares Genetic Programming and the Classification and Regression Trees algorithm as data driven modelling techniques on a case study in the ferrous metals and steel industry in South Africa. These industries are responsible for vast amounts of greenhouse gas production, and greenhouse gas emission reduction incentives exist that can fund these abatement technologies. Genetic Programming is used to derive pure classification rule sets, and to derive a regression model used for classification, and both these results are compared to the results obtained by decision trees, regarding accuracy and human interpretability. Considering the overall simplicity of the rule set obtained by Genetic Programming, and the fact that its accuracy was not surpassed by any of the other methods, we consider it to be the best approach, and highlight the advantages of using a rule based classification system. We conclude that Genetic Programming can potentially be used as a process model that reduces greenhouse gas production.

1 Introduction

According to Lotz et al [1] it is commonly accepted that anthropogenic greenhouse gas (GHG) emissions are at least partially responsible for current climate change problems. The Intergovernmental Panel on Climate Change is also a leading authority on the subject [2]. In 1997 the Kyoto Protocol was adopted at the Third Session of the Conference of the Parties to the United Nations Framework Convention on Climate Change [3]. The purpose of this protocol was to force Annex-I countries, or industrialized countries, to accept legally binding commitments to reduce GHG emissions. The targeted GHGs are CO2, CH4, N2O, HFCs, PFCs, and SF6 [3]. For further information see aspects of Lotz and Brent [4]. Developing countries, like South Africa, have no binding GHG emission production cap in place, but GHG emission reduction projects in these countries could obtain significant sources of revenue from polluting less. The United Nations' emission reduction scheme aimed at developing nations is known as the Clean Development Mechanism (CDM) [3]. The trading of the Certified Emission Reduction (CER) units is facilitated by the Carbon Finance Unit of the World Bank [5].

C. Di Chio et al. (Eds.): EvoApplications 2010, Part II, LNCS 6025, pp. 131–140, 2010.

According to Zhuwakinyu [9] South Africa is Africa's largest steel producer, accounting for 48% of total crude steel production in 2008. Globally, South Africa is the 21st biggest producer. ArcelorMittal South Africa is a major production company in South Africa and in 2008 the worldwide production of ArcelorMittal was ranked as the top steel-producing company, accounting for about 103-million tons of global steel production. The country also has one primary stainless steel producer, Columbus Stainless. The primary steel industry is a significant contributor to the South African economy and earns considerable amounts of foreign exchange [9]. The ferrous metals and steel industry involves very energy intensive production processing. This energy is sourced from fossil fuel, like coal and gas, and from electricity. The national electricity grid in South Africa generates ± 0.96 - 1.2 tCO2e/MWh [10]. The use of energy in South African ferrous metals and steel industry thus leads to significant amounts of GHG production. As example, in the calendar year 2008 ArcelorMittal (South Africa Ltd) reported 12,420,730 tCO2e [11], Scaw Metals 1,727,590 tCO2e [12] and Highveld Steel & Vanadium declined to disclose their GHG emissions [11].

Genetic Programming (GP) is the automated learning of computer programs, using Darwinian selection and Mendelian genetics as sources of inspiration [6]. Despite its obvious potential in process systems engineering, GP does not appear to have gained large-scale acceptance in process engineering applications, at least when compared to Artificial Neural Networks and Support Vector Machines. One of the problems usually associated to GP is the so called bloat, an excess of code growth without a corresponding improvement in fitness [7]. Bloat is one of the reasons why the solutions derived by GP are usually complex and hard to understand, and thus of difficult acceptance among practitioners. On the other hand, Classification and Regression Trees (CARTs) are a non-parametric, non-linear rule based classifier that generates classification rules through an induction procedure described in [14]. They are based on a hierarchical decision scheme where the feature space is subject to a binary recursive partitioning that successively splits the data. Unlike GP, the CART algorithm is very popular and successful.

The aim of this research is to compare GP and CART as a data driven modelling technique on a case study in the ferrous metals and steel industry. As discussed above, these industries are responsible for vast amounts of GHG production, and GHG emission reduction incentives exist that can fund these abatement technologies. GP is used to derive pure classification rule sets, and to derive a regression model used for classification. These results are compared to each other, and to the CART results, regarding accuracy and human interpretability/readability. We do not intend to draw conclusions regarding the general appropriateness of these or other techniques in similar problems, but instead use this particular application to exemplify how GP can provide a competitive solution. If GP can produce a model that outperforms a traditional approach like CART, then it can potentially be used as a process model that reduces GHG production.

In Section 2 the data and modelling techniques are discussed, while Section 3 focuses on the experiments/case study and results obtained. Section 4 is a discussion on the results and the paper ends with Section 5 summarizing the conclusions drawn during this research.

2 Data and Methods

The data set of this case study originated from an industrial process. In this process steel was hot-rolled into steel plates. This data consists of 20 inputs represented by X1 to X20, regarding different measurements related to the production. The output or target value is a binary variable. A value of 1 indicates that an error occurred during the production and the result was a defective steel plate, while a value of 0 indicates successful production. The aim is to predict whether the production was successful or not. The input variables X1–X20 are summarized in Table 1. The data set contains 3,015 samples, which were randomized and split into a training set of 2615 entries and a test set of 400 entries, across which examples of both classes were spread equally. The classes are not equally represented, as only 27.4% of the samples represent defective steel plates.

Table 1. Input variables, measurement and units

Input variable	Measurement	Units
X1	Phosphorus content	%
X2	Grinding loss	%
X3	Reheating retention time	h
X4	Width of steel plate	mm
X5	Speed at which plate moves	m/min
X6	Mould level difference	m
X7	Stopper movement	m
X8	Tundish steel mass	ton
X9	Superheated steel temperature	°C
X10	Silicon content	%
X11	Rinse end temperature	°C
X12	Contact time	h
X13	Add to gas end	h
X14	Rinse station stir parameter	Nm^3 / min / °C
X15	Ti3O5 content	kg
X16	TiN content	kg
X17	Product thickness	mm
X18	Mould temperature measurement at specific point	°C
X19	Mould temperature measurement at specific point	°C
X20	Mould temperature measurement at specific point	°C

The modelling methods employed were:

Classification and Regression Trees (CART): This is the benchmark method against which the two GP methods are compared. The chosen input variables all act as binary splits as compared to a statistically determined value and all terminal nodes are class associated integers. See [14] for more information on CART systems;

GP evolved pure classification rules (Class-GP): These rules directly predict a class associated integer, requiring no rounding of outputs to integers. The function set used contained the following functions: 'if-then-else' (myif), 'greater than' (gt), 'less or equal than' (le), and 'or'. The terminal set was limited to '0', '1', and X1–X20. GPLAB [13] was used to derive the GP classification rules. (It is important to note that GPLAB could also have been used to derive the GP evolved regression function);

GP evolved regression function (Reg-GP): This is a single complex function which produces a value that must be rounded to the nearest class associated integer. The GP application used was a commercially available package called Discipulus [15]. This GP application directly executes mathematical operators on the data set;

No data filtering was done to compensate with difference in order of magnitude of input variables, investigate interdependencies of variables, statistical outliers or any other phenomena. It is expected that CART and both GP approaches are able to deal with such relationships without being assisted. The error of the derived models is calculated as the number of prediction errors that are made on the unseen test data set.

3 Experiments and Results

3.1 The CART Model

The model produced by CART consists of a decision tree containing 525 nodes and 36 terminal nodes. This means the data set is classified using 36 rules. Too large to be represented graphically, this tree was however pruned with success. Pruning is the process of reducing a tree by turning some branch nodes into terminal nodes and removing all other nodes of the original branch [16]. The tree was pruned to 17 nodes including 9 terminal nodes, denoting 9 classification rules. CART is not stochastic and, as such, always produces the same result. The pruned tree is illustrated in Figure 1.

The CART model must be interpreted from the top down to get to the 9 terminal nodes denoting the 9 distinct classification rules. Each non-terminal node is evaluated as:

IF (condition is TRUE)
THEN (go to the left branch)
ELSE (go to the right branch)

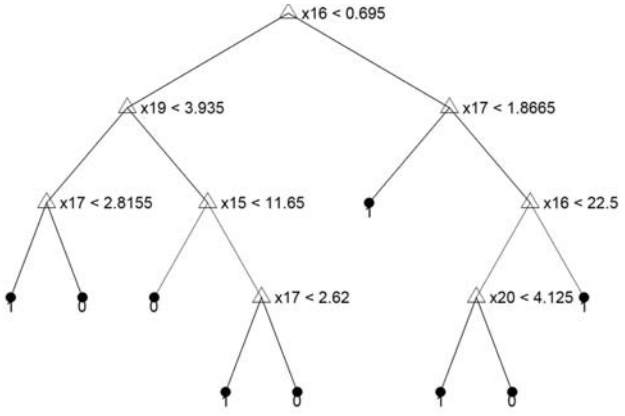


Fig. 1. Pruned CART decision tree

As example, the two first rules (leftmost terminals) are:

IF X16 $<$ 0.695
AND X19 $<$ 3.935
AND X17 $<$ 2.8155
THEN Class 1

IF X16 $<$ 0.695
AND X19 $<$ 3.935
AND X17 $\geq$ 2.8155
THEN Class 0

The original (unpruned) CART model misclassified 118 of the 400 unseen test cases. The pruned tree performed slightly better by misclassifying 108 entries. It is of interest to note that the pruned model only used 5 of the 20 available input variables.

3.2 GP Evolved Pure Classification Rules (Class-GP)

When using GP to evolve classification rules, the resulting model is, in its essence, similar to the CART model. It consists of "pure classification rules" since each rule has a single class associated integer as an output. No rounding of model outputs is necessary.

Since GP is stochastic, various runs (around 40) were executed on the same training and test data sets. Different population sizes were used (100–500 individuals), allowed to evolve for a varying number of generations (30–300). The populations were always initialized with the Full method [17] with an initial maximum depth of 6, and no maximum limits for the allowed depth during the run. Crossover and mutation were used with equal probabilities, the reproduction rate was 0.1, and no elitism was used. Apart from the usage of the Lexicographic Parsimony Pressure [18] tournament, no other bloat control measures were taken.

The absolute best model found misclassified 105 of the unseen test samples. This model had a tree depth of 14 levels and consisted of 40 nodes, and was

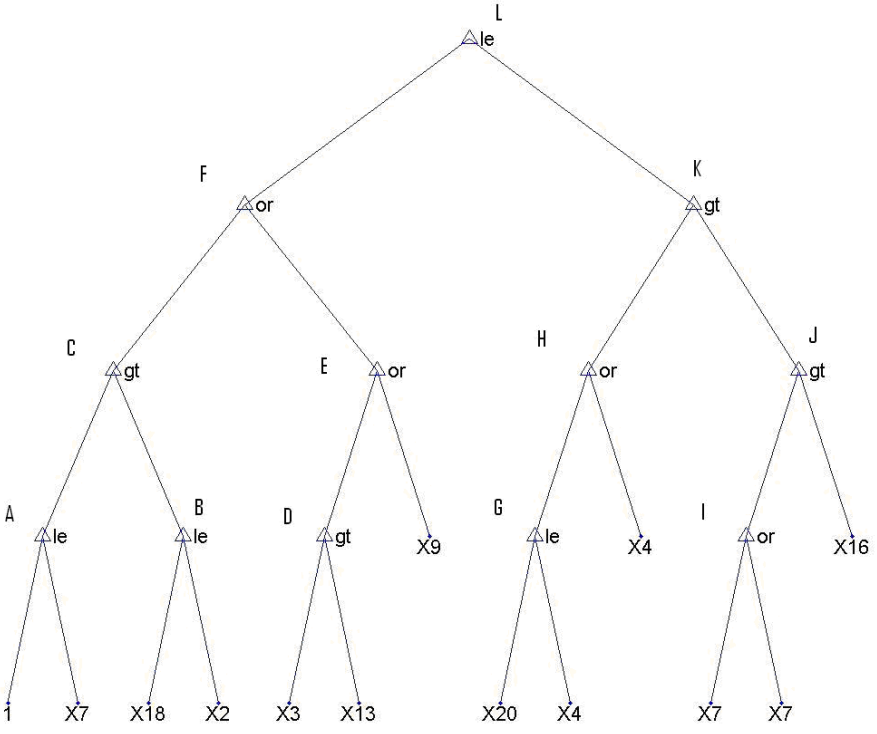


Fig. 2. Class-GP tree before simplification. Nodes are arbitrarily labelled.

obtained using 300 individuals evolved for 200 generations. Another Class-GP model was evolved, using exactly the same settings as the previous, that has similar accuracy and much less complexity. This model misclassified 109 test samples and consists of 25 nodes in a tree of 5 levels. Figure 2 represents this tree exactly as returned by GPLAB, before any simplification.

Generally the easiest way in which to interpret the GP trees is to start at a terminal node and "work your way up". As examples, nodes A, B and C in Figure 2 are interpreted as:

Node A:	Node B:	Node C:
IF $1 \leq$ X7	IF X18 $\leq$ X2	IF Node A $>$ Node B
THEN 1	THEN 1	THEN 1
ELSE 0	ELSE 0	ELSE 0

This interpretation method is then repeated for all tree branches up to the top node, L.

Given that the input variables are all positive, which in the boolean terms of the functions 'gt', 'le' and 'or' (function set in Section 2) always evaluates to 1 (true), most of the nodes on this tree were identified as being introns. As

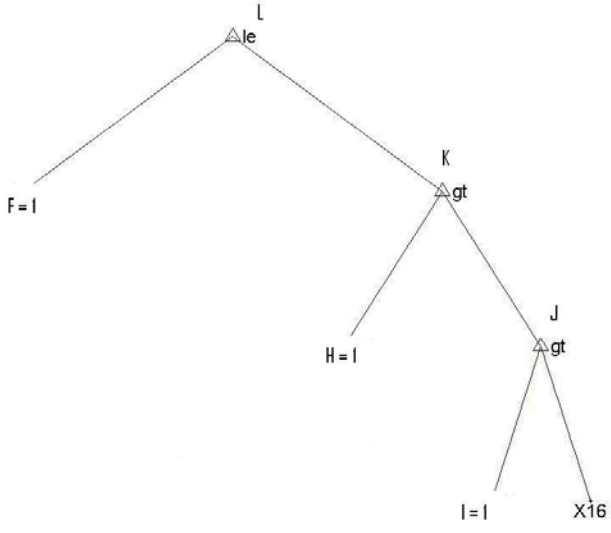


Fig. 3. Class-GP tree after simplification. Original tree shown in Figure 2.

example, evaluating all the nodes below node F on the training set indicated that node F always returns the value 1. Nodes A–E can then be removed and node F simply becomes '1'. The removal of all intron nodes resulted in the tree represented in Figure 3. This simplified model consists of 7 nodes, including 5 terminal nodes, that can be interpreted as the following set of rules:

Node J:
IF $1 > X16$
THEN 1
ELSE 0

Node K:
IF $1 >$ Node J
THEN 1
ELSE 0

Node L:
IF $1 \leq$ Node K
THEN 1
ELSE 0

Combining these nodes results in:

IF $1 \leq (1 > (1 > X16))$
THEN 1
ELSE 0

This can be simplified to the following simple rule:

IF $X16 \geq 1$
THEN 1
ELSE 0

It is clear that the prediction is only dependent on variable X16. This means that, remarkably, 291 out of 400 test samples are correctly classified by knowing

only the TiN content (see Table 1). Also note that this model is far less complex that the 9 rules generated by the pruned CART tree with basically the same accuracy.

3.3 GP Evolved Regression Function (Reg-GP)

To evolve the GP regression function a number of runs (around 10) consisting of 500 individuals evolved for 50,000 generations were executed. The outputs of the Reg-GP function must be rounded to the nearest integer assigned to a class, resulting in a binary classification model. The absolute best model found used 8 of the 20 available input variables, and misclassified 136 of the unseen test entries. The C/C++ model returned by this GP system does not lend itself to a tree representation comparable to CART or Class-GP. Apparently using no explicit bloat control, even the simplest model obtained with this system consists of a mathematical expression several pages long, considered too long to be included here.

4 Discussion

Table 2 is a summary of the results obtained by the three different approaches. CART and Class-GP have provided comparable models, with the same accuracy and interpretability of the classification rules. However, CART used more variables, produced a larger tree (even after pruned) and, as a consequence, requires more rules to perform the classification. Class-GP produced a tree that could be simplified to use only one variable, requiring only one very simple classification rule. On the other hand, Reg-GP produced models that are very difficult in terms of human interpretability. They consist on a complex mathematical function whose outputs must be rounded to produce a binary classification. Besides using more variables, the Reg-GP model also revealed lower accuracy.

Table 2. Summary of model results by CART (pruned), Class-GP and Reg-GP

Method	Variables used	Accuracy	Model format
CART (pruned)	X15,X16,X17,X19,X20	73%	9 classification rules
Class-GP	X16	73%	1 classification rule
Reg-GP	X1,X4,X5,X10,X15-X17,X20	66%	1 regression function

The Class-GP approach has the additional advantage of being able to compare inputs to one another (see Figure 2, node B) as opposed to always comparing an input to a statistically derived value (see Figure 1, all nodes) like CART does. The result is that the Class-GP approach provides more insight into the fundamentals of the system as compared to the CART approach.

Having a set of pure and meaningful classification rules is of great advantage for plant design and plant operators. Presenting a "black box" model to an industry to adopt leads to resistance due to a lack of understanding of the model. The largest fundamental drawback of using Reg-GP is the lack of insight gained by the user, at least in this particular application. With CART and Class-GP, the model is a set of rules that can be easily understood and interrogated individually, and that clearly indicate which input variables are required to perform the prediction.

It is interesting to note that the only variable used by Class-GP, X16, is also the only variable that was always used by all models. Clearly the importance of the TiN content (X16 in Table 1) in predicting production errors was identified by all the models. Controlling the TiN concentrate will then have a beneficial influence on the successful production of steel plates and the identification of plates with surface defects. The implication is that less steel plates will have to be scrapped and put through the energy intensive production process. Having more steel plates produced, by scrapping less plates, results in having a lower energy requirement per plate produced. This in turn results in less GHG production as energy and GHG production are closely correlated in South Africa.

5 Conclusions

We have compared GP and CART as a data driven modelling technique on a case study in the ferrous metals and steel industry in South Africa, with the goal of predicting production errors. GP was used to derive pure classification rule sets (Class-GP), and to derive a regression model used for classification (Reg-GP). These results were compared to each other, and to results obtained by the CART algorithm, regarding accuracy and human interpretability/readability. Considering the overall simplicity of the rule set obtained by Class-GP, and the fact that its accuracy was not surpassed by any other method, we consider it to be the best approach, and highlight the advantages of using a rule based classification system.

Successful production of steel plates results in lower energy requirements. This in turn results in less GHG production, as energy and GHG production are closely correlated in South Africa. The GHG emissions achieved are real and measurable. The result is that the GHG emissions reductions could be a major source of income if such a project is registered under the UN's Clean Development Mechanism (CDM) or other emission reduction incentive schemes. Not only did GP produce a real world process model, it can actually be used in curbing the production of GHGs.

Acknowledgements

The authors would like to thank Professor Chris Aldrich (University of Stellenbosch, South Africa, Department of Process Engineering) for the work done on the

case study as presented in [8]. They also acknowledge project "EnviGP – Improving Genetic Programming for the Environment and Other Applications" (PTDC/EIA-CCO/103363/2008) from Fundação para a Ciência e a Tecnologia, Portugal.

References

1. Lotz, M., Brent, A., Steyn, H.: Addressing the need for a Clean Development Mechanism (CDM) specific project management strategy. South African Journal of Environmental Management Science NS 12(2), 228–241 (2009)
2. Intergovernmental Panel on Climate Change (2009), http://www.ipcc.ch/
3. United Nation Framework Convention on Climate Change (UNFCCC), http://cdm.unfccc.int
4. Lotz, M., Brent, A.: Towards a comprehensive Clean Development Mechanism (CDM) approach for biodiesel. Energy and Sustainability II, 279–290 (2009)
5. Capoor, K., Ambrosi, P.: State and trends of the carbon market. World Bank Institute, http://carbonfinance.org/
6. Poli, R., Langdon, W.B., McPhee, N.F.: A field guide to genetic programming. Published via http://lulu.com and freely available at http://www.gp-field-guide.org.uk (2008) (With contributions by J. R. Koza)
7. Silva, S., Costa, E.: Dynamic limits for bloat control in genetic programming and a review of past and current bloat theories. Genet. Program. Evolvable Mach. 10(2), 141–179 (2009)
8. Lotz, M.: Modelling of process systems with genetic programming. Masters Dissertation, Stellenbosch University (2006), http://etd.sun.ac.za/jspui/handle/10019/570 (downloaded 2008-12-03)
9. Zhuwakinyu, M.: The Research Unit of Creamer Media (Pty) Ltd., South Africa's Steel Industry (October 2009), http://www.researchchannel.co.za (2009)
10. Institute for Global Environmental Studies: EnviroScope (2009), http://enviroscope.iges.or.jp/modules/envirolib/view.php?docid=2136
11. Carbon Disclosure Project (2009), https://www.cdproject.net/en-US/Results/Pages/overview.aspx
12. Scaw Metals Group: Sustainable Development Report 2008 (2008), http://www.scaw.co.za/
13. Silva, S.: A Genetic Programming Toolbox for MATLAB, http://gplab.sourceforge.net/ (2004)
14. Breiman, L., Friedman, J.H., Olshen, R.A., Stone, C.J.: Classification and regression trees. Wadsworth, Belmont (1984)
15. Discipulus, http://www.rmltech.com/
16. MATLAB, http://www.mathworks.com/access/helpdesk/help/toolbox/stats/treeprune.html
17. Koza, J.R.: Genetic programming – on the programming of computers by means of natural selection. MIT Press, Cambridge (1992)
18. Luke, S., Panait, L.: Lexicographic parsimony pressure. In: Langdon, W.B., et al. (eds.) Proceedings of GECCO 2002, pp. 829–836. Morgan Kaufmann, San Francisco (2002)

Influence of Topology and Payload on CO_2 Optimised Vehicle Routing

Cathy Scott, Neil Urquhart, and Emma Hart

School of Computing, Edinburgh Napier University, Edinburgh, UK
c.scott@napier.ac.uk

Abstract. This paper investigates the influence of gradient and payload correction factors used within a CO_2 emission model on the solutions to shortest path and travelling salesman problems when applied to freight delivery.

Problem instances based on real life examples using the road network of Scotland are studied. Solutions are obtained using a range of metrics and vehicles. The results are compared to determine if the inclusion of gradient and payload as inputs to the emission model have any influence on the final routes taken by vehicles or the order of visiting customers. For the problem instances studied no significant influence was found. However for vehicle routing problems with large differences in payload and hilly road networks further investigation is needed.

1 Introduction

The routing and scheduling of vehicles is a well researched topic and one that has many practical applications. The efficient routing of freight delivery, waste collection, courier services and road gritting are examples.

Typically the objective function to be optimised is related to the number of vehicles used, the distance travelled or to an overall cost. Optimising routing and scheduling problems with respect to CO_2 emissions has, until recently, received little attention.

The Intergovernmental Panel on Climate Change (IPCC) concludes that the main source of increased CO_2 levels in the atmosphere are a result of the burning of fossil fuels [2]. A number of international treaties to control greenhouse gases have been agreed. These incude the United Nations Framework Convention on Climate Change [3] and the Kyoto Protocol to the Framework Convention on Climate Change [4]. The Climate Change Act 2008 legally committed the UK government to reduce CO_2 emissions by at least 26% from the 1990 levels by 2020 [5]. Road freight transport accounted for 6% of all UK CO_2 emissions in 2004 [6].

In this paper we explore some of the factors that influence road vehicle optimisation problems, particularly with respect to the minimisation of CO_2 emissions from the transport of goods. In particular the effect of road gradient and vehicle payload are examined as these factors have been shown to have a considerable impact on the level of fuel consumed and thus CO_2 emitted.

C. Di Chio et al. (Eds.): EvoApplications 2010, Part II, LNCS 6025, pp. 141–150, 2010.

2 Related Work

The traveling salesman problem and the Vehicle Routing Problem (VRP) have been well studied. A survey of methods for solving the Traveling Salesman Problem can be found in Lawler et al. [10]. A review of the application of evolutionary algorithms to the TSP can be found in [12].

However only a few studies have been done with regard to CO_2 emissions. Sbihi and Eglese [13] reviewed the link between vehicle routing problems and green logistics. Palmer [14] integrated an emissions model with a genetic algorithm to minimise CO_2 emissions for supermarket home deliveries. This study utilised a light goods vehicle for which the fuel consumption is not influenced by payload and road gradient.

The effect of payload and gradient on optimum route choice has been studied by [15] [16] on the mountainous Cape Verde isles in the context of a waste collection service. These effects are most noticeable for long distance transportation when the vehicle is fully laden with waste. Fuel savings of 52% were achieved when optimising the route for low fuel consumption rather than distance.

Ericsson, Larsson and Brundell-Freij [17] developed a routing tool for drivers that minimised fuel consumption.

In [18] Jabali et al. optimised a time dependent vehicle routing problem for both total travel time and total CO_2 emissions. A simple average speed CO_2 model was used and payload and gradient were ignored.

3 CO_2 Emission Modelling

This study employed the COPERT [9] model which is based on the average speed of a vehicle and includes payload and correction factors for heavy goods vehicles (HGVs). Payload and gradient do not have a significant impact on the fuel consumption and emissions from light goods vehicles. Although mainly used to calculate the annual emissions of a country's vehicle fleet, the methodology used within COPERT has been shown to be sufficiently accurate to calculate emissions with a temporal resolution of 1 hour and a spatial resolution of 1 km. The use made of this model within this study falls outside these bounds. Other models exist [8] that would permit a much smaller spatial and temporal resolution. However these models require more data inputs and involve more complex processing than the model selected. In addition these models were not available to the authors at the time of writing.

The COPERT model models both cold start and hot engine emissions. In this study only hot engine emissions were considered.

For this study several vehicle classifications were used. These are a light goods vehicle (LGV) and 7.5 tonne, 18 tonne and 26 tonne rigid-bodied heavy goods vehicles (HGVs). All were assumed to have engines conforming to the most recent engine standards, that is Euro 6 for the LGV and EuroVI for HGVs.

The output from the COPERT model is a baseline emission factor giving the grams of CO_2 emitted per kilometre. The emission factor is a function of

the vehicle classification, the European emission standard of the engine and the mean speed of the vehicle. CO_2 emissions are directly proportional to fuel consumption.

For HGVs a gradient correction factor is then applied to this emission factor to correct for uphill and downhill slopes. The gradient correction factor is a function of the vehicle mass, the mean vehicle speed and the road gradient. The baseline emission factor is corrected as follows:

$$\eta_i = G_i e_i \ . \tag{1}$$

where

η_i is the gradient corrected emission factor in g CO_2/km of vehicle i, G_i is the gradient correction factor of vehicle i and e_i is the baseline emission factor of vehicle i.

The above emission factors assume that the vehicle is 50% loaded. To compensate for different vehicle loadings the emission factor for HGVs can be corrected as follows:

$$\beta_i = \eta_i[1 + 2\lambda_i(L - 50)/100] \ . \tag{2}$$

where

β_i is the load corrected emission factor in g CO_2/km of vehicle i, η_i is the gradient corrected emission factor of vehicle i, λ_i is the load correction factor of vehicle i and L is the actual loading of vehicle as a percentage of the maximum load.

The final emission factor is then used to calculate the cost of traversing a given road link as follows:

$$cost_{i,j} = \beta_{i,j} l_j \ . \tag{3}$$

where

$cost_{i,j}$ is the cost in g CO_2 of vehicle i traversing roadlink j, $\beta_{i,j}$ is the load corrected emission factor of vehicle i on roadlink j and l_j is the length of roadlink j in km.

4 Experimental Approach

The study is based on two very different sets of problem data. The first concerns the delivery of groceries to households within the City of Edinburgh from a supermarket store. The second concerns the delivery of paper from a warehouse in North Lanarkshire to commercial customers throughout Scotland. In the first instance a diesel-fuelled light goods vehicle (LGV) was studied. In the second instance a variety of rigid-bodied heavy goods vehicles (HGVs) and LGVs are employed.

The mapping data employed for this study is from the UK Ordnance Survey's (OS) Integrated Transport Network (ITN) layer with the Ordnance Survey's Land-Form PROFILE providing height data. This mapping data was loaded into a MySQL database. The map for Scotland includes approximately 500,000

nodes and 600,000 edges. The study only included deliveries within mainland Scotland and no use of ferries or railways was made.

There is no information on vehicle speeds in the OS ITN data. While the ITN data does contain basic information about road categories, the assignment of an appropriate vehicle speed is far from simple. For example a road category of 'A Road' can represent both a rural road with a legal speed limit of 60mph or a city centre street with a legal limit of 30mph.

Land-Use and Transport Integration in Scotland (LATIS)[20] models road transport on the strategic road network in Scotland. LATIS takes into account land-use planning and travel demand and can be used to predict the congested average speed of traffic on all roads in the model, as well as the free-flow speed. Thus the LATIS model can supplement the OS data providing vehicle speed data for a subset of the roads. For roads outside the LATIS model only estimates based on the limited OS road categories can be used. The estimates for this study are based on measurements made by the Department of Transport [19] and are shown in table 1.

The vehicle speeds employed in this paper are an initial estimate only. They do not consider the effect of congestion, the varying of congestion with time or driver choice of speed. These will be studied in future work.

Table 1. Default average speeds for road categories not in the LATIS model

ITN Road category	speed km/h
Single carriageway	45
A Road	45
B Road	45
Minor road	45
Local street	45
Pedestrianised street	30
Alley	30

Several traveling salesman problem (TSP) instances were randomly chosen from the problem datasets. Each of these represented a typical route for that vehicle type in an operational solution to the problem data. As the loadings of the vehicles were often far less than the maximum, the delivery quantities to the customers was increased so that the vehicle typically left the depot 80-100% fully laden. This does not apply to LGVs where the payload has no effect on the fuel emission factor. The number of deliveries ranged from 5 to 13. Details are diplayed in table 2.

A bidirectional form of Dijkstra's routing algorithm [7] was used to find the shortest paths between all pairwise combinations of customers and the depot. The following objectives were used;-

- Least Distance.
- Least CO_2 emissions without the payload and gradient correction factors.

Table 2. Problem instances studied

Problem	Number of deliveries	Vehicle	%laden	Average distance (km)
1	10	LGV Diesel	n/a	9.6
2	5	7.5t rigid HGV	81	115.3
3	6	18t rigid HGV	80	132.8
4	13	26t rigid HGV	93	102.2

- Least CO_2 emissions with gradient correction factor.
- Least CO_2 emissions with the gradient and payload correction factors.

These objective functions are hereafter referred to with the following names;-

- Distance
- CO_2(basic)
- CO_2(gradient)
- CO_2(gradient/payload)

The COPERT model has gradient correction factors for slopes between -6% and 6%. Where an edge was found that had a slope outside of this range the nearest corresponding factor was employed.

The Distance and CO_2(basic)shortest paths are symmetrical, whereas taking the gradient into account makes the shortest paths asymmetrical. That is the cost of travelling from A to B is not necessarily the same as travelling from B to A. Likewise the optimal route may also be different. When calculating the shortest paths for the CO_2(gradient/payload) objective the process was repeated for vehicle loadings of 0, 50 and 100% to find out if the loading of a vehicle affected the optimal route between 2 points.

Although the shortest paths were constructed to minimise the above objective functions they were all costed and analysed using the full CO_2 model that is using both gradient and payload correction factors. For each of the 3 CO_2 objective functions the shortest paths found were compared with the corresponding paths found for the distance objective. The CO_2 savings for each path and the additional distances travelled to acheive that saving were then analysed.

The TSP instances were then solved for each of the objective functions using the appropriate shortest paths as inputs. Since it is the input factors to the emission model that are being studied, the choice of TSP algorithm is not the focus of this paper, therefore, the smaller instances were solved by iterating through all solution possibilities. The larger instances were solved using an evolutionary algorithm proposed by [11]. The best solution found in each case is reported.

When optimising using the CO_2(gradient/payload) objective function three paths between each pair of customers or customer/depot have been calculated. These paths are the optimal path for a 0, 50 or 100% laden vehicle and may or may not be different. The path chosen depends on the current loading of the vehicle which, in turn, depends on the customer's position in the overall TSP route. Vehicle with loading of 25% or less were assigned to the 0% laden path, those with loading of 75% or greater the 100% laden path and the rest the 50% laden path.

As before the calculated TSP solutions were also costed and analysed using the full CO_2 model with both gradient and correction factors. This calculation also took into account the current loading of the vehicle that was updated as the vehicle completed its tour of customers.

5 Evidence

The results of all the shortest path calculations are displayed in figures 1 and 2.

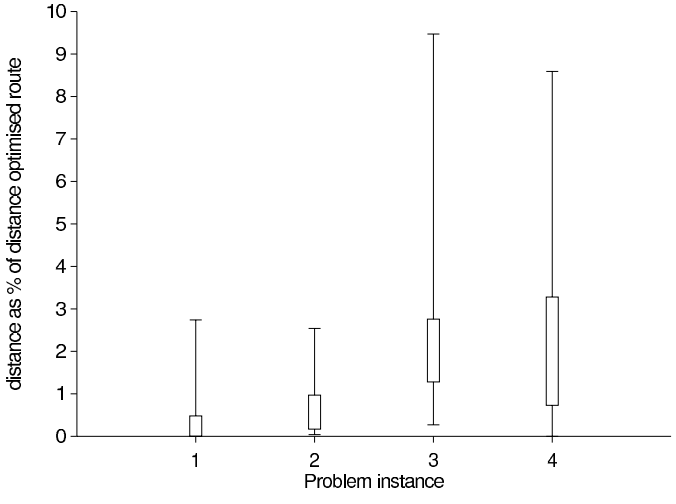


Fig. 1. Extra distance of CO_2 optimised paths for all pairwise combinations of customers and depot compared with distance optimised paths. The box represents the 2nd and 3rd quartiles and the whiskers give the minimum and maximum values.

These show that, for the problem instances chosen, CO_2 savings for any one path of up to 23% are possible. However the CO_2 savings are highly dependent on the problem instance. For example problem instance 1 is based on grocery home deliveries in Edinburgh and for many pairs of customers the distance and CO_2 optimised paths are identical. The average saving is 0.5%. Problem instance 2 is based on paper deliveries from Lanarkshire, Scotland to customers in Edinburgh, Glasgow, Fort William, Angus and West Lothian. CO_2 savings ranged from 3.21% to 23.77% with an average of 12.47%. This difference in CO_2 savings is to be expected as the distances involved are much longer and there is more potential for alternative paths.

T-Tests were undertaken to compare the CO_2 savings of the CO_2(basic) objective function and the CO_2(gradient) function against the distance objective. Similarly the CO_2 savings of fully-laden and empty vehicles using the CO_2(gradient/payload) objective function against the distance objectie were

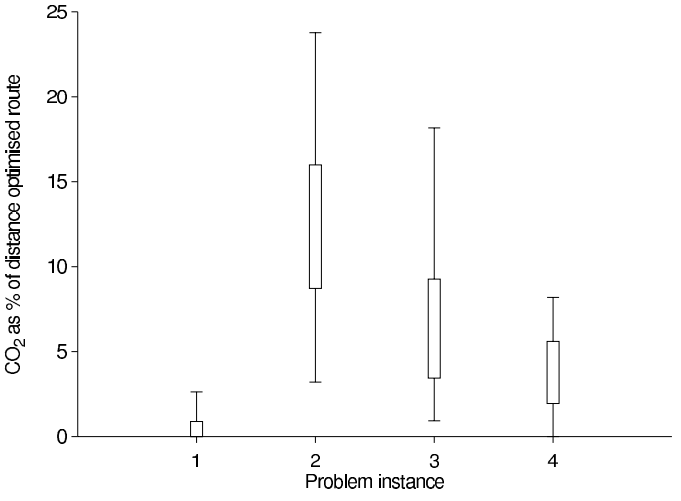


Fig. 2. CO_2 savings of CO_2 optimised paths for all pairwise combinations of customers and depot compared with distance optimised paths. The box represents the 2nd and 3rd quartiles and the whiskers give the minimum and maximum values.

compared. The t-test results are display in tables 3 and 4. These show that taking the gradient of a road into account or not when searching for a shortest path does not make a significant difference to the distance travelled or the CO_2 saved. Similarly shortest paths for empty and fully-laden vehicles are not significantly different.

Table 3. T-Test to compare the differences between CO_2 optimised paths and distance optimised paths with the differences between CO_2(gradient) optimised paths and distance optimised paths

Problem	Extra distance	CO2 saving
2	1.00	1.00
3	0.95	0.87
4	0.81	0.72

The results of the TSP solutions are compared in table 5. As for the shortest paths the level of CO_2 savings possible is dependent on the problem instance. The additional distance necessary to achieve reductions in CO_2 is not high. In addition the solutions were examined to check the order that customers are served. For problem instances 1, 2 and 3 there was no difference in the delivery order for any of the metrics. Thus the CO_2 savings come from different paths between customers rather than a different TSP solution.

For problem 4, however, the ordering of customers does vary. The CO_2(basic) optimised TSP solution visits customers in the same order as that of the distance

Table 4. T-Test to compare the differences between CO_2 optimised paths with 0% payload and distance optimised paths with the differences between CO_2 optimised paths with 100% payload and distance optimised paths

Problem	Extra distance	CO2 saving
2	0.99	1.00
3	0.98	0.99
4	1.00	1.00

Table 5. Extra distance travelled and CO2 savings achieved by CO2 optimised TSP solutions when compared with distance optimised solutions

Vehicle	Problem	Metric	Distance (km)	CO2 (g)	Extra distance	CO2 saving
LGV Diesel	1	Distance	70495	12225		
LGV Diesel	1	CO2	70748	12112	0.36%	-0.92%
18t HGV	2	Distance	557130	284806		
18t HGV	2	CO2	558259	246556	0.20%	-13.43%
18t HGV	2	CO2/gradient	558156	246553	0.18%	-13.43%
18t HGV	2	CO2/gradient/payload	558156	246553	0.18%	-13.43%
7.5t HGV	3	Distance	769328	209889		
7.5t HGV	3	CO2	787610	195818	2.38%	-6.70%
7.5t HGV	3	CO2/gradient	787600	195816	2.38%	-6.70%
7.5t HGV	3	CO2/gradient/payload	787621	195821	2.38%	-6.70%
26t HGV	4	Distance	835894	434228		
26t HGV	4	CO2	849199	426456	1.59%	-1.79%
26t HGV	4	CO2/gradient	850045	424761	1.69%	-2.18%
26t HGV	4	CO2/gradient/payload	852130	426099	1.94%	-1.87%

optimised solution. The CO_2(gradient) optimised solution shows a slight variation in order whilst the CO_2(gradient/payload) has a further customer order. However, although the solutions in terms of order of delivery are different, the overall difference in the CO_2 emission levels is very small.

6 Conclusions

The potential for CO_2 savings and the influence of gradient and vehicle payload on vehicle routing solutions are highly dependent on the problems studied. For the problem instances considered in this paper a wide range of potential CO_2 savings was found for both shortest path and traveling salesman problems. Even in problems where there are alternative paths between two customers with a large potential CO_2 saving this does not always lead to a difference in the final TSP solution in terms of a different order of visiting customers. In these cases the CO_2 savings arise from different paths between customers rather than a different order. Gradient and payload were found to effect the TSP order solution in one problem instance. However the difference in CO_2 emissions between the solutions

is less than the 2.1% coefficient of variation of the COPERT model for CO_2 as reported in [21].

In this study considerable computing effort and time was spent in calculating the paths between customers, especially when taking the vehicle payload into account. This was due to the size of the mapping data rather than the size of the TSP problem. Further work is needed to examine the algorithm used to calculate the inputs for a CO_2 optimised TSP to reduce the pre-processing effort needed.

In this study the effect of time windows, congestion and the opportunity for a driver to select an optimal vehicle speed were ignored. Future work will examine the effect of these on potential CO_2 emissions.

Acknowledgements

The authors would like to thank Tim Pigden of Optrak Distribution Software Limited, Chris Newcombe of Paperlinx Services (Europe) Limited, Kevin Lumsden of MVA Consultancy, Stephen Cragg of Transport Scotland and Andrew Palmer of Cranfield University for providing advice and data for use in this study.

References

1. 2008 Guidelines to Defra's GHG Conversion Factors: Department for Environment, Food and Rural Affairs (2008), http://www.defra.gov.uk/environment/business/reporting/pdf/passenger-transport.pdf (cited January 13, 2010)
2. Fourth Assessment Report: Climate Change 2007: Working Group I Report: The Physical Science Basis. Geneva: IPCC (2007), http://www.ipcc.ch/ipccreports/ar4-wg1.htm (cited January 13, 2010)
3. United Nations Framework Convention on Climate Change. United Nations (1992), http://unfccc.int/resource/docs/convkp/conveng.pdf (cited January 13, 2010)
4. Kyoto Protocol to the United Nations Framework Convention on Climate Change. United Nations (1998), http://unfccc.int/resource/docs/convkp/kpeng.pdf (cited January 13, 2010)
5. Climate Change Act 2008 Office of Public Sector Information (2008), http://www.opsi.gov.uk/acts/acts2008/pdf/ukpga_20080027_en.pdf (cited January 13, 2010)
6. McKinnon, A.: CO_2 emissions from freight transport in the UK. Commission for Integrated Transport (2006)
7. Dijkstra, E.W.: A note on two problems in connexion with graphs. Numerische Mathematik 1, 269–271 (1959)
8. Barlow, T.J., Boulter, P.G., Mccrae, I.S.: Scoping study on the potential for instantaneous emission modelling: summary report. Transport Research Laboratory (2007)
9. Ntziachristos, L., Samaras, Z.: COPERT III Computer programme to calculate emissions from road transport. Methodology and emission factors (Version 2.1). Technical Report No 49. European Environment Agency, Copenhagen (2000), http://lat.eng.auth.gr/copert (cited January 13, 2010)

10. Lawler, E.L., Lenstra, J.K., Rinnoy Kan, A.H.G., Shmoys, D.B.: The Traveling Salesman Problem. Wiley and Sons, Chichester (1985)
11. Urquhart, N., Scott, C., Hart, E.: Using an evolutionary algorithm to discover low CO_2 tours within a Travelling Salesman Problem. In: Di Chio, C., et al. (eds.) EvoApplications 2010, Part II. LNCS, vol. 6025, pp. 421–430. Springer, Heidelberg (2010)
12. Larranaga, P., Kuijpers, C.M.H., Murga, R.H., Inza, I., Dizdarevic, S.: Genetic Algorithms for the Travelling Salesman Problem: A Review of Representations and Operators. Artificial Intelligence Review 13, 129–170 (1999)
13. Sbihi, A., Eglese, R.W.: Combinatorial optimization and Green Logistics. 4OR: A Quarterly Journal of Operations Research 5, 99–116 (2007)
14. Palmer, A.: The Development of an Integrated Routing and Carbon Dioxide Emissions Model for Goods Vehicles, Cranfield (2007)
15. Tavares, G., Zsigraiova, Z., Semiao, V., Carvalho, M.G.: Optimisation of MSW collection routes for minimum fuel consumption using 3D GIS modelling. Waste Management 29(3), 1176–1185 (2009)
16. Tavares, G., Zsigraiova, Z., Semiao, V., Carvalho, M.d.G.: A case study of fuel savings through optimisation of MSW transportation routes. Management of Environmental Quality: An International Journal 19(4), 444–454 (2008)
17. Ericsson, E., Larsson, H., Brundell-Freij, K.: Optimizing route choice for lowest fuel consumption - Potential effects of a new driver support tool. Transportation Research Part C: Emerging Technologies 14(6), 369–383 (2006)
18. Jabali, O., Woensel, T.V., Kok, A.G.D.: Analysis of Travel Times and CO Emissions in Time-Dependent Vehicle Routing (2009)
19. Road Statistics 2008: Traffic, Speeds and Congestion. Department for Transport (2008), http://www.dft.gov.uk/pgr/statistics/datatablespublications/roadstraffic/speedscongestion/roadstatstsc/roadstats08tsc (cited January 13, 2010)
20. Land-Use and Transport Integration in Scotland (LATIS). Transport Scotland (2010), http://www.latis.org.uk/ (cited January 13, 2010)
21. Barlow, T.J., Boulter, P.G.: Emission factors 2009: Report 2 - a review of the average-speed approach for estimating hot exhaust emissions. Transport Research Laboratory (2009)

Start-Up Optimisation of a Combined Cycle Power Plant with Multiobjective Evolutionary Algorithms

Ilaria Bertini[1], Matteo De Felice[1,2], Fabio Moretti[2], and Stefano Pizzuti[1]

[1] ENEA (Italian Energy New Technology and Environment Agency)
{ilaria.bertini,matteo.defelice,stefano.pizzuti}@enea.it
[2] Università degli Studi "Roma Tre", Dipartimento di Informatica e Automazione
moretti@dia.uniroma3.it

Abstract. In this paper we present a study of the application of Evolutionary Computation methods to the optimisation of the start-up of a combined cycle power plant. We propose a multiobjective approach considering different objectives for the optimisation in order to reduce the pollution emissions and to maximise the efficiency of the plant. We compare a multiobjective evolutionary algorithm (NSGA-II) with 2 and 5 objectives on a software simulator and then we use different metrics to measure the performances. We show that NSGA-II algorithm is able to provide a set of solutions, defined as Pareto Front, that represent the best trade-off on the different objectives among those the decision maker can choose.

1 Introduction

Air pollution emission reduction is nowadays a common requirement for the operation of industrial plants, from the Kyoto Protocol the attention about this issue is growing and therefore a more efficient utilisation of the plants is needed.

Moreover, with the liberalization of the energy market and the introduction of distributed energy power plants in the territory, it is required also for the gas-steam combined plants a greater flexibility in order to vary the provided power according to the needs or to implement variable running strategies. Such management makes more critical the problem of the identification of the best parameters during the start-up in order to reduce the emissions and the thermal stress by maintaining the production efficient.

The problem of finding the best trade-off between production and emissions (of course we can consider more factors) can be arranged like an optimisation problem. Usually such problems are solved by minimizing (or maximizing) a function through the concurrent fulfilment of some constraints, often conflicting each others (multiobjective optimisation, MOOP).

This kind of problems can be solved with a single-objective function approach, which is a combination of various objective functions, but also with a multiobjective approach based on the Pareto Theory.

C. Di Chio et al. (Eds.): EvoApplications 2010, Part II, LNCS 6025, pp. 151–160, 2010.

Evolutionary Computation (EC) methods can be considered a good choice to cope with multiobjective optimisation problems and several applications with effective results in different fields can be found in literature [1]. The main advantage of such methods consists in performing a parallel search of the optimal solution without a priori information about the problem.

In section 2 we introduce the multiobjective optimisation approach based on the Pareto Theory and we present the implemented evolutionary algorithm. In section 3 we describe the problem of the start-up optimisation of a combined cycle, in section 4 and 5 we respectively show the experimentations we made for this work and the results. Finally, in section 6 we comment the results and we give an orientation of our future work.

2 Multiobjective Optimisation

Optimisation techniques for solving Single-Objective Problems (SOPs) are largely developed and well known. However the modelisation of many real world problems leads to more than one and often conflicting objective functions. A problem dealing with two or more objectives is called Multiobjective Problem (MOP).

Since in MOPs objectives can be conflicting, such problems may lead to a set of solution instead of a single solution. Solutions belonging to this set are the result of a trade off between conflicting objectives and in order to define a set of good trade off solution the concept of dominance is introduced. Formally a solution dominates another solution if it is better at least in one objective and it is not worse in all objectives than the other solution. A set composed of all non-dominated solutions is called *Pareto optimal set* or *Pareto front*.

In MOPs, besides finding a set of solutions as close as possible to Pareto front, we need to mantain the solutions as diverse as possible. This because a well spaced set of solutions leaves the decision maker a wider choice of trade off between objectives.

Point-by-point methods have several disadvantages with respect to population based methods on complex problems with non-linear and non-convex spaces. Thus, in multiobjective optimisation Evolutionary Algorithms are largely used.

2.1 Multiobjective Evolutionary Algorithms

Multiobjective Evolutionary Algorithms (MOEAs) allow to find a set of non-dominated solution in each optimisation run. Because of their stochastic nature they are robust enough to tackle non-linear problems. The first algorithm that uses the non-dominated classification is the Multiobjective Genetic Algorithm (MOGA) proposed by Fonseca and Fleming in 1993 [2]. They proposed to assign a rank to each solution based on the number of solutions that dominates that one. This rank allows in some cases to compare two solutions without any fix-up like weights or other parameters. Subsequently many algorithms used non-dominated classification as Non-Dominated Sorting Genetic Algorithm (NSGA) proposed by Deb in 1994 [3] and then upgraded with elitism in 2000 with the name of Elitist

Non-Dominated Sorting Genetic Algorithm (NSGA-II) [4,5]. Since NSGA-II is a well known, efficent algorithm we use this MOEA for our work and we analyze it in detail. A survey about MOEAs can be found in [6].

In order to compare the quality of two solutions NSGA-II uses the ranking level approach. Given a population, this approach assignes rank level 1 to all non-dominated solution of the entire population, then it assignes rank 2 to all non-dominated solution of the population without solution of rank 1 and so on until it is assigned a rank to all solutions.

In order to mantain a diverse set of solution, it is assigned a crowding distance to each solution. This value is high for isolated solutions and low for solutions with many neighbors of the same rank. Extreme solutions are always taken, with an infinity crowding value, and other solutions are compared to their nearest neighbors (see figure 1).

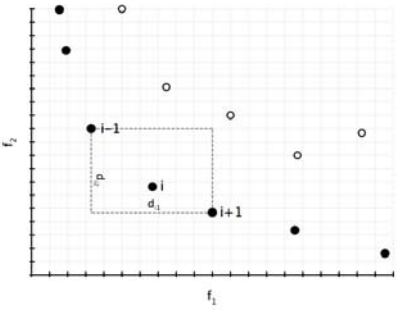


Fig. 1. Crowding Distance

3 Start-Up Optimisation of a Combined Cycle Power Plant

Gas and steam turbines are an established technology available in sizes ranging from several hundred kilowatts to over several hundred megawatts. Industrial turbines produce high quality heat that can be used for industrial or district heating steam requirements. Alternatively, this high temperature heat can be recuperated to improve the efficiency of power generation or used to generate steam and drive a steam turbine in a combined-cycle plant.

The gas-steam combined cycle produces electricity more efficiently than either gas or steam turbine alone because it performs a very good ratio of transformed electrical power per CO_2 emission. CC power plants are characterized as the 21st century power generation by their high efficiency and possibility to operate on different load conditions by reason of the variation in consumer load.

CC plants are highly complex systems but with the availability of high powerful processors and advanced numerical solutions there is a great opportunity to develop high performance simulators for modeling energy systems in order to consider various aspects of the system. This is a complex task including several

limitations that have to be fulfilled simultaneously like the maximum allowed thermal stress caused by temperature gradients. In literature one of the most studied problems of CC operation is the start-up optimisation.

As example, in [7] through a parametric study, the start-up time is reduced while keeping the life-time consumption of critically stressed components under control. In [8] an optimum start up algorithm for CC, using a model predictive control algorithm, is proposed in order to cut down the start-up time keeping the thermal stress under the imposed limits. In [9] a study aimed at reducing the start-up time while keeping the life-time consumption of the more critically stressed components under control is presented. In this work the optimisation of the start-up procedure for a CC power plant has been studied by means of a system simulator.

In general, most studies on CC are based on simulators and the goal is to minimise the start-up time alone in single-objective approach managing all other important aspects of the problem, like thermal stress or pollutant emissions, as constraints. In this way the global operations are not optimised because an effective start-up optimisation would be multiobjective.

The start-up scheduling is as follows (see figure 2). From zero to time t_0 ($\approx$ 1200 sec) the rotor engine velocity of the gas turbine is set to 3000 rpm. From time t_0 to t_1 the power load is set to 10 MW and then the machine keeps this regime up to time t_2. All this initial sequence is fixed. From time t_2 to t_3($\approx$ 3600 sec) the machine must achieve a new power load set point which has to be set optimal and then the machine has to keep this regime up to time t_4. The time lag $t_4 - t_3$ has to be optimised and during this interval the steam turbine starts with the rotor reaching the desired velocity. Then the turbines have to reach at time t_5 the normal power load regime (270 MW for the gas turbine) according to two load gradients which need optimising.

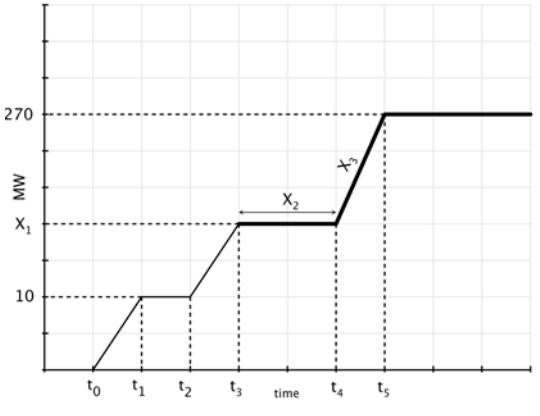


Fig. 2. Start-up sequence

Therefore, the process variables to be optimised with their operating ranges are visible in Table 3

Table 1. Process Variables

Name	Description	Range	Measure Unit
X_1	Intermediate power load set point	$[20, 120]$	MW
X_2	Intermediate waiting time $t_4 - t_3$	$[7500, 10000]$	sec
X_3	Gas turbine load gradient	$[0.01, 0.2]$	MW/s
X_4	Steam turbine load gradient	$[0.01, 0.2]$	$\%/s$

In order to optimise the overall start-up operations, the following objectives should be fulfilled:

1. minimise fuel consumption (Kg)
2. minimise time (s)
3. maximise energy production (KJ)
4. minimise pollutant emissions ($\frac{mg \cdot s}{Nm^3}$)
5. minimise thermal stress

It is therefore obvious that a multiobjective approach would improve the overall performance of such a power plant with a remarkable positive effect on the environment. Thus, we have approached the start-up problem in a multiobjective way by optimising the mentioned criteria with multiobjective genetic algorithms. Experimentations has been carried out on data obtained by means of a software simulator provided by AnsaldoEnergia.

4 Experimentations

We applied our implementation of the NSGA-II algorithm on the multiobjective optimisation of the problem described in Section 3 and we compared it to the following algorithms:

1. RAND: A random search algorithm
2. WSGA: Weighted-Sum Genetic Algorithm
3. NSGA-II: Non-Dominated Sorting Genetic Algorithm

The RAND algorithm is a trivial random search in the input space, with the same number of overall fitness evaluations of the other algorithms. At the end of this sampling, all the non-dominated solutions are considered inside the Pareto Front.

The WSGA applies a weighted sum of all objectives in order to reduce the original MOP to a single objective one. At each run a Genetic Algorithm is executed with a different random convex combination of the weights of the fitness function.

All the algorithms were executed 10 times and the resulting non-dominated set of the union of the Pareto fronts obtained at the end of each run was taken. In order to fairly compare the algorithms, each one is run over the same number of fitness evaluations.

Each input variable (see Table 3) can assume 21 different values, we encoded the decision variables with a Gray code binary string, whose minimal length can be obtained by:

$$\log_2 21^4 \approx 18 \tag{1}$$

The encoding is simple, we enumerated all the solutions (with numbers from 1 to 21^4) assigning each value of the 18-bit string to a solution.

Table 4 describes the algorithms' parameters used during our tests.

Table 2. Algorithm Parameters

	NSGA-II	WSGA
Population Size	100	50
Generations	50	30
Selection	Binary Tournament	
Crossover	Single Point	
Crossover Probability	0.75	
Mutation	Bitwise	
Mutation Probability	1/18	

To evaluate the performance of the different methodologies we used the following metrics:

- Dominance Ratio
- Spacing
- Hypervolume

Dominance Ratio metric was suggested by Zitzler in 1999 [10]. It compares two fronts and returns the percentual of solutions of the first one dominated by the second one, with respect to all the solutions of the first front.

Therefore, the smaller the value the better the first front respect to the second is and a value of one implies that the first front is completely dominated by the second.

Spacing metric, proposed by Schott in 1995 [11], evaluates relative distance between consecutive solutions belonging to non-dominated set. It is related to the spread of each non-dominated set independently. The lower spacing value is the more uniform the distribution of solutions is. Since in MOPs we need to mantain the set of solutions as diverse as possible, as mentioned earlier in section 2, a uniform distribution of solutions is highly preferred.

Hypervolume metric was proposed by Zitzler and Thiele in 1999 [10]. It evaluates both dominance and spreading of solutions. This metric calculates the area covered by the hypervolume whose vertices are the solutions set and a reference point, a vector of worst values each objective function can assume. Since no scaling is used, a good spread of high magnitude solutions in an objective implies a better performance value with respect to a good spread of low magnitude solutions in another objective. In order to reduce the computational load of

such metric we used a Monte Carlo estimation of the hypervolume, considering the percentage of random points which are dominated by the Pareto front we are measuring. As reference point we considered the worst values among all the solutions of the Pareto fronts considered.

Even if in real-world problems the real optimal Pareto front usually is not available, we computed, for a complete comparison of the selected algorithms, the fitness values of all the points inside the solution space. Despite it was computationally expensive (it took several days on a cluster with 1024 CPUs) we have the real optimal Pareto front

In order to show graphically the behavior of the algorithms we tested the problem firstly for only two of the five objectives described in section 3. We considered two clearly conflicting objectives: maximising energy production while minimising pollutant emissions. Pareto fronts' graph and relative metrics are presented. Subsequently we considered the problem with all five objectives and we present only the related performance metrics results since the plot of Pareto Fronts is not possible.

5 Results

We show results of two and five objectives optimisation obtained with 10 run of all algorithms but RAND (see Section 4).

We performed a multiobjective optimisation considering two objectives and five objectives. In the first case we considered the maximisation of energy production and minimisation of pollutant emissions and the overall number of fitness evaluations is 15300. In the second case we have the same number of evaluations.

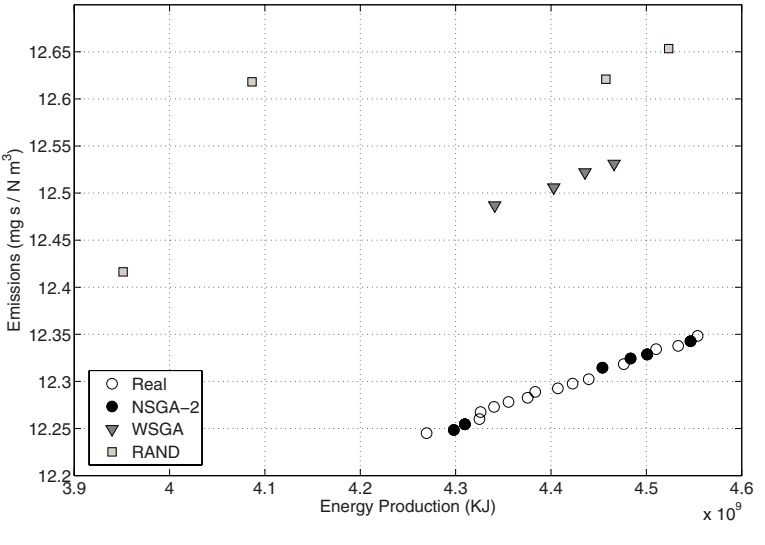


Fig. 3. Pareto Fronts plot on the 2-objectives problem

Table 3. Dominance Ratio

	Real		NSGA-II		RAND		WSGA	
Dimensions	2D	5D	2D	5D	2D	5D	2D	5D
Real	-	-	0	0	0	0	0	0
NSGA-II	0	0.659	-	-	0	0.406	0	0.008
RAND	1	0.637	1	0.014	-	-	0.5	0.001
WSGA	1	0.5	1	0.1	0	0.1	-	-

Table 4. Spacing, hypervolume and size of the real optimal Pareto front and the ones obtained by the considered algorithms

	Real		NSGA-II		RAND		WSGA	
	2D	5D	2D	5D	2D	5D	2D	5D
Spacing	0.015	0.007	0.002	0.07	0.013	0.023	0.004	0.376
Hypervolume	0.93	0.394	0.898	0.348	0.069	0.37	0.338	0.129
Size	20	15608	11	261	4	2435	4	10

Figure 3 shows that the NSGA-II Pareto front is overlapping with the real one and therefore it dominates all the solutions of the other algorithms while, as expected, the RAND front is dominated by both. The metrics values presented in Tables 3 and 4 reflect this situation. The columns labeled "2D" are related to the experimentations with 2 objectives and, similarly, for the problem with 5 objectives. The last line of Table 4 shows the number of solutions for different Pareto fronts.

For the 5-objectives problem we can't plot directly the Pareto Fronts and so we have to establish the comparison between the algorithms on the metrics' values. We can observe that the size of fronts of the algorithms shows an evident variability: from 10 (WSGA) to 2435 (RAND) and the same we can assert the same for spacing, WSGA shows that the solutions in its front cover a larger space than other two algorithms.

5.1 Discussion

In two dimensions the results we obtain aren't much different from the ones we expected: the ability of Evolutionary Computation based algorithms like NSGA-II permits to explore effectively the solution space and find the best solutions, achieving a Pareto front far better than those obtained with WSGA or random search.

With 5 dimensions the situation changes drastically. The RAND algorithm becomes the best algorithm, achieving a Pareto front which dominates about the 40% of solutions of the NSGA-II's front and the nearest hypervolume to the optimal one. A probable explanation of this situation should be found in the last line of Table 4, where we can observe that the size of the real optimal Pareto

front is about 800 times larger than the optimal one with two objectives. This means that random search is more effective because it's simpler to find randomly good solutions than in the 2D space.

It's an interesting observation the fact that the solution proposed from the plant manager results dominated in both the problem spaces, in 2 and 5 dimensions, by all the algorithms we tested. Therefore, all the solutions provided by the algorithms should be considered "better" (from a multiobjective point of view) than the real used ones.

6 Conclusion and Future Work

We underlined the capability of multiobjective optimisation techniques of providing a set of feasible solutions among which a decision can be taken. We made our experimentations on a precise software simulator of a combined cycle plant considering two and five objectives functions.

Considering only a subset of the objectives (maximisation of energy output and minimisation of pollutant emissions) we observe that NSGA-II algorithm works far better than a random search and a combined single-objective algorithm, finding solutions on the real optimal Pareto front. With all the objectives the situation changes and the results of a random search outperform the Evolutionary Computation based approach.

Despite these results seem inconsistent, we think that it is not simple to estimate the performances of a set of algorithms when increasing the number of considered objectives, because in real problems the objectives function to minimise (or maximise) are heterogeneous, i.e. the relation between results in low and high dimensional space is not straightforward. In the real case we considered, a deeper study of objective functions is needed, in order to explore mutual relations between them.

Although the primary goal of this paper is to highlight the application of multiobjective optimisation to a real world problem, comparisons can be extended also to other MOEA for a more complete overview.

Moreover this work raises the issue of reducing the computational load of stochastic algorithms such the ones we used of real problems, where the evaluation of a solution is based on the execution of a software simulator, which reflects the complexity of the problem it simulates. We think that such problem can be coped with by considering an algorithm which uses both the real fitness function and an approximated one, in order to lower the number of executions of the computationally expensive software simulator.

References

1. Cutello, V.: A multi-objective evolutionary approach to the protein structure prediction problem. Journal of The Royal Society Interface 3, 139–151 (2006)
2. Fonseca, C.M., Fleming, P.J.: Genetic algorithms for multi-objective optimization: Formulation, discussion and generalization. In: Proceedings of the Fifth International Conference on Genetic Algorithms, pp. 416–423 (1993)

3. Srinivas, N., Deb, K.: Multiobjective optimization using nondominated sorting in genetic algorithms. Evolutionary Computation 2(3), 221–248 (1994)
4. Deb, K., Agrawal, S., Pratap, A., Meyarivan, T.: A Fast Elitist Non-dominated Sorting Genetic Algorithm for Multi-objective Optimization: NSGA-II. In: Schoenauer, M., Deb, K., Rudolph, G., Yao, X., Lutton, E., Merelo, J.J., Schwefel, H.-P. (eds.) PPSN VI 2000. LNCS, vol. 1917, pp. 849–858. Springer, Heidelberg (2000)
5. Deb, K.: Multi-Objective Optimization Using Evolutionary Algorithms. Wiley, Chichester (2001)
6. Coello, C.A.: An updated survey of ga-based multiobjective optimization techniques. ACM Comput. Surv. 32(2), 109–143 (2000)
7. Alobaida, F., Postlera, R., Ströhlea, J., Epplea, B., Kimb, H.-G.: Modeling and investigation start-up procedures of a combined cycle power plant. Applied Energy 85(12), 1173–1189 (2008)
8. Tetsuya, F.: An optimum start up algorithm for combined cycle. Transactions of the Japan Society of Mechanical Engineers 67(660), 2129–2134 (2001)
9. Casella, F., Pretolani, F.: Fast Start-up of a Combined-Cycle Power Plant: a Simulation Study with Modelica. In: Proceedings 5th International Modelica Conference, Vienna, Austria, September 6-8, pp. 3–10 (2006)
10. Knowles, J., Corne, J.: On metrics for comparing non-dominated sets. In: Proceedings of the 2002 Congress on Evolutionary Computation (CEC 2002), vol. 1, pp. 711–716 (2002)
11. Schott, J.: Fault tolerant design using single and multicriteria genetic algorithm optimization. Masters thesis Department of Aeronautics and Astronautics. Massachusetts Institute of Technology (1995)

A Study of Nature-Inspired Methods for Financial Trend Reversal Detection

Antonia Azzini[1], Matteo De Felice[2,3], and Andrea G.B. Tettamanzi[1]

[1] Università degli Studi di Milano, Dipartimento di Tecnologie dell'Informazione
{antonia.azzini,andrea.tettamanzi}@unimi.it
[2] ENEA (Italian Energy New Technology and Environment Agency)
[3] Università degli Studi "Roma Tre", Dipartimento di Informatica e Automazione
defelice@dia.uniroma3.it

Abstract. This paper presents an application of two nature-inspired algorithms to the financial problem concerning the detection of turning points. Nature-Inspired methods are receiving a growing interest due to their ability to cope with complex tasks like classification, forecasting and anomaly detection problems. A swarm intelligence algorithm, Particle Swarm Optimization (PSO), and an artificial immune system one, the Negative Selection (NS), are applied to the problem of detection of turning points, modeled as an Anomaly Detection (AD) problem, and their performances are compared. Both methods are found to give interesting results with respect to an unpredictable behavior.

1 Introduction

A typical financial market action consists of alternating up- and down-trends, separated by turning points, which correspond to local maxima or minima values of the prices of a financial instrument. One of the most important objectives of financial market forecasting techniques is to detect turning points consistently and correctly. Such price movements are not regular and the application of common forecasting tools does not provide satisfactory results. In this sense, the task of predicting the overall price movements of a security in the next period can be modeled as an anomaly detection problem.

In behavior-based approaches to anomaly detection, the model of normal behavior is constructed from an observed sample of normally occurring patterns. In financial problems, a normal pattern of behavior for the price of a security may be defined as a continuing trend, whereas an anomaly in this setting would consist of a trend reversal. The problem of anomaly detection can be naturally recast into a classification problem, and machine learning and nature-inspired algorithms seem to be promising in such classification tasks, being able to discover satisfactory solutions with unbalanced datasets (the number of instances of normal behavior is, by definition, greater than the number of anomalies). In particular, in this paper we employ two types of nature-inspired algorithms, namely

C. Di Chio et al. (Eds.): EvoApplications 2010, Part II, LNCS 6025, pp. 161–170, 2010.

Particle Swarm Optimization and Negative Selection applied to the creation of an optimal hyperspherical detector set. The former algorithm leads to a positive selection mechanism whereas the latter uses a negative selection process, defining the anomalous space as the complementary of the normal one. The performance of the two approaches are compared with the aim of assessing which one is more suited for trend reversal detection.

This paper is organized as follows. Section 2 presents the financial problem, while the detectors models and the evolutionary algorithms are stated in Section 3. Section 4 presents the experiments carried out and discusses the results obtained. Section 5 provides some concluding remarks.

2 Problem Description

As a general principle, wherever there is an oscillation or a vibration, whether in the tides of an ocean or in the heat of a system or in a pendulum, it should be possible to extract some energy by means of an implement that acts so as to damp the oscillation. If the mass or energy of the oscillating system were very large with respect to the energy-extracting implement, the damping caused by energy extraction would be negligible and the process could continue indefinitely.

Even the most casual observer of a financial time series will notice that prices of financial instruments move up and down [10]; furthermore, this behavior happens and can be observed at all scales [11]. Therefore, a trader on the financial markets should be able to extract a profit from the oscillations in the price of a security, much like one should be able to extract energy from an oscillating physical system, the liquidity of a security being the financial notion corresponding to the mass of the system under exploitation.

An obstacle to putting this idea in practice is that the price movements of real securities traded on financial markets are not regular and look, at least to some extent, unpredictable.

However, if an algorithm were available that is able to predict with a less than 50% error the overall direction of price movement in the next period, an arbitrary profit could be extracted from trading a very liquid security, provided that a sound money management strategy is enforced [14].

We equate the task of predicting the overall price movement of a security in the next period to the task of detecting an anomaly in a system. More specifically, a normal pattern of behavior for the price of a security may be defined as continuing a trend, whereas an anomaly, in this setting, would consist of a trend reversal.

To demonstrate the viability of such an idea, we approach here a very simple instance of the above general problem, where daily series of an index are considered and a reversal is defined as a change in the sign of the log return of the next period with respect to the previous period.

3 Detector Set Definition and Nature-Inspired Methodologies

Our main goal is to obtain a set of detectors covering the feature space in the best way for this particular anomaly detection problem. Generally different kinds of models are used to define detectors, depending on the approach. Some use neural networks (NNs), while some others are based on geometrical solutions. Hyperspherical Detectors consist of a set of coordinates, representing a point in the n-dimensional feature space, and a radius value (hyperradius). This work considers sets of hypershperical detectors as defined in [1], working in an eight-dimension space. In particular we use a vector-based representation where each detector d is described by its coordinates $x_1, \ldots, x_n$ in the n-dimensional space and its radius r, with $d = (x_1, \ldots, x_n, \quad r)$.

An observation (i.e., a point in the feature space) is considered non-anomalous if the distance (or similarity) between the point and the detector's coordinates is less than or equal to the radius. As a measure of distance (or similarity), Euclidean distance will be used. The main advantage of this detector type is the ease of representation and implementation.

Models of normal behavior can represent either the set of allowed patterns (positive detection) or the set of anomalous patterns (negative detection). This allows us to study the financial price movements either with a positive selection algorithm, as the Particle Swarm Optimization (PSO), or a negative one with the negative selection algorithm of the Artificial Immune Systems. Their basic steps are briefly reported in the following section. This work considers the financial instrument S&P 500. The overall behavior of this index is represented by putting all the points of the considered dataset in the eight-dimension space.

The two implemented approaches follow the commonly accepted practice of machine learning by defining a training and a test set, used, respectively, to create the detector sets and to test their generalization capabilities. In particular the training set contains only normal cases, while the test set also contains anomalies. The hyperspherical detectors found by each of the two methods through the training set will then cover, respectively, the normal or the anomalous behaviors in that space during the test phase.

3.1 Particle Swarm Optimization

Particle Swarm Optimization (PSO) is a swarm intelligence optimization technique [3,12] where each individual (or particle) moves in the solution space following two main attraction points: the best solution it has encountered so-far and the best solution found by any other individual in the swarm. The algorithm implemented follows the common steps of a PSO algorithm, with the parameters defined in Table 2.

We used the algorithm in order to create a set of detectors that are able to maximize the coverage of the normal behaviors in the training phase and to minimize the radius of each detector. We implemented an algorithm that, for each iteration, removes from the dataset all the points covered by the best

detector found in that iteration. In this way at the end of the algorithm we obtain a set of detectors, each of which covers in an optimal way a subset of points. Maximum and minimum values are also defined for the radius of each particle, in order to avoid disruptive effects on the detection of unseen points in the space. The fitness function of particles in the PSO algorithm is given by Equation 1:

$$f(\mathbf{x}, r) = \frac{n^d}{r^d}, \tag{1}$$

where $\mathbf{x}$ is the vector with the coordinates in the solution space of the detector, r is the radius, n is the number of points covered by the detector, and d is the number of dimensions of the solution space.

The entire process of the particle swarm optimization algorithm is also reported in Algorithm 1.

Algorithm 1. Particle Swarm Optimization Algorithm for Detectors Optimization

```
dataset ← load_dataset()
not_covered_points ← size(dataset)
detectors_set ← ∅
while not_covered_points > 0 do
    BEST_DETECTOR ← PSO_ALGORITHM(dataset, particles_number, max_iterations, max_radius)
    dataset ← dataset \ data_covered_by(BEST_DETECTOR)
    not_covered_points ← size(dataset)
    detectors_set ← detectors_set ∪ BEST_DETECTOR
end while
```

3.2 Negative Selection

Negative Selection (NS) is one of the major algorithms developed in the field of AIS [8]. Its aim is to define a set of detectors that cover the complementary space to the normal one, in order to classify new, unseen data as normal or anomalous.

When the hyperspherical detector model is considered, each normal sample is defined as a hypersphere centered in c_i with constant radius r_s, i.e. $s_i = (c_i, r_i)$, $i = 1, \ldots, l$, where l is the number of input samples. Also the detectors d are represented as hyperspheres: $d_j = (c_j, r_j)$, with $j = 1, \ldots, p$, where p is the number of detectors generated, with center c_j and radius r_j. In the first step, the radius of detectors is randomly generated within a given range, and the detector set is generated by checking whether each hypersphere generated contains at least one normal sample: only the detectors that do not include them are kept; the others are discarded. In this algorithm the radius of a new detector is set to the Euclidean distance of the nearest normal case from its center c_j in the range $[0, \texttt{MAX\_RADIUS}]$. The general steps of the negative selection algorithm are reported in Algorithm 2 and its parameters are shown in Table 2.

Algorithm 2. Negative Selection algorithm for anomalous space covering

```
dataset, NDETS, dets_options ← load_dataset(dataset, NDETS, options)
detectors_set ← ∅
while number of detectors < NDETS do
    det(c,r) ← create_detector(dets_options)
    if Euclidean_distance(det, dataset) < det_radius then
        discard_detector()
    else
        detector_set ← add(det)
    end if
end while
anomalous_check_detection(detector_set,dataset)
```

4 Experiments and Results

In our experiments we have considered a time series of daily prices of the S&P 500 index from April the 14th, 1992 until September the 30th, 2009. As previously stated, the considered time series has been divided into two parts in order to define the training and the test sets. In particular a selection of 2000 normal cases (i.e. continuing a trend) will be chosen from the first part in order to create the training set, while the most recent 400 cases (containing both normal and anomalous behaviors in the same quantity) will be used in the test set to evaluate the generalization capabilities on unknown data. A graphical representation of the time series is depicted in Figure 1.

For each day of the time series, eight input values are defined by considering the log return of the daily historical prices and 7 different technical indicators for the same financial instrument. Such technical indicators correspond to some of the most popular indicators used in technical analysis and summarize important features of the time series of the considered financial instrument [5]. They also represent useful statistics and technical information, that otherwise should be calculated both by PSO and NS, increasing the computational cost of each algorithm. A brief description of the inputs is reported in Table 1.

Each input value represents a coordinate in the multi-dimension space considered. Therefore, each pattern of the dataset identifies a point in the eight-dimension space. All values in the dataset are preprocessed by considering a normal distribution with mean 0 and standard deviation set to 1.

The target of our detection problem is defined for each day i of the time series with a value equal to 0 or 1, corresponding respectively to a normal behavior defined as continuing a trend, or an anomaly, when a trend reversal occurs. The target setting is defined by Equation 2:

$$\text{target}(t) = \begin{cases} 0 \text{ if } \operatorname{sgn}\left(\log\left(\frac{C(t)}{C(t-1)}\right)\right) = \operatorname{sgn}\left(\log\left(\frac{C(t+1)}{C(t)}\right)\right), \\ 1 \text{ otherwise,} \end{cases} \tag{2}$$

where $C(t)$ represents the closing price of the financial instrument at time t.

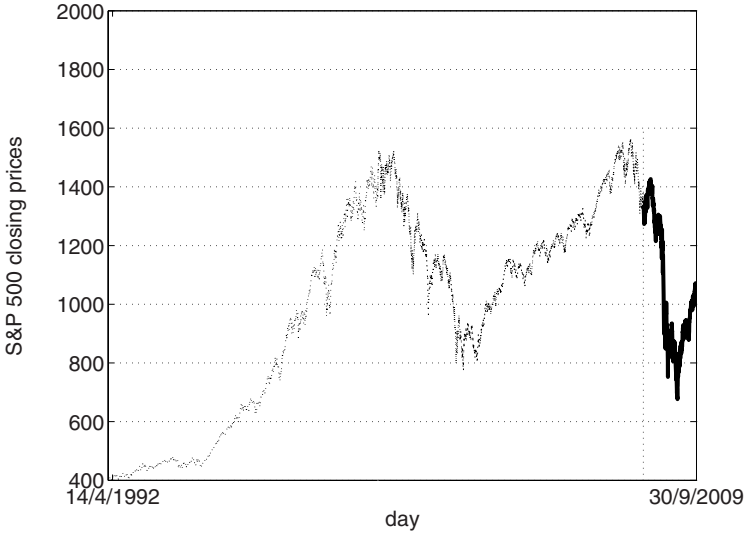


Fig. 1. S&P 500 time series. A selection of 2000 normal cases in the fine part of the shape will define the training set. The thick black part of the shape (the last 400 cases) corresponds to the test set.

In this work a first group of experiments is carried out in order to find the optimal settings of the parameters for both algorithms, listed in Table 2. We have performed 10 runs for each setting defined in the two approaches. Generally, for each of the two algorithms, we noticed that, hyperspheres with too small a radius will not be able to well cover the anomalies in the space, but, on the other hand, a large radius could have disruptive effects, since the hyperspheres will also cover normal cases, reducing the overall accuracy. The latter represents a critical aspect for the maximum radius. For example, in NS, a maximum radius set to the value 3 was too small, while maximum radius equal to 10 was not enough to improve the performances obtained with values set to 5, with high computational effort. Similar considerations have been carried out about the space in which the

Table 1. Input Technical Indicators

Index	Input Technical Indicator	Description
1	r_i	Log Return
2	MA50(i)	50-day Moving Average
3	EMA50(i)	50-day Exponential Moving Average
4	RSI(i)	Relative Strength Index
5	MACD(i)	Moving Average Convergence/Divergence
6	SIGNAL(i)	Exponential Moving Average on MACD
7	Momentum(i)	Rate of price change
8	ROC(i)	Rate Of Change

Table 2. Algorithms parameters

Approach	Parameter Description	Value
PSO	Feature's Space	$-5, 5$
	Maximum Velocity of the point	2
	Hypersphere Minimum Radius	10^{-4}
	Hypersphere Maximum Radius	1
NS	Feature's Space	$-5, 5$
	Hypersphere Minimum Radius	10^{-4}
	Hypersphere Maximum Radius	5

features are defined. The experiments show that both algorithms perform well on a solution space defined in the range $[-5, 5]$. Then, a second group of experiments was carried out by using the best parameter settings reported in Table 2.

At the end of each simulation, two different metrics, accuracy and AUC (Area Under Curve), are computed on the output vector in order to compare the performances for both methods. Accuracy is a relatively straight metric, as reported in Equation 3,

$$\text{accuracy}(S) = \frac{\text{TP} + \text{TN}}{\text{P} + \text{N}}, \tag{3}$$

where TP and TN are, respectively, the number of anomalous points detected as anomalous and the number of normal points correctly detected. P and N refer to the total number of anomalous (positive) and normal (negative) points. Since S values of the accuracy, also reported in Table 3, are obtained from a balanced dataset of normal and anomalous behaviors, no further normalization is needed for TP and TN values. The second parameter, AUC, corresponds to the area of the surface under the ROC (Receiver Operating Characteristic) curve. This curve is the graphical representation of true positives (TPR, the true positive rate) and false positives (FPR, the false positive rate) as the discrimination threshold is varied. The threshold defines the minimum number of detectors that should cover a point in the space in order to identify it as an anomalous behavior in such space. The performances of the two approaches also depend on the computational effort, represented in this work by calculating the number of evaluations of the distance function in the algorithms. Such function has been chosen since it represents a fundamental aspect for the fitness evaluation of the PSO algorithm and for the detector generation in the NS algorithm.

4.1 Discussion

A first group of considerations is suggested by Figure 2, showing the ROC curve and the AUC area of two of the best solutions found for both algorithms. This graphical representation underlines in a straightforward manner the performances of PSO and NS. In particular we can see how PSO and NS always work better than a random choice algorithm (the gray line reported in Figure 2, and generally how NS outperforms the PSO.

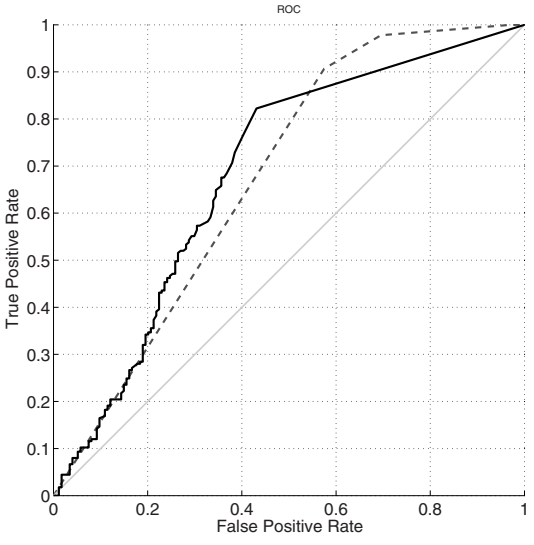


Fig. 2. ROC curve of two of the best solutions of both algorithms calculated over the test set. The thick black line is the NS algorithm with 10 millions of detectors, the black dashed line is the best solution of a PSO with 1000 individuals and 300 iterations.

As we expected, performances of two algorithms are quite different. Indeed, despite the shown results present similar accuracy (or AUC) values, particular attention has to be given to the performances with respect to number of distance function evaluations. In Figure 2, for example, the best NS solution is obtained with a number of distance function evaluations equal to $4.80 \cdot 10^{10}$, while the best PSO one is obtained with $7.78 \cdot 10^{10}$ evaluations.

The simplicity of NS is an important feature. Indeed, it could be considered as a "brute force" approach: it creates a large number of simple detectors only by selecting those that do not include the points of the training dataset. Instead, PSO, like other bio-inspired search heuristics, tries to optimize a set of initial points with respect to a particular fitness function.

One of the most critical aspects about the PSO regards the tuning of all the parameters defined in the algorithms, in particular due to the fact that often there are no empirical rules helping in this choice but only trial-and-error approaches. On the other hand, NS could require a huge amount of detectors in order to cover the space, increasing the computational effort of the approach, as reported in the first column of Table 3. However the small and quite similar values obtained for the standard deviations of accuracy and AUC rates for all the experiments carried out allow us to define these as two stable methodologies.

We can see that, by doubling the number of distance function evaluations, the performances increase about $7 - 8\%$ in the PSO algorithm, while with negative selection we observe smaller increments. In particular, the accuracy increases of about 3% from 500,000 to 1 million and from 5 to 10 millions of detectors; while

Table 3. Average results obtained by the two algorithms

Description	Accuracy	AUC	TP	TN	FP	FN	Dist. f. evals.
PSO							
100 ind., 200 iter.	0.60 ± 0.01	0.56 ± 0.02	0.52	0.09	0.35	0.05	$1.23 \cdot 10^{10}$
100 ind., 500 iter.	0.64 ± 0.02	0.60 ± 0.02	0.52	0.12	0.32	0.04	$2.49 \cdot 10^{10}$
500 ind., 100 iter.	0.64 ± 0.01	0.60 ± 0.01	0.51	0.13	0.31	0.05	$2.30 \cdot 10^{10}$
1000 ind., 300 iter.	0.67 ± 0.02	0.65 ± 0.01	0.50	0.17	0.27	0.07	$7.78 \cdot 10^{10}$
NS							
500,000 ind.	0.63 ± 0.01	0.64 ± 0.01	0.34	0.29	0.14	0.22	$1.20 \cdot 10^{9}$
750,000 ind	0.64 ± 0.01	0.64 ± 0.01	0.36	0.28	0.15	0.20	$1.80 \cdot 10^{9}$
1,000,000 ind.	0.65 ± 0.02	0.65 ± 0.01	0.37	0.28	0.16	0.19	$2.40 \cdot 10^{9}$
5,000,000 ind.	0.68 ± 0.01	0.68 ± 0.01	0.43	0.26	0.18	0.13	$1.20 \cdot 10^{10}$
10,000,000 ind.	0.70 ± 0.01	0.68 ± 0.01	0.46	0.24	0.19	0.11	$2.40 \cdot 10^{10}$
20,000,000 ind.	0.71 ± 0.01	0.69 ± 0.01	0.47	0.23	0.20	0.08	$4.80 \cdot 10^{10}$

only a rise of 1.5% is shown when the detectors increase from 10 to 20 millions. Such an aspect can be due to the fact that, in all the experiments carried out by considering different parameter settings, NS starts with higher values both for accuracy and AUC. Therefore, their increment is slow with respect to the PSO algorithm.

Furthermore, by comparing the false positive and the false negative errors of NS and PSO, we notice that they occur in different areas of the space. This aspect could suggest that a possible combination of both algorithms could exploit this difference to provide a better space coverage, by reducing the values of such errors.

5 Conclusion and Future Work

This work presents a comparison of two well known Soft Computing algorithms, the Negative Selection and the Particle Swarm Optimisation, applied to the complex task of detecting the "turning points" of the financial time series of the S&P 500 index. Particular attention has been paid to the comparison of the computational effort of both algorithms, in order to provide a fair basis of comparison.

NS results clearly outperform the PSO algorithm, by considering performances obtained with the same number of distance function evaluations. As we discussed in Section 4.1, an in-depth study of the PSO parameters tuning and a joint PSO-NS solution could become helpful in the improvement of the overall performance.

Further steps in the investigation of this problem will also consider an "online" anomaly detection approach, for example by considering the data of the S&P 500 market index updated every 15 seconds during trading sessions. We think that in that case the features of swarm intelligence algorithms would prove useful.

References

1. Azzini, A., De Felice, M., Meloni, S., Tettamanzi, A.G.B.: Soft computing techniques for Internet backbone traffic anomaly detection. In: Giacobini, M., et al. (eds.) EvoWorkshops 2009. LNCS, vol. 5484, pp. 99–104. Springer, Heidelberg (2009)
2. Balachandran, S., Dasgupta, D., Nino, F., Garrett, D.: A framework for evolving multi-shaped detectors in negative selection. In: Proc. of IEEE Symposium on Foundations of Computational Intelligence, FOCI 2007, pp. 401–408 (2007)
3. Bonabeau, E., Dorigo, M., Theraulaz, G.: Swarm intelligence: From natural to artificial systems. Oxford University Press, Oxford (1999)
4. Bouvry, P., Seredynsky, F.: Anomaly detection in TCP/IP networks using immune systems paradigm. Computer Comm. 30, 740–749 (2007)
5. Colby, R.: The Encyclopedia of Technical Market Indicators, 2nd edn. McGraw-Hill, New York (2002)
6. Dasgupta, D., Ji, Z.: Real-valued negative selection algorithm with variable-sized detectors. In: Deb, K., et al. (eds.) GECCO 2004. LNCS, vol. 3102, pp. 287–298. Springer, Heidelberg (2004)
7. De Castro, N.L., von Zuben, F.J.: Learning and optimization using the clonal selection principle. IEEE Trans. on Evolutionary Computation 6(3), 239–251 (2002)
8. Forrest, S., Perelson, A., Allen, L., Cherukuri, R.: Self-nonself discrimination in a computer. In: Proc. of the IEEE Symposium on Research in Security and Privacy, Los Alamitos, CA, pp. 202–212 (1994)
9. Freitas, A.A., Timmis, J.: Revisiting the Foundations of Artificial Immune Sustems for Data Mining. Trans. on Evolutionary Computation 11(4) (August 2007)
10. Herbst, A.: Analyzing and Forecasting Futures Prices. Wiley, New York (1992)
11. Peters, E.: Chaos and Order in the Capital Markets, 2nd edn. Wiley, New York (1996)
12. Kennedy, J., Eberhart, R.: Particle swarm optimization. In: Proc. of IEEE International Conf. on Neural Networks, Perth, Australia, vol. 4, pp. 1942–1948 (1995)
13. Kim, J., Bentley, P.J., Aickelin, U., Greensmith, J., Tedesco, G., Twycross, J.: Immune system approaches to intrusion detection - a review. Natural Computing 6, 413–466 (2007)
14. Vince, R.: The Handbook of Portfolio Mathematics: Formulas for optimal allocation & leverage. Wiley, New York (2007)

Outperforming Buy-and-Hold with Evolved Technical Trading Rules: Daily, Weekly and Monthly Trading

Dome Lohpetch and David Corne

School of MACS, Heriot-Watt University
Edinburgh, UK
dl73@hw.ac.uk, d.w.corne@hw.ac.uk

Abstract. Genetic programming (GP) is increasingly popular as a research tool for applications in finance and economics. One thread in this area is the use of GP to discover effective technical trading rules. In a seminal article, Allen & Karjalainen (1999) used GP to find rules that were profitable, but were nevertheless outperformed by the simple "buy and hold" trading strategy. Many succeeding attempts have reported similar findings. There are a small handful of cases in which such work has managed to find rules that outperform buy-and-hold, but these have tended to be difficult to replicate. Recently, however, Lohpetch & Corne (2009) investigated work by Becker & Seshadri (2003), which showed outperformance of buy-and-hold. In turn, Becker & Seshadri's work had made several modifications to Allen & Karjalainen's work, including the adoption of monthly rather than daily trading. Lohpetch et al (2009) provided a replicable account of this, and also showed how further modifications enabled fairly reliable outperformance of buy-and-hold. It remained unclear, however, whether adoption of monthly trading is necessary to achieve robust outperformance of buy-and-hold. Here we investigate and compare each of daily, weekly and monthly trading; we find that outperformance of buy-and-hold can be achieved even for daily trading, but as we move from monthly to daily trading the performance of evolved rules becomes increasingly dependent on prevailing market conditions.

Keywords: genetic programming, technical trading rules, data mining.

1 Introduction

There are several opportunities in the area of financial markets for advanced machine learning and optimization methods, and applications of evolutionary computation are now common in this area [1]. Genetic Programming (GP) [2,3,4] is a relatively popular technique in this field, with many studies reporting GP applications in finance (e.g. [5—12]). The focus in this paper is the area known as *technical analysis* [13—16]. Technical analysis is the name given to the general enterprise of forecasting the future direction of equity prices via the study of historical market price data. Technical analysis relies on the principle that patterns and trends exist in markets, and that these can be identified (for example by discovering rules) and exploited to predict price movements in the near future.

C. Di Chio et al. (Eds.): EvoApplications 2010, Part II, LNCS 6025, pp. 171–181, 2010.

Successful technical analysis uses tools such as moving averages (the mean price for a given stock or index over a given recent period), relative strength indicators (a function of the ratio of recent upward movements to recent downward movements), and others. A typical GP approach in this area is for rules to combine technical indicator 'primitives' with other mathematical operations. Such a rule constitutes a 'signal', which may be interpreted, for example, as a recommendation to buy if the signal is above a threshold. The first attempts to use GP in this way were by Chen and Yeh [5] and Allen and Karjalainen [7], and these and succeeding works regularly report that GP is able to find rules that are profitable on unseen future data. However, the 'elephant in the room' in such work has been a common and persistent failure for such rules to show greater returns than a standard "buy-and-hold" trading approach. The 'buy-and-hold' strategy is, for a given trading period, to buy the stock at the beginning of the period, and sell at the end – hence, always a good strategy in an upwardly moving market, and far simpler than using technical indicators.

Nevertheless, a small amount of research in this area seems able to find rules that outperform buy-and-hold [8,17,18]. In particular, GP-evolved technical trading rules with such success have been reported in Becker and Seshadri [19—21], who adopted the overall approach of Allen and Karjalainen [7] (who did not outperform buy-and-hold), and made several alterations. One of Becker & Seshadri's alterations was to adopt monthly trading rather than (as in Allen & Karjalainen) daily trading. That is, in [19], rules assume that trades will only be made (if at all) on the first day of the month, and hence deal with a less volatile view of the market. It is intuitively reasonable to suggest that this was an important feature of Becker & Seshadri's work, in the sense that outperformance of buy-and-hold may not have been achieved without this modification, however that hypothesis has not yet been tested. In this paper we test a modified version of Becker & Seshadri's approach and explore each of monthly, weekly and daily trading. We build on [24], which provided full details to enable replication of [19] as well as showing that a modified experimental setup led to more robust outcomes. In [24], outperformance of buy-and-hold was found to be robustly delivered by using Becker & Seshadri's approach [19] in the context of a monthly trading strategy, as long as the rules chosen for evaluation were selected according to performance over a validation period, and with the additional proviso that there was certainly some dependence on the specific data splits (training/validation/test) employed ([7] and [19] explored only one such data split). In the current work, we continue to evaluate this approach for several data splits, but in the context of weekly and daily trading too.

We end this section with a brief account of other related work which has attempted to outperform buy-and-hold. Potvin et al [12], for example, showed that GP trading rules can be generally beneficial in falling or stable markets – this is not particularly impressive, since buy-and-hold is naturally poor in such markets. In another line of work, risk metrics such as the Sharpe ratio [22] have been included in rules (or in their evaluation). Such metrics typically reduce the fitness of rules that promote trading in volatile conditions, and therefore lead to rules more likely to be applied by investors. For instance, building on Fyfe et al. [6] (not superior to buy-and-hold), Marney et al. [8,17] included risk metrics, while Cheng and Khai [10] using a modified Sterling return measure, but none of these attempts produced usable rules that compared well in comparison to buy-and-hold.

The incorporation of risk measures nevertheless seems a promising thread of work, but in this paper we concentrate on further exploring the performance of GP for evolving robust technical trading rules. In the remainder, we detail the overall approach (section 2), and summarize the findings of several experiments in section 3; we then have a concluding discussion in section 4, and point to where the reader may obtain our code for further experimentation.

2 Evolving Robust Trading Rules: The Modified AK/BS Approach

2.1 Overview

The approach we use is based on Becker & Seshadri's work [19,21] (BS) which in turn was a modification of Allen & Karjalainen's work [7] (AK). This approach uses standard genetic programming (GP), with a function set comprising arithmetic, Boolean and relational operators, while the terminal set comprises basic technical indicators, along with real and Boolean constants, and real-valued variables (such as *stock price*). An example of a specific rule found by [19] is in Fig. 1.

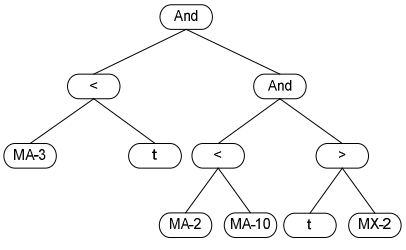


Fig. 1. Example of a trading rule

The rule in Figure 1 is to be interpreted as follows. **"**the 3-month moving average (MA-3) is less than the lower trend line (*t*) and the 2-month moving average (MA-2) is less than the 10-month moving average (MA-10) and the lower trend line (*t*) is greater than the second previous 3-month moving average maxima (MX-2)**"**. The rule therefore evaluates to either true or false. This translates into trading behaviour as follows: "If currently out of the market and the rule yields true, then *buy*; if currently in the market and the rule becomes false, then *sell*.". This procedure assumes a fixed amount to invest (e.g. $1,000) whenever there is a buy signal.

The remaining subsections explain the approach in further detail. Essentially we are explaining the approach in [19], making notes now and then to indicate where this departed from [7]. The data we use (as in [7,19—21] is the Standard and Poors 500 (S & P 500) index – a fixed set of 500 stocks which aggregate to daily price indicators (opening, closing, high, low). When considering weekly and monthly trading, the opening price (for example) for a week or a month is the opening price on the first day of that week or month.

2.2 Function and Terminal Sets

The function set is: `and`, `or` and `not`, together with the relational operators > and <. We use strongly typed GP, automatically ensuring, for example, that relational operators receive Boolean inputs. The terminal set is as follows, where 'unit' is either day, week or month, depending on whether we are evolving rules for daily, weekly or monthly trading:

- opening, closing, high and low prices for the current unit;
- 2,3,5 and 10-unit moving averages;
- Rate of change indicator: 3-unit and 12-unit;
- Price Resistance indicators: the two previous 3-unit moving average minima, and the two previous 3-unit moving average maxima;
- Trend Line Indicators: a lower resistance line based on the slope of the two previous minima; an upper resistance line based on the slope of the two previous maxima.

The l-unit moving average at time m is the mean of the closing prices of the l units from m back to $m-(l-1)$. The l-unit rate of change indicator measured at time m is: $(p(m)-p(m-(l-1))\times 100)/p(m-(l-1))$, where $p(x)$ indicates the closing price for time x. Previous maxima MX1 and MX2 are obtained by considering the 3-unit moving averages at each point in the previous 12 units. Of the two highest values, the one closest in time to the current is MX1, and the other is MX2. The two previous minima are similarly defined. Finally, to identify trend line indicators, the two previous maxima are used to define a line in the obvious way, and the extrapolated value of that line from the current time becomes the upper trend line indicator; the lower trend line indicator is defined similarly by using the two previous minima.

2.3 The Fitness Function

The fitness function has three aspects. First is 'excess return', which is how much would have been earned by using the trading rule, in excess of the return from a buy-and-hold strategy. The excess return is $E = r - r_{bh}$, where r is the return on an investment of \$1,000, and r_{bh} is the corresponding return that would have been achieved using a buy and hold strategy. To calculate r we use [7,19,21]:

$$r = \sum_{t=1}^{T} r_t I_b(t) + \sum_{t=1}^{T} r_f(t) I_s(t) + n \ln\left(\frac{1-c}{1+c}\right)$$

where: $r_t = \log P_t - \log P_{t-1}$ – indicating the continuously compounded return, where P_t is the price at time t. Meanwhile, $I_b(t)$ indicates the buy signal, and is 1 if the rule indicates *buy* at time t, 0 otherwise. Similarly defined is the sell signal, $I_s(t)$. The first component of r therefore calculates the return on investment over the times when the investor is (as guided by the rule) in the market. In the second component, $r_f(t)$ indicates the risk-free return, which is taken for any particular day t from US Treasury bill data (these data are available from http://research.stlouisfed.org/fred/data/irates/tb3ms). Hence, the second component represents time out of the

market, assuming that the investor's funds are earning a standard risk-free interest. Finally, the third component corrects for transaction costs. The cost of a single buy or sell transaction is assumed to be 0.05% (i.e. 0.005) – e.g. $5 for a transaction of volume $1,000. The number of transactions actioned during the period by the rule is *n*. This component estimates the compounded loss from the expenditure on transactions.

The other two aspects of the fitness function, introduced in [19], are a modification that penalizes rule complexity, and a further modification that considered 'performance consistency' (PC). The second main part of the fitness function, r_{bh}, is calculated as:

$$r_{bh} = \sum r_t + \ln(\frac{1-c}{1+c})$$

where r_t is as indicated above. Hence it calculates the return of buying at the first day and selling at the last day of the period, involving a single buy and a single sell transaction.

The excess return *E*, calculated as described, was originally the objective function in [7], but improvements in [19,21] arose from two adjustments. One of these is an adjustment to fitness according to the size of the tree. Given a fitness value *f*, the adjusted fitness becomes $5f/\max(5,depth)$, where *depth* is the depth of the tree being evaluated, and the constant 5 is a 'desired' depth. Clearly there are many potential alternatives, but we simply adopt the stated method used in [19,21]. The other aspect of the fitness function which led to more consistent results was as follows, which we call *Performance Consistency* (PC). *E* is calculated for each successive period of *K* units covering the entire test period. The value returned is simply the number of these periods for which *E* was greater than both the corresponding buy and hold return (from investing in the index over that period) and the risk-free return during that period.

Finally we can state the objective function *f* used in this work: the fitness of a GP tree was the PC-based fitness (i.e. a number from 0 to *X*, where there were *X* periods covering the test data), adjusted for tree complexity by $5f/\max(5,depth)$.

2.4 Operators and Initialization

We used the four mutation operators described by Angeline [3], as follows:

- Grow: randomly select a leaf and replace with a randomly generated new subtree.
- Shrink: randomly select an internal node and replace the subtree below it with a randomly generated terminal node.
- Switch: randomly select an internal node and reorder its argument subtrees.
- Cycle: select a random node and replace it with a new node of the same type. If a terminal node is selected, it is replaced by a terminal node. If an internal node is selected, it is replaced by a function that takes an equivalent number of arguments.

We used standard subtree-swap crossover [2]. The population was initialized by growing trees to a maximum depth of 5, but no further constraint was placed on tree size during evolution, other than the pressure offered by the objective function.

3 Experiments

3.1 GP Parameters, Data Periods and Consistency-of Performance Periods

In all experiments, the GP approach was as described in section 2, with population size 500. In each generation, the current best was copied into the next generation, and the rest were the produced by crossover of two parents (probability 0.7) or mutation of a single parent. Parents were always selected via rank-based selection. Each run continued for 100 generations.

In common with [7] and [19], the period 1960—1991 was generally used for training in the monthly-trading case. In common with [24] we continued to explore two different regimes for choosing and evaluating a rule from the training run. In regime 1, the fittest rule found during training (as measured on the training set) was applied to test data in an immediately succeeding period of N years. In regime two, each rule found during training was validated against the ensuing N year period, and the rule that was best during this validation period was chosen, and tested over a further K years period beyond. These two regimes were each explored for four data period splits:

Table 1. Monthly splits

Name	Training Length	Validation Length (N)	Test Length (K)
MonthlySplit1	31 years	12 years	5 years
MonthlySplit2	31 years	8 years	8 years
MonthlySplit3	31 years	9 years	9 years
MonthlySplit4	25 years	12 years	12 years

For example, when regime 1 was used for data MonthlySplit1, the rule chosen is the best on the 31-year training period, and this rule is evaluated on the subsequent 12 year period. When regime 2 is used for this split, the rule found, while training on the 31-year period, which happened to be best on the subsequent 12 year period, was then evaluated on the further subsequent 5 year period. The results for Monthly splits 1, 2 and 3 were reported in [24], but are also summarized here to aid contrast and comparison with the daily and weekly trading results. Data periods for weekly and for daily trading were chosen to be reasonably consistent with the monthly splits, so that the numbers of days (weeks) involved corresponded with the number of months involved in the monthly splits. The details are as follows:

Table 2. Weekly splits

Name	Training Length	Validation Length (N)	Test Length (K)
WeeklySplit1	366 weeks from 1st Jan 1960	next 158 weeks	next 157 weeks
WeeklySplit2	366 weeks from 1st Jan 1972	next 158 weeks	next 158 weeks
WeeklySplit3	367 weeks from 1st Jan 1984	next 157 weeks	next 158 weeks
WeeklySplit4	366 weeks from 1st Jan 1996	next 157 weeks	next 158 weeks

Table 3. Daily splits

Name	Training Length	Validation Length (*N*)	Test Length (*K*)
DailySplit1	378 days from 1st Jan 1960	next 126 days	next 127 days
DailySplit2	380 days from 1st Jan 1975	next 127 days	next 127 days
DailySplit3	379 days from 1st Jan 1990	next 128 days	next 127 days
DailySplit4	376 days from 1st Jan 2006	next 128 days	next 126 days

In [7,19,21], only regime 1 was used, and MonthlySplit1. In [24] we confirmed that regime 2 gave more robust performance, but we continue to experiment with both regimes here, to see if that conclusion extends to weekly and daily trading. Figure 2 shows the four Monthly data splits aligned against the S&P 500 index for the period 1960—2008. We also show, in Figure 3, a representation of the returns from buy-and-hold for each data split. The market movements were net positive in each part of each split, indicating that outperforming buy-and-hold was in all cases a challenge.

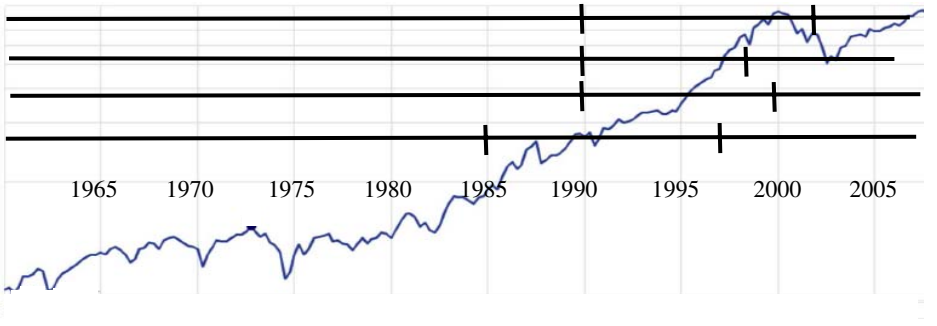


Fig. 2. The S&P500 index over the period 1960—2008, illustrating the four data splits for the case of monthly trading

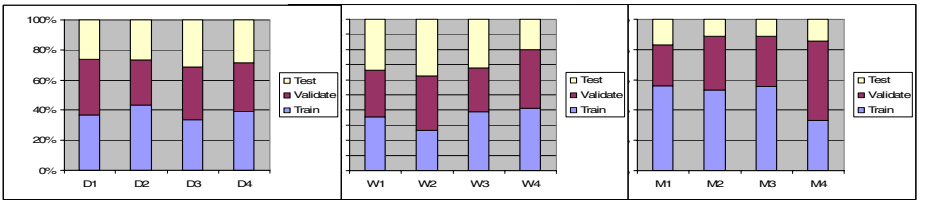


Fig. 3. Characterising the buy-and-hold performance for each data split. Daily splits are on the left, weekly splits in the middle, and monthly splits on the right. Each bar shows relative proportions of the buy-and-hold performance in the training (lower), validation (middle) and test (upper) periods of the data split.

We experimented also with the evaluation periods of the Performance Consistency (PC) term of the fitness function. In Becker and Seshadri's work, PC clearly improves performance (also true in our replication; we omit details for reasons of space). But they only report on 12-month periods. We experiment with different lengths for the "PC period", namely: 6, 12, 18 and 24 months periods for monthly trading; 12 and 24 weeks for weekly trading, and 12 and 24 days for daily trading.

3.2 Results

For each trading period (monthly, weekly, daily), we performed 10 runs each for each combination of data split and consistency of performance period, and we report results for each of regime 1 and regime 2. To save space, we summarize each set of 10 runs in terms of the number of times that the result outperformed buy-and-hold. All results are summarized in Tables 4—6.

Table 4. Summary of results for monthly trading

Data split	PC Period	Eval. regime	Trials outperforming buy-and-hold.	PC Period	Eval. regime	Trials outperforming buy-and-hold.
Monthly Split1	6	1	10 out of 10	18	1	10 out of 10
		2	10 out of 10		2	10 out of 10
Monthly Split2	6	1	5 out of 10	18	1	4 out of 10
		2	9 out of 10		2	10 out of 10
Monthly Split3	6	1	9 out of 10	18	1	7 out of 10
		2	10 out of 10		2	9 out of 10
Monthly Split4	6	1	9 out of 10	18	1	6 out of 10
		2	10 out of 10		2	10 out of 10
Monthly Split1	12	1	10 out of 10	24	1	10 out of 10
		2	10 out of 10		2	10 out of 10
Monthly Split2	12	1	4 out of 10	24	1	4 out of 10
		2	8 out of 10		2	10 out of 10
Monthly Split3	12	1	10 out of 10	24	1	5 out of 10
		2	8 out of 10		2	7 out of 10
Monthly Split4	12	1	9 out of 10	24	1	5 out of 10
		2	10 out of 10		2	10 out of 10

As Table 4 shows, Monthly split 1 was clearly well-disposed to good performance. This split, evaluated with regime 1, was that used in [7] and [19], and perhaps gave an over-optimistic view of the general promise of the method. But it is clear from these results (and from [24]) that performance depends on details of the data split, and also that regime 2 is a better choice. If we now consider Figure 3, in attempt to understand relative performance in terms of the overall market movements in the data splits, we find that this is quite hard to do. Outperforming buy and hold would seem to be more likely when the performance of buy-and-hold in the test period is relatively weak, but this is not the case for Monthly splits 1 and 4. Referring to Figure 2, we see that the market conditions were fairly similar for the training and validation parts of each monthly split, and were `up and down' for each of the four test periods.

Table 5. Summary results for weekly trading

Data split	PC Period	Eval. Regime	Trials outperforming buy-and-hold.	PC Period	Eval. regime	Trials outperforming buy-and-hold.
Weekly Split1	12	1	6 out of 10	24	1	2 out of 10
		2	2 out of 10		2	7 out of 10
Weekly Split2	12	1	10 out of 10	24	1	9 out of 10
		2	10 out of 10		2	5 out of 10
Weekly Split3	12	1	4 out of 10	24	1	3 out of 10
		2	4 out of 10		2	4 out of 10
Weekly Split4	12	1	10 out of 10	24	1	10 out of 10
		2	10 out of 10		2	10 out of 10

Tables 5 and 6 shows the corresponding results for weekly and daily trading respectively. These clearly show increasingly less robust results. It certainly seems that the method can find robust rules for weekly trading that outperform buy-and-hold in some circumstances (splits 2 and 4), with less reliable performance in other cases. However, again, it seems there is no easily spotted pattern that underpins this from the basic summary of the data splits' buy-and-hold performance in Figure 3. For daily trading, outperforming buy-and-hold is less likely, with strong performance in only one of the four data splits, and very poor performance in two of the data splits.

Table 6. Summary of results for daily trading

Data split	PC Period	Eval. regime	Trials outperforming buy-and-hold.	PC Period	Eval. regime	Trials outperforming buy-and-hold.
Daily Split1	12	1	0 out of 10	24	1	0 out of 10
		2	0 out of 10		2	0 out of 10
Daily Split2	12	1	0 out of 10	24	1	0 out of 10
		2	0 out of 10		2	0 out of 10
Daily Split3	12	1	10 out of 10	24	1	10 out of 10
		2	10 out of 10		2	9 out of 10
Daily Split4	12	1	2 out of 10	24	1	3 out of 10
		2	2 out of 10		2	4 out of 10

4 Concluding Summary and Discussion

In most research that uses genetic programming (GP) to induce technical trading rules, the most common outcome tends to be that, despite finding rules that seem successful on their own terms, they are usually not competitive with "buy and hold" (in upwardly moving markets) or the exploitation of risk-free investments (in downward markets). Building on [7], however, [19] was one of few that have shown more promise. This was replicated in [24], with further advice on how reliably to generate effective rules, and also showing that the approaches in [19,21] were rather sensitive to the data splits, and that it is clearly better to use a validation set to choose the trading rule.

However, the above was in the context of monthly trading, one of Becker & Seshadri's changes to the approach in [7], which used daily trading. It is reasonable to

suppose that this might be salient in the ability to beat buy-and-hold. We examined this by testing the approach on each of monthly, weekly and daily trading. We again found fairly robust generation of rules which outperform buy-and-hold for monthly trading, but with such being relatively rare for daily trading, and the situation for weekly trading was inbetween. The approach seems capable of finding rules that outperform buy-and-hold, even when tested in upwardly-moving markets, but performance depends on the data split, and as we move from monthly to daily trading, this dependence on the data split seems to increase sharply. It turns out to be very difficult to pin down the likelihood of success in advance based on simple summary analyses of the data splits. Visual analysis of the charts during the split periods (not shown here for space limitations), and summary measures such as Figure 3, so far fail to yield obvious indicators that might correlate with the possibility of evolving successful rules. This is a topic of continuing research. Finally, we note that interested researchers may pick up our source code at http://www.macs.hw.ac.uk/~dwcorne/gptrcode.

References

[1] Brabazon, A., O'Neill, M.: Biologically Inspired Algorithms for Financial Modelling. Natural Computing Series. Springer, New York (2005)
[2] Koza, J.R.: Genetic Programming: On the Programming of Computers by means of Natural Selection. MIT Press, Cambridge (1992)
[3] Angeline, P.J.: Genetic Programming's Continued Evolution. In: Angeline, P., Kinnear, K. (eds.) Advances in Genetic Programming, vol. 2, pp. 89–110. MIT Press, Cambridge (1996)
[4] Banzhaf, W., Nordin, P., Keller, R.E., Francone, F.D.: Genetic Programming - An Introduction: On the Automatic Evolution of Computer Programs and Its Applications. Morgan Kaufmann, San Francisco (1998)
[5] Chen, S.H., Yeh, C.: Toward a Computable Approach to the Efficient Market Hypothesis: An Application of Genetic Programming. J. Econ. Dyn. & Control 21, 1043–1063 (1996)
[6] Fyfe, C., Marney, J.P., Tarbert, H.: Technical Trading versus Market Efficiency: A Genetic Programming Approach. Applied Financial Economics 9, 183–191 (1999)
[7] Allen, F., Karjalainen, R.: Using genetic algorithms to find technical trading rules. Journal of Financial Economics 51, 245–271 (1999)
[8] Marney, J.P., Fyfe, C., Tarbert, H., Miller, D.: Risk Adjusted Returns to Technical Trading Rules: A Genetic Programming Approach. In: Computing in Economics and Finance, Soc. for Computational Economics, June 2001. Yale University, USA (2001)
[9] Chen, S.H.: Genetic Algorithms and Genetic Programming in Computational Finance. Kluwer, Boston (2002)
[10] Cheng, S.L., Khai, Y.L.: GP-Based Optimisation of Technical Trading Indicators and Profitability in FX Market. In: Proc. 9th ICONIP, vol. 3, pp. 1159–1163 (2002)
[11] Farnsworth, G., Kelly, J., Othling, A., Pryor, R.: Successful Technical Trading Agents Using Genetic Programming, Tech. Rep. SAND2004-4774, Sandia Nat'l Labs (2004)
[12] Potvin, J.Y., Soriano, P., Vallée, M.: Generating Trading Rules on the Stock Markets with Genetic Programming. Computers and Operations Research 31(7), 1033–1047 (2004)
[13] Pring, M.J.: Technical Analysis Explained. McGraw-Hill, New York (1980)
[14] Ruggiero, M.A.: Cybernetic Trading Strategies. Wiley, New York (1997)

[15] Murphy, J.: Technical Analysis of the Financial Markets. New York Inst. Finance, New York (1999)
[16] Lo, A.W., Mamaysky, H., Wang, J.: Foundations of Technical Analysis: Computational Algorithms, Statistical Inference, and Empirical Implementation. J. of Finance 55, 1705–1770 (2000)
[17] Marney, J.P., Miller, D., Fyfe, C., Tarbert, H.: Technical Analysis versus Market Efficiency: A Genetic Programming Approach. In: Computing in Economics and Finance, Society for Computational Economics, Barcelona, Spain (paper #169) (July 2000)
[18] Neely, C.: Risk-adjusted, ex ante, optimal technical trading rules in equity markets. In: Working Papers 99-015D, Revised, Federal Reserve Bank of St. Louis (August 2001)
[19] Becker, L.A., Seshadri, M.: Comprehensibility and Overfitting Avoidance in Genetic Programming for Technical Trading Rules, Worcester Polytechnic Institute, Computer Science Technical Report WPI-CS-TR-03-09 (2003)
[20] Becker, L.A., Seshadri, M.: Cooperative Coevolution of Technical Trading Rules, Worcester Polytechnic Institute, Computer Science Tech. Rep. WPI-CS-TR-03-15 (2003)
[21] Becker, L.A., Seshadri, M.: GP-evolved technical trading rules can outperform buy and hold. In: Proc. 6th Int'l Conf. on Comp. Intell. & Natural Computing (2003)
[22] Sharpe, W.F.: Mutual Fund Performance. Journal of Business 39(S1), 119–138 (1966)
[23] Marney, J.P., Tarbert, H., Fyfe, C.: Risk Adjusted Returns from Technical Trading: A Genetic Programming Approach. Appl. Financial Economics 15, 1073–1077 (2005)
[24] Lohpetch, D., Corne, D.: Discovering Effective Technical Trading Rules with Genetic Programming: Towards Robustly Outperforming Buy-and-Hold. In: Proc. NABIC 2009. IEEE Press, Los Alamitos (2009)

Evolutionary Multi-stage Financial Scenario Tree Generation

Ronald Hochreiter

Department of Finance, Accounting and Statistics
WU Vienna University of Economics and Business
ronald.hochreiter@wu.ac.at

Abstract. Multi-stage financial decision optimization under uncertainty depends on a careful numerical approximation of the underlying stochastic process, which describes the future returns of the selected assets or asset categories. Various approaches towards an optimal generation of discrete-time, discrete-state approximations (represented as scenario trees) have been suggested in the literature. In this paper, a new evolutionary algorithm to create scenario trees for multi-stage financial optimization models will be presented. Numerical results and implementation details conclude the paper.

Keywords: Optimization under uncertainty, scenario generation, scenario optimization, financial decision theory, risk management.

1 Introduction

Stochastic programming is a versatile method to model and solve decision problems under uncertainty. See [1] for an overview of the area of stochastic programming, and [2] for stochastic programming languages, environments, and applications.

We consider the following generalized formulation of a multi-stage stochastic financial optimization model. The decision taker faces a discrete-time decision horizon $t = 1, \ldots, T$, and a set of investment assets (or asset categories) $\mathcal{A}$ with uncertain future returns V_a. These uncertain returns are represented by a stochastic process discretized into a multi-variate, multi-stage scenario tree. This scenario tree is used to build either a deterministic equivalent model formulation, which can be solved using off-the-shelf solvers, or to use a stochastic decomposition algorithm to obtain numerical solutions of the problem. The objective function consists of a risk-return bi-criteria functional, whereby the aim is to maximize the expected wealth and to minimize some risk functional $\mathbb{F}$ of the wealth at the terminal stage T. This resembles the classical Markowitz-style asset allocation [3], see also the multi-stage generalization presented by [4]. The chosen risk factor does not necessarily have to be the variance, e.g. other coherent risk measures as shown by [5] or similar probability-based measures might be better suited for different risk management purposes, and can be integrated into the model. Both dimensions - expectation and risk - are weighted using a

C. Di Chio et al. (Eds.): EvoApplications 2010, Part II, LNCS 6025, pp. 182–191, 2010.

risk-aversion parameter κ, which can be adapted to the needs of the investor and to the current market situation. The main decision is concerned with the amount of budget b_a to be invested into each asset (or asset category) a, as the portfolio is rebalanced at each stage $t = 2, \ldots, T-1$. There is no rebalancing at terminal stage T. Furthermore, additional investment budget B is available at each stage up to $T-1$, which is deterministically determined in advance in this basic model. An important constraint is that the amount of purchases p in each stage cannot exceed the sum of the amount of sales s plus the additional budget available at the respective stage.

Given the above problem specification, we may formulate our multi-stage stochastic programming model as shown in Eq. 1. The numbers in square brackets represent the stage(s) at which the respective constraint is active.

$$\begin{array}{lll} \text{maximize} & \mathbb{E}(\sum_{a\in\mathcal{A}} b_a, T) + \kappa\mathbb{F}(\sum_{a\in\mathcal{A}} b_a, T) & \\ \text{subject to} & \sum_{a\in\mathcal{A}} b_a = B & [1] \\ & b_a \leq V_a b_a^{(-1)} + p_a - s_a & \forall a \in \mathcal{A}\ [2, \ldots, T-1] \\ & \sum_{a\in\mathcal{A}} p_a \leq \sum_a s_a + B & [2, \ldots, T-1] \\ & b_a \leq V_a b_a^{(-1)} & \forall\mathcal{A}\quad [T] \\ & b_a, p_a, s_a \geq 0 & \forall\mathcal{A}\quad [1, \ldots, T] \end{array} \tag{1}$$

The multi-stage recourse decision can be observed in the second and the fourth constraint: V_a represents the future asset return of asset (or asset category) a in the respective stage (on the scenario tree) and is multiplied by the invested budget $b_a^{(-1)}$ of the previous stage.

The parameters which have to be specified by the decision taker are the asset returns V_a, which are stochastic and need to be tree-approximated, as well as the deterministic budget B. The stochastic decision variables, which will be calculated via numerical optimization solver include the current (investment) budget b_a, purchases p_a, as well as sales s_a of each asset a out of the given investment universe $\mathcal{A}$ at each stage t. This model represents the basic building block and can be arbitrarily extended to the needs of the decision taker, e.g. by integrating dynamic risk measures, see e.g. [6].

However, the crucial part of the whole stochastic programming workflow is to generate a multi-stage scenario tree, i.e. the V_a, which represents a careful approximation of the uncertainty of the asset (or asset category) returns, such that a sensible risk management can be based on it.

This paper is organized as follows. Multi-stage scenario tree generation will be briefly sketched in Section 2. A new evolutionary algorithm to create multi-stage scenario trees is presented in Section 3. Section 4 summarizes selected numerical results and the implementation, while Section 5 concludes the paper.

2 Multi-stage Scenario Tree Generation

That scenario tree should represent the uncertain structure of the reality as close as possible, because the quality of the tree severely affects the quality of the solution of the multi-stage stochastic decision model, such that any approximation

scheme should be done in consideration of some optimality criteria, i.e. before a stochastic optimization model is solved, a scenario optimization problem has to be solved independently of the optimization model.

In the context of scenario optimization, optimality can be defined as the minimization of the distance between the original (continuous or highly discrete) stochastic process and the approximated scenario tree. Choosing an appropriate distance may be based on subjective taste, e.g. moment matching as proposed by [7], selected due to theoretical stability considerations (see [8] and [9]), which leads to probability metric minimization problems as shown by [10] and [11], or it may be predetermined by chosen approximation method, e.g. by using different sampling schemes like QMC in [12] or RQMC in [13], see also [14]. It is important to remark that once the appropriate distance has been selected, an appropriate heuristic to approximate the chosen distance has to be applied, which affects the result significantly.

Single-stage scenario generation, i.e. an optimal approximation of a multivariate probability distribution without any tree structure can be done via various sampling as well as clustering techniques. The real algorithmic challenge of multi-stage scenario generation is maintaining a tree structure while still minimizing the overall distance. Only in rare cases, this problem can be solved without the application of heuristics. See [15] for a general overview of algorithmic aspects of multi-stage scenario generation, and [16] for details on financial multi-stage scenario generation.

3 Evolutionary Multi-stage Scenario Tree Generation

The list of successful applications of evolutionary algorithms for solving financial problems is quickly growing, see especially [17], [18], [19], [20], and the references therein. This motivates for creating an evolutionary algorithm for the process of optimal multi-stage stochastic financial scenario generation.

We assume that there is a finite set $\mathcal{S}$ of multi-stage, multi-variate scenario paths, which are sampled using the preferred scenario sampling engine selected by the decision taker. Stages will be denoted by $t = 1, \ldots, T$ where $t = 1$ represents the (deterministic) root stage (root node), and T denotes the terminal stage. Therefore, the input consists of a scenario path matrix of size $|\mathcal{S}| \times (T-1)$. Furthermore, the desired number of nodes of the tree in each stage is required, i.e. a vector n of size $(T-1)$.

For the rest of the paper we will focus on the uni-variate case. However, the extension to the multi-variate case does not pose any structural difficulties besides that a dimension-weighting function for calculating the total distance on which the optimality of the scenario tree approximation is based on has to be defined.

A crucial part in designing a multi-stage scenario tree generator based on evolutionary techniques is finding a scalable genotype representation of a tree - both in terms of the numbers of stages as well as the number of input scenarios. The approach taken in this paper is using a real-valued vector in the range $[0, 1]$

and mapping it to a scenario tree given the respective node format n. The length of the vector is equal to the number of input scenarios $s = |\mathcal{S}|$ plus the number of terminal nodes n_T. Thus, the presented algorithm is somewhat limited by the number of input scenarios. This means that input scenarios should not be a standard set of mindlessly sampled scenario paths, but rather a thoughtfully simulated view on the future uncertainty. This should not be seen as a drawback, as it draws attention to this often neglected part of the decision optimization process.

To map the real-valued vector to a scenario tree, which can be used for a subsequent stochastic optimization, two steps have to be fulfilled. First, the real-valued numbers are mapped to their respective node-set given the structure n of the tree, and secondly, values have to be assigned to the nodes. There are different approaches to determine the center of the node-sets, which also affects the distance calculation, see below for more details.

It should be noted, that a random chromosome does not necessarily lead to a valid tree. This is the case if the number of mapped nodes is lower than the number of nodes necessary given by n_t of the respective stage t. If an uniform random variable generator and a thoughtful node structure is used, which depends on the number of input scenarios, invalid trees should not appear frequently, and can be easily discarded if they do appear.

Consider the following example of the mapping procedure. For demonstration purposes, we only take one stage into account. We do have 10 input scenarios (asset returns), each equipped with the same probability $p = 0.1$, which might be the output of some sophisticated asset price sampling procedure, e.g.

$$(0.017, -0.023, -0.008, -0.022, -0.019, 0.024, 0.016, -0.006, 0.032, -0.023).$$

We want to separate those values optimally into 2 clusters, which then represent our output scenarios and take a random chromosome, which might look as follows:

$$(0.4387, 0.3816, 0.7655, 0.7952, 0.1869, 0.4898, 0.4456, 0.6463, 0.7094, 0.7547)$$

If we map this vector to represent 2 centers we obtain: $(1, 1, 2, 2, 1, 1, 1, 2, 2, 2)$. Now we need to calculate a center value, e.g. the mean, and have to calculate the distance for each value of each cluster to its center, e.g. we obtain center means $(0.0032, -0.0055)$, which represent the resulting scenarios, each with a probability of 0.5. The l_1 distance for each cluster is $(0.0975, 0.0750)$, so the objective function value is 0.1725. Now flip-mutate chromosome 9, i.e. $(1 - 0.7094) = 0.2906$, such that input scenario 9 (return $= 0.032$) will now be part of cluster 1 instead of cluster 2. We obtain new scenarios $(0.0080, -0.0149)$ with probabilities $(0.6, 0.4)$. The objective function value is 0.1475 (or 0.1646 if you weight the distances with the corresponding output scenario probability), i.e. this mutation led to a better objective value.

It should be noted that this mapping is rather trivial in the single-stage case, but this simple approach leads to a powerful method for the tedious task of constructing multi-stage scenario trees for stochastic programming problems,

because the nested probability structure of the stochastic process is implicitly generated.

For multi-stage trees, a crucial point is finding a representative value for the node-sets determined in the first step of the mapping. There exists a range of methods, which can be used for the determination of centers, i.e. mapping all values of a node-sets to one node. The distance of the approximation, which is used for the calculation of the objective function, will also be affected by this method. Some selected methods are summarized below:

- Median. A straight-forward solution is to use the median of the values.
- Extreme. If the mean of the values is below the stage mean, the lowest value of the set will be selected, if it is above the stage mean, the highest value is selected. This can prove useful if one aims at capturing extremes, which already might have been flattened out by the scenario path simulation.
- Mixture. Using the median approach might be smoothing the tail values too much, while the extreme approach neglects normal market phases. To overcome this, a mixture model can be defined, i.e. by splitting the range of stage values into three sections and using the minimum or maximum value of the node-set if the mean of it is in the lowest or highest section, or using the median if the node-set mean lies in the intermediate section.
- Random. A randomly selected value will used. While this method generally leads to balanced results, decision takers might not favor this non-reproducible approach.

To visualize the differences of these approaches, see Fig. 1. The same scenario generation procedure was used both for the left and the right part of the Figure, i.e. the same set of input scenarios, the same evolutionary algorithm parameters, and the same tree structure $n = [10, 40]$. The only difference was choosing either the extreme node-to-value mapping (left) and the median mapping (right). It is clear that these two trees will lead to different decisions. The set of input scenarios will be specified in detail in the next section, see the visualization of the scenario paths in Fig. 2 below.

The evolutionary algorithm chosen is based on the commonly agreed standard as surveyed by [21]. Thereby, the following evolutionary operators have been implemented and used:

- Elitist selection (o_1).
- N-point crossover, with $N = 1$ (o_2) and $N = 2$ (o_3).
- Intermediate crossover with a random intermediate probability, which is different for each chromosome (o_4).
- Mutation/Flip: Invert an initially specified number of m chromosomes by $1 - c$, where c is the current value (o_5).
- Mutation/Random: An initially specified number of m chromosomes will be randomly mutated (o_6).
- Random addition: Randomly sampled chromosomes, also used for creating the initial population (o_7).

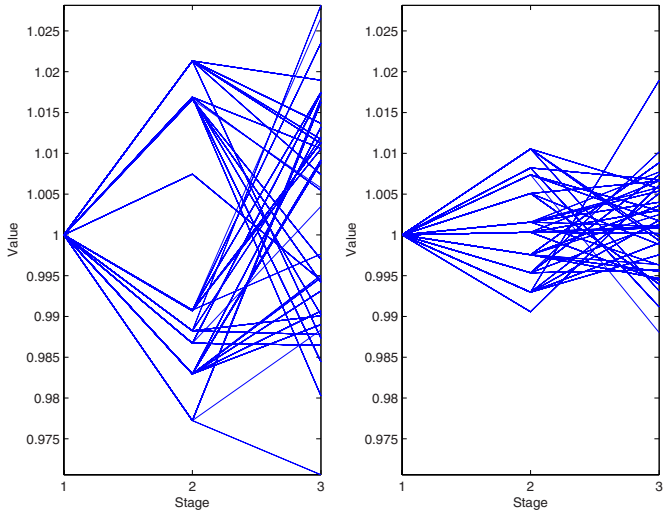


Fig. 1. Difference between Extreme (left) and Median (right) node-value mapping

For each of crossover operator, one parent is taken from the $o_8\%$ best of the previous population, and one entirely randomly. For each mutation operation one of the best $o_9\%$ will be randomly used. These nine values $o_1, \ldots, o_9$ will be used for the description of numerical results and specify the percentage of the given population size, e.g. $(20, 10, 10, 10, 15, 15, 20, 10, 30)$ means that 20% of each new population are created by applying elitist selection and random addition, while 10% of each new population are created by crossovers (1-point crossover, 2-point crossover, intermediate crossover) and 10% by mutations (flip as well as random), where the crossovers are conducted with one parent randomly selected from the top 10% and the other parent randomly selected from the whole previous population and the mutation is executed on one the top 30% chromosomes from the previous population.

4 Numerical Results

The code was implemented using MatLab 2008b without using further toolboxes. Input scenarios have been estimated and simulated using a GARCH(1,1) time series model using historical data from the NASDAQ composite index. The input scenarios are shown in Fig. 2.

The results presented below have been calculated with the following parameters: The initial population consists of 1000 randomly selected chromosomes. The population size during the evolutionary process has been set to 300, and a maximum of 300 iterations is calculated, using the above set of 200 scenarios in

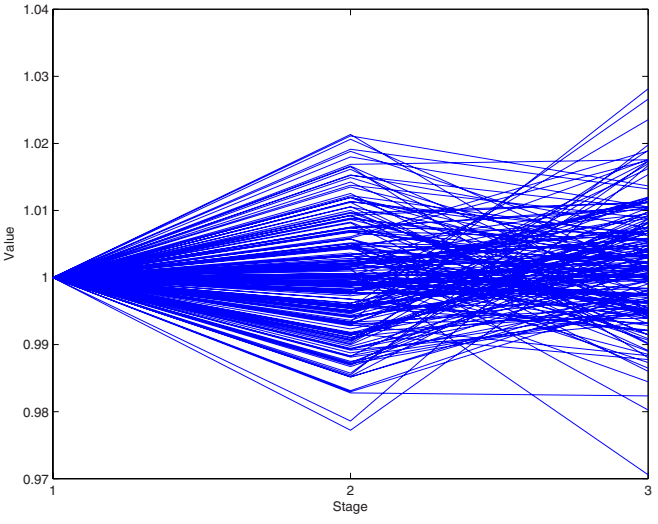


Fig. 2. The set of input scenarios used for numerical results ($s = 200$)

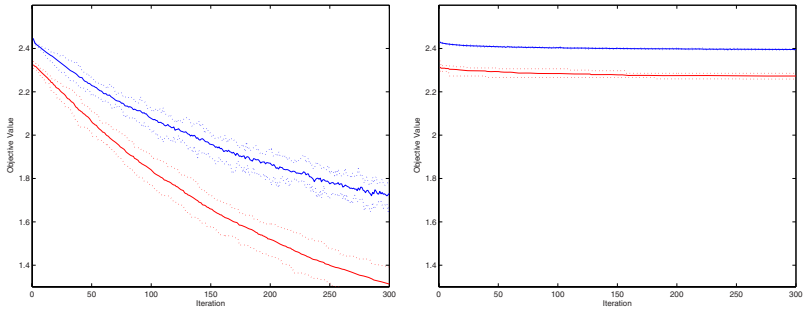


Fig. 3. Convergence of operator structure $(20, 10, 10, 20, 10, 10, 20, 10, 30)$ (left) and $(50, 0, 0, 0, 0, 0, 50, 10, 30)$ (right)

two stages. The mutation parameter m was set to 2. The tree structure has been $[10, 40]$ for all runs and each run takes around $4 - 5$ minutes to solve on an up-to-date desktop computer, which is excellent, when compared to other heuristic global optimization techniques, which often report a scenario generation time of many hours of computation.

The first results show the convergence of different evolutionary operators. We compare four different operator structures:

- Using all operators = $(20, 10, 10, 20, 10, 10, 20, 10, 30)$, see Fig. 3,
- no crossover nor mutation = $(50, 0, 0, 0, 0, 0, 50, 10, 30)$, see Fig. 3,
- no mutation operators = $(20, 20, 20, 30, 0, 0, 10, 10, 30)$ see Fig. 4, and
- no crossover operators = $(30, 0, 0, 0, 30, 30, 10, 10, 30)$ see Fig. 4.

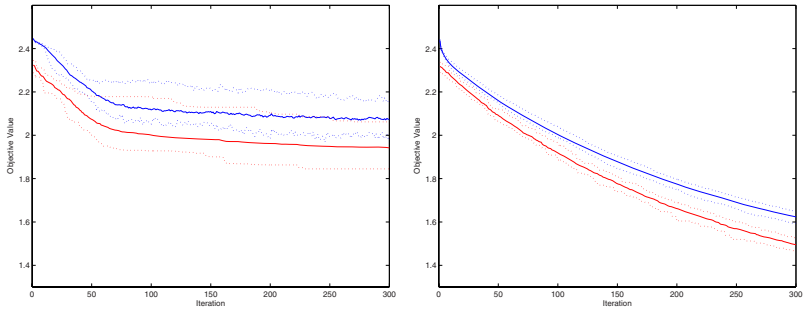


Fig. 4. Convergence of operator structure $(20, 20, 20, 30, 0, 0, 10, 10, 30)$ (left) and $(30, 0, 0, 0, 30, 30, 10, 10, 30)$ (right)

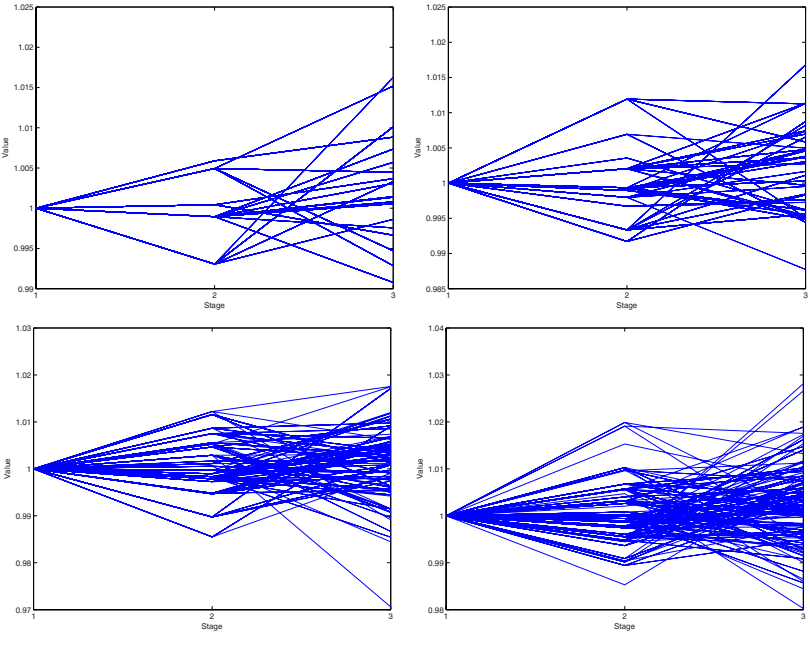


Fig. 5. Scenario trees with $n = [5, 20]$, $[10, 40]$, $[20, 80]$, and $[40, 120]$

The convergence graphs contain the minimum objective function value as well as the population mean. Each test has been repeated 10 times, and the graphs show the mean of the two values, as well as the minimum and maximum per iteration.

In all calculations above, the same tree structure has been used, i.e. $n = [10, 40]$. Of course, the method works for arbitrary scenario trees structures as shown in Fig. 5 for trees with a structure of $n = [5, 20]$, $n = [10, 40]$, $n = [20, 80]$, and $n = [40, 120]$ respectively. It should be noted that for any realistic application

the evolutionary parameters have to be adapted to the specific instance of the scenario tree needed for the given stochastic optimization model.

5 Conclusion

In this paper an evolutionary multi-stage scenario tree generation method has been presented. It could be shown that multi-stage financial scenario generation can be successfully done by applying pure evolutionary optimization techniques. The results motivate for an extension of the implemented code for multi-variate scenario input paths and other features.

References

1. Ruszczyński, A., Shapiro, A. (eds.): Stochastic programming. Handbooks in Operations Research and Management Science, vol. 10. Elsevier Science B.V., Amsterdam (2003)
2. Wallace, S.W., Ziemba, W.T. (eds.): Applications of stochastic programming. MPS/SIAM Series on Optimization, vol. 5. SIAM, Philadelphia (2005)
3. Markowitz, H.M.: Portfolio selection. The Journal of Finance 7(1), 77–91 (1952)
4. Steinbach, M.C.: Markowitz revisited: mean-variance models in financial portfolio analysis. SIAM Review 43(1), 31–85 (2001)
5. Artzner, P., Delbaen, F., Eber, J.M., Heath, D.: Coherent measures of risk. Mathematical Finance 9(3), 203–228 (1999)
6. Eichhorn, A., Römisch, W.: Polyhedral risk measures in stochastic programming. SIAM Journal on Optimization 16(1), 69–95 (2005)
7. Høyland, K., Wallace, S.W.: Generating scenario trees for multistage decision problems. Management Science 47(2), 295–307 (2001)
8. Rachev, S.T., Römisch, W.: Quantitative stability in stochastic programming: the method of probability metrics. Mathematics of Operations Research 27(4), 792–818 (2002)
9. Heitsch, H., Römisch, W., Strugarek, C.: Stability of multistage stochastic programs. SIAM Journal on Optimization 17(2), 511–525 (2006)
10. Pflug, G.C.: Scenario tree generation for multiperiod financial optimization by optimal discretization. Mathematical Programming 89(2, Ser. B), 251–271 (2001)
11. Dupačová, J., Gröwe-Kuska, N., Römisch, W.: Scenario reduction in stochastic programming. An approach using probability metrics. Mathematical Programming 95(3, Ser. A), 493–511 (2003)
12. Pennanen, T., Koivu, M.: Epi-convergent discretizations of stochastic programs via integration quadratures. Numerische Mathematik 100(1), 141–163 (2005)
13. Koivu, M.: Variance reduction in sample approximations of stochastic programs. Mathematical Programming 103(3, Ser. A), 463–485 (2005)
14. Pennanen, T.: Epi-convergent discretizations of multistage stochastic programs via integration quadratures. Mathematical Programming 116(1-2, Ser. B), 461–479 (2009)
15. Hochreiter, R.: Algorithmic aspects of scenario-based multi-stage decision process optimization. In: Rossi, F., Tsoukiàs, A. (eds.) ADT 2009. LNCS, vol. 5783, pp. 365–376. Springer, Heidelberg (2009)

16. Hochreiter, R., Pflug, G.C.: Financial scenario generation for stochastic multi-stage decision processes as facility location problems. Annals of Operations Research 152(1), 257–272 (2007)
17. Brabazon, A., O'Neill, M. (eds.): Natural Computing in Computational Finance. Studies in Computational Intelligence, vol. 100. Springer, Heidelberg (2008)
18. Brabazon, A., O'Neill, M. (eds.): Natural Computing in Computational Finance, vol. 2. Studies in Computational Intelligence, vol. 185. Springer, Heidelberg (2009)
19. Dang, J., Brabazon, A., Edelman, D., O'Neill, M.: An introduction to natural computing in finance. In: Giacobini, M., et al. (eds.) EvoWorkshops 2009. LNCS, vol. 5484, pp. 182–192. Springer, Heidelberg (2009)
20. Brabazon, A., O'Neill, M., Dempsey, I.: An introduction to evolutionary computation in finance. IEEE Computational Intelligence Magazine 3(4), 42–55 (2008)
21. Blum, C., Roli, A.: Metaheuristics in combinatorial optimization: Overview and conceptual comparison. ACM Computing Surveys 35(3), 268–308 (2003)

Evolving Dynamic Trade Execution Strategies Using Grammatical Evolution

Wei Cui[1,2], Anthony Brabazon[1,2], and Michael O'Neill[1,3]

[1] Natural Computing Research and Applications Group
University College Dublin, Ireland
wei.cui.1@ucdconnect.ie, anthony.brabazon@ucd.ie
[2] School of Business, University College Dublin, Ireland
[3] School of Computer Science and Informatics, University College Dublin, Ireland
m.oneill@ucd.ie

Abstract. Although there is a plentiful literature on the use of evolutionary methodologies for the trading of financial assets, little attention has been paid to potential use of these methods for efficient trade execution. Trade execution is concerned with the actual mechanics of buying or selling the desired amount of a financial instrument of interest. Grammatical Evolution (GE) is an evolutionary automatic programming methodology which can be used to evolve rule sets. In this paper we use a GE algorithm to discover dynamic, efficient, trade execution strategies which adapt to changing market conditions. The strategies are tested in an artificial limit order market. GE was found to be able to evolve quality trade execution strategies which are highly competitive with two benchmark trade execution strategies.

1 Introduction

Grammatical Evolution is an Evolutionary Automatic Programming (EAP) technique which allows the generation of computer programs in an arbitrary language. GE can conduct an efficient exploration of a search space, and notably permits the incorporation of existing domain knowledge in order to generate 'solutions' with a desired structure. In finance (for example), this allows the users to seed the evolutionary process with their current trading strategies in order to see what improvements the evolutionary process can uncover. Recently GE has been successfully applied to a number of financial problems. These include financial time series modelling, intraday financial asset trading, corporate credit rating, and the uncovering of technical trading rules [2,16].

Trade execution is the process of trading a particular instrument of interest. A practical issue in trade execution is how to trade a large order as efficiently as possible. For example, trading of a large order in one lot may produce significant market impact costs. Conversely, by dividing an order into smaller lots and spreading these over time, a trader can reduce market impact cost but increases the risk of suffering opportunity cost. An efficient trade execution strategy seeks to balance out these costs in order to minimise the total trade cost. In this paper,

C. Di Chio et al. (Eds.): EvoApplications 2010, Part II, LNCS 6025, pp. 192–201, 2010.

GE is used to discover a dynamic trade execution strategy adapting to changing market conditions, while balancing off market impact cost and opportunity cost.

Instead of optimizing strategies by back-testing them over and over on the same historical data, we novelly test them in an artificial limit order market. One advantage of doing this is that the strategies can interact with the changing market. An agent-based modelling approach is adopted to simulate the artificial limit order market.

This paper is organized as follows. The next section provides a brief synopsis of the typical operation of an electronic double auction marketplace; Section 3 discusses trade execution strategies using GE and describes our performance evaluation approach; Section 4 explains agent-based modeling and describes how we implement the artificial stock market used in this study; Section 5 provides our results, with conclusion and some future work being presented in the final section of this paper.

2 Background

Today most market places operate an electronic double auction *limit order book*. Traders can either submit *a limit order* or *a market order*. A market order is an order to buy or to sell a specified number of shares. It guarantees immediate execution but provides no control on its execution price. In contrast, a limit order is an order to buy or to sell a specified number of shares at a specified price. It provides control over its execution price but does not guarantee its execution.

Table 1. Order Book 1

Bid		Ask	
Shares	Prices	Prices	Shares
300	50.19	50.22	200
200	50.18	50.23	300
400	50.17	50.24	100
500	50.16	50.25	300
300	50.15	50.26	200
100	50.14	50.27	400

Table 2. Order Book 2

Bid		Ask	
Shares	Prices	Prices	Shares
300	50.19	50.22	**200**
500	50.18	50.23	300
400	50.17	50.24	100
500	50.16	50.25	300
300	50.15	50.26	200
100	50.14	50.27	400

Table 3. Order Book 3

Bid		Ask	
Shares	Prices	Prices	Shares
300	50.19	50.22	**100**
500	50.18	50.23	300
400	50.17	50.24	100
500	50.16	50.25	300
300	50.15	50.26	200
100	50.14	50.27	400

Table 1 shows a sample order book, where all the buy and sell orders are visible to traders in the market. It consists of two queues which store buy and sell limit orders, respectively. Buy limit orders are called *bids*, and sell limit orders are called *offers* or *asks*. The highest bid price on the order book is called *best bid*, and the lowest ask price on the order book is called *best ask*. The difference between best bid and best ask is called *bid-ask spread*. Prices on the order book are not continuous, but rather change in discrete quanta called *ticks*.

Limit orders on the order book are typically (depending on market rules) executed strictly according to (1) price priority and (2) time priority. Bid (ask) orders with higher (lower) prices get executed first with time of placement being used to break ties. A buy (sell) market order is executed at the best ask (bid) price. The limit order book is highly dynamic, because new limit orders will be

added into the order book, and current limit orders will get executed or cancelled from the order book throughout the trading day. Table 2 shows the order book after a trader submits a buy limit order with 300 shares placed at price 50.18. Table 3 shows the order book after a trader submits a buy market order with 100 shares.

3 Evolving Dynamic Trade Execution Strategies

A trade execution strategy is a set of rules determining a number of trade execution components designed to minimize transaction cost. These components include number of orders to be submitted, size of each order, what type each order should be and when each order should be submitted to the market.

The total trading volume of the order to be traded is often expressed as a percentage of the *average daily volume* (ADV) of the stock [11]. An order of less than 5% of ADV can generally be traded over a day without using complex strategies. On the contrary, if the target volume is larger than 15% of ADV, it may require execution over several days in order to minimize market impact. Normally, 5-15% of ADV is a reasonable order size which could expect to be tradable over a day using appropriate trade execution tactics. In this paper, the trading horizon of all strategies is one trading day and hence we assume that the order size is of this magnitude.

We assume that the order to be traded consists of V shares. The order is sliced into N smaller child orders (each of which will be submitted to the market according to our trading strategy), with order size s_1, s_2, ..., s_N, where

$$V = \sum_{i=1}^{N} s_i$$

A time window of half an hour is adopted in evolving our trading strategies. We benchmarked the results from our evolved trading strategies against two simple execution strategies. One simple trade execution strategy is a pure market order strategy in which each child order is submitted as a market order every half hour. This strategy takes market liquidity immediately by crossing the bid-ask spread. The other benchmark trade execution strategy is a pure limit order strategy. Traders submit each child order as a limit order placed at the best price, and amend its price to best price at a fixed frequency until this order is fully executed or until the trading period expires. At the end of trading day, any unexecuted orders are traded by crossing the bid-ask spread in order to ensure order completion. For instance, a buy order s_n may be submitted to the market as a limit order placed at the best bid price with an amendment frequency of Δt minutes. If Δt minutes after submission, this limit order is not fully executed, it will be amended to the best bid price. This amendment process continues in Δt intervals up to the end of trading day, at which time the uncompleted order(s) are traded as market orders by crossing the bid-ask spread.

In the simple market order strategy, *order aggression* (crossing the bid-ask spread) happens immediately after order submission which guarantees order

execution, at the cost of market impact. In the simple limit order strategy, order aggression happens at the end of trading period aiming to reduce market impact, at the risk of opportunity cost. A more sophisticated limit order strategy would allow for order aggression between these two extreme cases. A general limit order strategy is to cross the uncompleted limit order over the spread after submission but before the end of trading day. In our GE evolved strategies, the timing of order aggression is determined by an execution rule evolved using GE. At each amendment time (an integral multiple of Δt minutes after submission), if the market condition satisfies the condition of the execution rule, order aggression happens, otherwise, the uncompleted order is amended to the best price. In this paper, an amendment frequency of 10 minutes is adopted in all limit order strategies. The market variables representing the market condition are examined in the next section.

3.1 Information Indicators

There are a large number of studies in the literature analyzing the relationship between order placement and the information content of limit order books.

Table 4. Definitions of Market Variables

Variables	Definitions
BidDepth	Number of shares at the best bid
AskDepth	Number of shares at the best ask
RelativeDepth	Total number of shares at the best five ask prices divided by total number of shares at the best five bid and ask prices
Spread	Difference between the best bid price and best ask price
Volatility	Standard deviation of the most recent 20 mid-quotes
PriceChange	Number of positive price changes within the past ten minutes divided by the total number of quotes submitted within the past ten minutes

Traders are more willing to place market orders when the market depth on the same side of the order book is large. If the market depth on the opposite side is larger, traders prefer to submit limit orders [3,6,18,23]. The incoming limit orders will have lower execution probability, suffering higher non-execution risk. When the bid-ask spread widens, traders prefer to submit limit orders in order to avoid large bid-ask spread cost [3,6,17,18,22,23]. Prior research is inconclusive on the effect of market volatility. Pascual and Verdas [17] show that higher historic volatility suggests limit order submission in mid cap stocks, but the opposite phenomenon is observed in large cap stocks. Hall and Hautsch [10] observe an increase of all kinds of order submission during periods of high volatility. Ranaldo [18] supports an inverse relation between order aggression and volatility, while Lo and Sapp [14] report a positive relationship between order aggression and volatility. Cao et al. [3] find that volatility has a minimal effect on order aggression. Verhoeven [22] argues that greater price volatility implies that a trader has a greater chance of executing his order at a better price. Hence, prior literature suggests a range of possible explanatory variables, but indicates that we have an incomplete theoretical understanding of how these factors interact. This suggests that there will be particular utility for the application of

evolutionary methods to uncover a suitable model structure (trade execution strategy). Based on the explanatory factors considered in the literature, we selected six information indicators to construct a dynamic trade execution strategy (Table 4).

3.2 Grammar of Grammatical Evolution Algorithm

The grammar adopted in our experiments is defined as follows:

```
<lc> ::= if (<stamt>)
                class = "CrossingSpread"
         else
                class = "NotCrossingSpread"
<stamt> ::= <cond1><op><cond2><op><cond3><op><cond4>
            <op><cond5><op><cond6>
<op> ::= and | or
<cond1> ::= (BidDepth>AvgBidDepth) is <boolean>
<cond2> ::= (AskDepth>AvgAskDepth) is <boolean>
<cond3> ::= (RelativeDepth>AvgRelativeDepth) is <boolean>
<cond4> ::= (Spread>AvgSpread) is <boolean>
<cond5> ::= (Volatility>AvgVolatility) is <boolean>
<cond6> ::= (PriceChange>AvgPriceChange) is <boolean>
<boolean> ::= True | False
```

In the grammar, $AvgBidDepth$ represents the average bid depth of the market, $AvgAskDepth$ represents the average ask depth of the market, $AvgRelativeDepth$ represents the average relative depth of the market, $AvgSpread$ represents the average spread of the market, $AvgVolatility$ represents the average volatility of the market and $AvgPriceChange$ represents the average price change of the market. The six financial variables are observed at the time of order amendment. An example of an evolved dynamic strategy using three financial variables is as follows.

```
if ( (BidDepth>AvgBidDepth) is True  or  (AskDepth>AvgAskDepth) is False
      and  (Spread>AvgSpread) is True  )   class = "CrossingSpread"
else        class = "NotCrossingSpread"
```

In this strategy, if the market condition satisfies

```
(BidDepth>AvgBidDepth) is True and (Spread>AvgSpread) is True
```

or satisfies

```
(AskDepth>AvgAskDepth) is False and (Spread>AvgSpread) is True
```

the uncompleted limit order will be crossed over the bid-ask spread. Otherwise, its limit price will be amended to the best price.

3.3 Performance Evaluation

The standard industry metric for measuring trade execution performance is the *VWAP measure*, short for *Volume Weighted Average Price*. It is calculated as the ratio of the value traded and the volume traded within a specified time horizon

$$VWAP = \frac{\sum(Volume * Price)}{\sum(Volume)}$$

where $Volume$ represents each traded volume and $Price$ represents its corresponding traded price. An example is shown in Figure 1.

	Submission Time	Shares		Traded Price		Value
Child Order 1:	t_0	400	*	50.15	=	20,060
		600	*	50.16	=	30,096
Child Order 2:	$t_1(t_0 + \Delta t)$	1,000	*	50.40	=	50,400
Child Order 3:	$t_2(t_0 + 2\Delta t)$	200	*	50.34	=	10,068
		800	*	50.36	=	40,288
Child Order 4:	$t_3(t_0 + 3\Delta t)$	1,000	*	50.39	=	50,390
Child Order 5:	$t_4(t_0 + 4\Delta t)$	1,000	*	50.68	=	50,680
Child Order 6:	$t_5(t_0 + 5\Delta t)$	1,000	*	51.10	=	51,100
Child Order 7:	$t_6(t_0 + 6\Delta t)$	1,000	*	50.87	=	50,870
Child Order 8:	$t_7(t_0 + 7\Delta t)$	700	*	50.98	=	35,686
		300	*	51.00	=	15,300
Child Order 9:	$t_8(t_0 + 8\Delta t)$	1,000	*	50.39	=	50,390
Child Order 10:	$t_9(t_0 + 9\Delta t)$	1,000	*	50.26	=	50,260
Total:		10,000				505,588
VWAP $= 505,588/10,000 = 50.5588$						

Fig. 1. VWAP Calculation of A Sample Buy Strategy

In order to evaluate the performance of a trade execution strategy, its VWAP is compared against the VWAP of the overall market. The rationale here is that performance of a trade execution strategy is considered good if the VWAP of the strategy is more favorable than the VWAP of the market within the trading period and bad if the VWAP of the strategy is less favorable than the VWAP of the market within the trading period. For example, if the VWAP of a buy strategy ($VWAP_{strategy}$) is lower than the market VWAP ($VWAP_{market}$), it is considered as a good trade execution strategy. Conversely, if the $VWAP_{strategy}$ is higher than the $VWAP_{market}$, it is considered as a bad trade execution strategy. Although this is a simple metric, it largely filters out the effects of volatility, which composes market impact and price momentum during the trading period [1]. The performance evaluation functions for each trading day are as follows:

$$VWAP\ Ratio = \begin{cases} \frac{10{,}000*(VWAP_{strategy} - VWAP_{market})}{VWAP_{market}} & \text{Buy Strategy} \\ \frac{10{,}000*(VWAP_{market} - VWAP_{strategy})}{VWAP_{market}} & \text{Sell Strategy} \end{cases}$$

where $VWAP_{market}$ is the average execution price which takes into account all the trades over the day excluding the strategy's trades. This corrects for bias, especially if the order is a large fraction of the daily volume [13]. For both buy and sell strategies, the smaller the VWAP Ratio, the better the strategy is.

4 Simulating an Artificial Market

In our experiments, the training and evaluation of all trade execution strategies are implemented in an artificial limit order market, which is simulated using an agent-based model.

Agent-based modelling is a computerized simulation consisting of a number of agents. The emergent properties of an agent-based model are the results of "bottom-up" processes, where the decisions of individual and interacting agent at a microscopic level determines the macroscopic behavior of the system. For a more detailed description of agent-based modelling in finance, please refer to [12,19,20]. In this paper, our agent-based artificial limit order market is built based on the *Zero-Intelligence* (ZI) model [5] with a continuous double auction price formation mechanism. The notion of ZI agents was first mentioned in Gode and Sunder [9]. These agents randomly generate buy and sell orders. The orders are then submitted to a market agent, who manages all incoming orders according to the order matching mechanism in a real limit order market. The trading process is continuous, where unmatched orders are stored in an order book.

At each time step, an agent is equally likely to generate a buy order or a sell order. This order can be a market order, or a limit order, or a cancellation of a previous order, with probabilities λ_m, λ_l, and λ_c respectively. The sum of these probabilities is one ($\lambda_m + \lambda_l + \lambda_c = 1$). For a limit buy (sell) order, it has a probability of $\lambda_{inSpread}$ falling inside the bid-ask spread, a probability of λ_{atBest} falling at the best bid (ask) price, and a probability of λ_{inBook} falling off the best bid (ask) price in the book, ($\lambda_{inSpread} + \lambda_{atBest} + \lambda_{inBook} = 1$). The limit price inside the spread follows a uniform distribution. The limit price off the best bid (ask) price follows a power law distribution with the exponent of $(1 + \mu_1)$. The log order size of a market order follows a power law distribution with the exponent of $(1 + \mu_2)$, while the log order size of a limit order follows a power law distribution with the exponent of $(1 + \mu_3)$.

As each incoming buy (sell) market order arrives, the market agent will match it with the best ask (bid) limit order stored in the order book. If this market order is fully filled by the first limit order, the unfilled part will be matched to the next best ask (bid) limit order until it is fully filled. As each incoming limit order arrives, the market agent will store it in the order book according to price

Table 5. Initial Parameters for Artificial Limit Order Market

Explanation	Value
Initial Price	$price^0 = 50$
Tick Price	$\delta = 0.01$
Probability of Order Cancellation	$\lambda_c = 0.34$
Probability of Market Order	$\lambda_m = 0.16$
Probability of Limit Order	$\lambda_l = 0.50$
Probability of Limit Order in Spread	$\lambda_{inSpread} = 0.32$
Probability of Limit Order at Best Quote	$\lambda_{atBest} = 0.33$
Probability of Limit Order off the Best Quote	$\lambda_{inBook} = 0.35$
Limit Price Power Law Exponent	$1 + \mu_1 = 2.5$
Market Order Size Power Law Exponent	$1 + \mu_2 = 2.7$
Limit Order Size Power Law Exponent	$1 + \mu_3 = 2.1$

and time priority. As each incoming cancelation order arrives, the market agent will delete the relevant limit order in the order book.

In order to ensure that the order flows generated by the artificial market are economically plausible, all the parameters in our model are derived from empirical evidence [4,7,8,15,21]. The parameters used in our simulation are presented in Table 5.

5 Results

In this study we consider a large order of 10% of ADV of the artificial market, which is to be traded over one day (5 hours in the artificial market). This order is equally divided into ten child orders. In all trade execution strategies, any uncompleted orders are crossed over the spread at the end of trading day in order to ensure order completion.

Our experiments comprise of two periods (training and test periods). In the training period, GE is used to evolve dynamic trade execution strategies. Each individual is exposed to 20 continuous trading days in the artificial market and their fitness is calculated as their average VWAP ratio over the 20 trading days. The GE experiment is run for 20 generations, with variable-length, one-point crossover at a probability of 0.9, one point bit mutation at a probability of 0.01, roulette selection, steady-state replacement and a population size of 100. In the test period, the best evolved strategy in the training period is tested out of sample over 240 days in the artificial market. The performances of simple market order strategies (SM) and simple limit order strategies (SL) are also evaluated in order to benchmark the GE strategies.

Table 6. Results of best evolved GE strategies and two benchmark strategies

	SM	**SL**	**GE**		
	Mean (S.D.)	Mean (S.D.)	Mean (S.D.)	H_1	H_2
Buy Order	69.64 (0.42%)	42.54 (1.45%)	-1.42 (0.49%)	0.00	0.01
Sell Order	68.73 (0.36%)	13.81 (1.59%)	-23.21 (0.48%)	0.00	0.01

The results (all out of sample) of buy strategies and sell strategies are provided in Table 6. The "Mean" is the average VWAP ratio of each strategy over the 240 days, and "S.D." represents the standard deviation of the average (daily) VWAP ratio. P-values for the null hypothesis $H_1 : mean_{SM} \leq mean_{GE}$ and $H_2 : mean_{SL} \leq mean_{GE}$ are also shown in the table, to indicate the degree of statistical significance of the performance improvement of GE strategies over the two simple strategies. The figures show that the null hypotheses are rejected at the ≤ 0.01 level.

Based on the results, GE evolved strategies notably outperform the two benchmark strategies, simple market order strategy (SM) and simple limit order strategy (SL). The negative VWAP ratios of -1.42 and -23.21 show that the GE

evolved strategies achieve better execution price than the average execution price of the market. The small standard deviations of 0.49 and 0.48 suggest that the applicability of GE for evolving quality dynamic trade execution strategies. Comparing the performance of the strategies for buy and sell orders, we observe that the performances of sell strategies are better than those of buy strategies.

6 Conclusions and Future Work

Trade execution is concerned with the actual mechanics of trading an order. Traders wishing to trade large orders face tradeoffs in balancing market impact and opportunity costs. Trade execution strategies are designed to balance out these costs, thereby minimizing transaction cost relative to some benchmark like VWAP. Despite the importance of optimising trade execution, there has been relatively little attention paid in the literature to the application of evolutionary methods for this task. In this paper, GE was novelly applied for the purposes of evolving dynamic trade execution strategies, and an artificial limit order market was simulated for testing the evolved trade execution strategies. GE was found to be able to evolve quality trade execution strategies which proved highly competitive against two benchmark trade execution strategies.

There is notable scope for further research utilising GE in this problem domain. One obvious route is to widen the number of market variables which can be included in the evolved execution strategies. Another route is to evolve the full structure of the trade execution strategy. In our approach, we focused on one aspect of trade execution strategy (when to cross the spread), and other components like the number of orders are determined in advance. Future work will embrace the evolution of the full structure of trade execution strategy.

Acknowledgement

This publication has emanated from research conducted with the financial support of Science Foundation Ireland under Grant Number 08/SRC/FM1389.

References

1. Almgren, R.: Execution costs. In: Encyclopedia of Quantitative Finance, pp. 346–357. Wiley, Chichester (2008)
2. Brabazon, A., O'Neill, M.: Biologically Inspired Algorithms for Financial Modeling. Springer, Heidelberg (2006)
3. Cao, C., Hansch, O., Wang, X.: Order placement strategies in a pure limit order book market. The Journal of Financial Research 26, 113–140 (2008)
4. Chakraborti, A., Toke, I., Patriarca, M., Abergel, F.: Econophysics: Empirical facts and agent-based models (2009), http://arxiv.org/pdf/0909.1974
5. Daniel, G.: Asynchronous simulations of a limit order book. PhD thesis, University of Manchester, UK (2006)

6. Duong, H., Kalev, P., Krishnamurti, C.: Order aggressiveness of institutional and individual investors. Pacific-Basin Finance Journal 1, 1–14 (2009)
7. Farmer, D., Gillemot, L., Iori, G., Krishnamurthy, S., Smith, E., Daniels, M.: A random order placement model of price formation in the continuous double auction. In: Blume, L., Durlauf, S. (eds.) The economy as an evolving complex system III. Oxford University Press, Oxford (2005)
8. Farmer, J.D., Patelli, P., Zovko, I.: The predictive power of zero intelligence models in financial markets. Proceedings of the National Academy of Sciences of the United States of America 102, 2254–2259 (2005)
9. Gode, D., Sunder, S.: Allocative efficiency of markets with zero-intelligence traders. Journal of political economy 101, 119–137 (1993)
10. Hall, A., Hautsch, N.: Order aggressiveness and order book dynamics. Empirical Economics 30, 973–1005 (2006)
11. Kissell, R., Glantz, M.: Optimal Trading Strategies, Amacom, USA (2003)
12. LeBaron, B.: Agent-based computational finance. In: Tesfatsion, L., Judd, K. (eds.) Handbook of Computational Economics. Agent-based Computational Economics, vol. 2, pp. 134–151. Elsevier, Amsterdam (2005)
13. Lim, M., Coggins, R.: Optimal trade execution: an evolutionary approach. Proceedings of IEEE Congress on Evolutionary Computation 2, 1045–1052 (2005)
14. Lo, I., Sapp, S.: Order aggressiveness and quantity: How are they determined in a limit order market (2007), http://test.bank-banque-canada.ca/fr/res/wp/2007/wp07-23.pdf
15. Mike, S., Farmer, J.: An empirical behavioral model of liquidity and volatility. Journal of Economic Dynamics and Control 32, 200–234 (2008)
16. O'Neill, M., Ryan, C.: Grammatical Evolution. Kluwer Academic Publishers, USA (2003)
17. Pascual, R., Verdas, D.: What pieces of limit order book information matter in explaining order choice by patient and impatient traders. Quantitative Finance 9, 527–545 (2009)
18. Ranaldo, A.: Order aggressiveness in limit order book markets. Journal of Financial Markets 7, 53–74 (2004)
19. Samanidou, E., Zschischang, E., Stauffer, D., Lux, T.: Agent-based models of financial markets. Reports on Progress in Physics 70, 409–450 (2007)
20. Tesfatsion, L.: Agent-based computational economics: A constructive approach to economic theory. In: Tesfatsion, L., Judd, K. (eds.) Handbook of computational economics: agent-based computaional economics, pp. 52–74. Elsevier, Amsterdam (2006)
21. Toth, B., Kertesz, J., Farmer, J.: Studies of the limit order book around large price changes. European Physical Journal B 71, 499–510 (2009)
22. Verhoeven, P., Ching, S., Ng, H.: Determinants of the decision to submit market or limit orders on the ASX. Pacific-Basin Finance Journal 12, 1–18 (2004)
23. Xu, Y.: Order aggressiveness on the ASX market. International Journal of Economics and Finance 1, 51–75 (2009)

Modesty Is the Best Policy: Automatic Discovery of Viable Forecasting Goals in Financial Data

Fiacc Larkin and Conor Ryan

Biocomputing and Developmental Systems
University of Limerick, Ireland
fiacclarkin@gmail.com, conor.ryan@ul.ie

Abstract. This paper presents a new approach to financial forecasting, inspired by strategies used by market traders. We demonstrate that a trading system with the relatively modest task of spotting *trends in progress* rather than the usual goal of spotting *peaks and troughs* can produce highly accurate forecasts. This is achieved by using a Genetic Algorithm to select appropriate training cases which are then fed to a trading system composed of multiple GP derived trees.

1 Introduction

Market movements are notoriously difficult to forecast with any accuracy. If we accept the widely held view that markets are by and large *efficient* [6], with scattered pockets of predictable inefficiency, then we must concede that many of the movements (as observed through price alone) appear to happen for arbitrary reasons and bear no relation to the prior price action. Under these circumstances, attempting to forecast every movement found in a market time series is not only impossible but also detrimental to the chances of predicting the subset of movements that really are foreseeable.

Insight can be extracted from the trader's adage that most successful positions capture only the middle 50% of a market trend [12], see figure 1 for an example of such a situation, this indicates that the professionals themselves consider absolute turning points in the market to be difficult if not impossible to foresee. Despite this, traders still make consistent profits by taking advantage of the market's tendency to perpetuate a trend once it has begun. Any system that is designed to forecast absolute peaks and troughs may result in poor performance, not because of its competence as a methodology, but because its stated objective (forecasting absolute turning points) is inherently impossible. As the trader's adage points out: a system with the less ambitious objective of spotting a trend once it has already begun will be more profitable in practice than the more ambitious system. This paper experimentally demonstrates an approach that embodies the wisdom of the trader's adage. It is shown that the best opportunity of success is given to the underlying forecast methodology by selectively weeding out the unpredictable training data through automatic evolution. A **Hybrid Forecasting System** (HFS) is used to implement this notion of **selective case training**.

C. Di Chio et al. (Eds.): EvoApplications 2010, Part II, LNCS 6025, pp. 202–211, 2010.

Fig. 1. Two hypothetical entry trades (A,B) and two exits (C,D). It is common to attempt to forecast the absolute peak and trough A to D, and, although this appears to make financial sense, there may be no indication of change prior to these points, making them inherently unpredictable. It is likely that B and C may be foreseen with much greater accuracy.

The HFS is a novel combination of GA and GP. A first phase uses a GA to identify a subset of price moves. These moves are then used as the training examples for a series of GP runs in a classification context. Thus, the master system (the GA) is responsible for finding the *best way to teach* the slave (GP) system, where *best way to teach* means finding the optimal subsets of training cases from a financial time series. The optimal subset will give the slave algorithm the best opportunity to learn how to forecast, free from much of the arbitrary noise so prevalently seen in the markets.

The paper is organized as follows. Section 2 gives discusses related work, while section 3 elaborates on the HFS. Next, section 4 discusses the experimental procedure used to test the HFS before section 5 gives the experimental results. Finally, section 6 gives some concluding remarks.

2 Background

In the Evolutionary Computation literature, *market forecasting* is traditionally broken down into several categories. **Portfolio Optimization** [1],[7] seeks to find the optimal *basket* of stocks at any given time whereby a number of constraints (investment diversification for example) are satisfied. **Symbolic Regression** [10], [9] is the search for polynomial equations that most optimally fits a given piece of financial data. The hope is that some underlying process in the market can be distilled out and used to forecast beyond the known horizon. **Strategy Creation** [14],[8] involves finding successful combinations of technical analysis trading rules, while **Trade Execution** algorithms [3],[13] seek the optimal implementation of a market order given the dynamic reaction of other market participants (which usually erode a trade's potential for profits). Trade Execution Algorithms, along with the recent trend of ultra low latency links to the exchanges, are the key weapons being wielded in a technological arms race that has emerged between the large financial institutes in recent years.

Regularly, there is only one data source used in any market forecasting endeavor, *price action*, these are time series of the fluctuating value of a financial instrument as determined by the market participants. Notable exceptions in GP include expanding the input set to include *analyst ratings* [2] and *news sentiment* [11]. These additional inputs give the aforementioned systems a sense of the prevalent subjective opinions regarding the target instrument.

To our knowledge, no EC system explicitly attempts to shield the underlying forecasting methodology from inherently unpredictable market moves during the training process.

3 The Hybrid Forecasting System

The HFS is essentially a GA with a meta-GP. There is one master GA run but every time a GA genome has to be evaluated, it requires seperate GP runs to do so. Each GA genome contains a bit string which has a single binary value corresponding to every time point along the *training* segment of data. The bit string is used to identify which *training cases* will be used to train this genome's classifiers.

Next, GP runs are used to evolve classifiers that correctly distinguish these cases from all others in the training data. The performance scores of these classifiers serve as an approximation for how good the set of training cases was for learning.

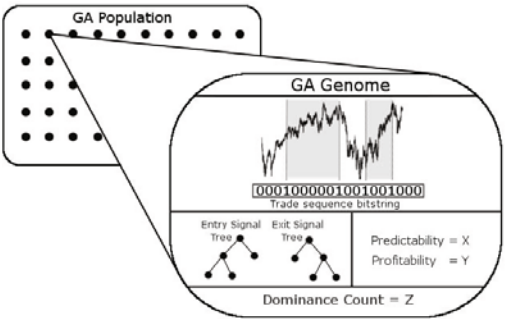


Fig. 2. The contents of each genome in the GA population

There are two classifiers associated with each GA genome; one for position entries (forecasting future rises in the data) and one for exits (forecasting future falls). Both classifiers take the form of a **Signal Generation Tree** and both are found with a separate GP run. Together, the **Entry tree** and **Exit tree**, pair to make a coherent *trading strategy*.

Once a strategy has been found its performance must be evaluated. Trading strategies have dual-objectives: **Predictability** and **Profitability**. A strategy's **Predictability** is a metric quantifying its classification accuracy on the

training data, as selected by the GA bit string. The **Profitability** is a metric quantifying its cumulative profit when applied to the **testing** data, completely independent of the GA bit string. Finally, the Pareto **Dominance Count** for the GA genome can be calculated by comparing these dual objectives against every other solution in the population. Figure 2 shows the hierarchical layout of the algorithm.

Representation. The classification goal is to get the correct answer for the following two questions at every time point t along the data:

- Given the data prior to t, will the market subsequently rise in value?
- Given the data prior to t, will the market subsequently fall in value?

Every time point t is considered a separate **case** that has a set of data inputs and two target answers. The HFS attempts to find the optimal pair of signal trees (the Entry Tree and Exit Tree) to address these questions.

3.1 Fitness Evaluation

As mentioned above, to assess the abilities of each GA genome, three metrics must be calculated; these are Predictability, Profitability and the Dominance Count. The steps required to calculate each metric are detailed below.

Calculating the Predictability for each Genome. Ideally, after being trained, the two signal generation trees should be able to reproduce the exact bit string they were trained on when presented with the raw training data. That is, the Entry Tree would indicate that the system should enter the market at each entry point marked by the GA genome, while the Exit Tree would indicate the system should exit the market at its corresponding points.

The interpretation of the **On** bits found in a genome cycle alternately between Extry Trade and Exit Trade. All the Entry Trades serve as **Positive** examples for training the entry tree and similarly the **Positive** examples are gathered to train the exit tree. It is then necessary to harvest a corresponding number of **Negative** cases for both trees. Once all example cases have been found a seperate GP run is used to evolve each tree. The standard classification measures of **Sensitivity** and **Specificity** are used to gauge how well the trees can distinguish their respective cases, and the average combined Sensitivity and Specificity for the pair of trees is used as the **Predictability** score.

Calculating Profitability for each Genome. To calculate **Profitability** we use the second block of data which remains unseen, this is known as the *testing segment*. Using separate segments in this manner promotes generalization of the models and avoids over-fitting to the training data. Both signal generation trees are applied against every case found in the testing segment. There are no prescribed right or wrong answers this time. Instead a financial tally is kept as the two trees dictate trading.

If the entry tree flags a positive signal at time t we enter a long position at the current market price $P(t)$. Once engaged in a position, further signals from the entry tree are ignored and attention is switched to the exit tree. A positive signal from the exit tree x time steps later causes the close of that position at price $P(t+x)$ for a gain of $P(t+x) - P(t)$. The exit tree is now ignored and the cycle repeats until every case in the data has been applied and the cumulative returns calculated. The final balance at the end of the testing period becomes the **Profitability** score.

Calculating the Dominance Count for each Genome. With Predictability and Profitability calculated for every solution it is now possible to compare all solutions and find the few that are undominated in terms of Pareto optimality, that is those solutions who are as good as, if not better than all other solution in at least one of their fitness traits. In keeping with the NSGA-II [5] multi-objective approach, we take the overall fitness of a genome to be that of its Dominance Count. A lower dominance count is better, with zero meaning that no other solution is better than this one in both predictability and profitability. Our GA algorithm can now be thought of as a regular single objective GA that is simply trying to minimize the dominance count of all solutions.

3.2 Sexual Operations

The GA uses a standard two-point crossover operator. The mutation operator however, is specialized to suit the context of a linear trading sequence. Rather than flipping bits with a random probability, mutation for the HFS randomly selects a proportion ($pMutation$) of the trades (individual bits found in the bit string) and shifts each one to a random position left or right within the bounds of the two adjacent trades. This is intended to fine tune the range of existing trades rather than create new ones.

Only the GA bit string is genetically passed from parent to offspring; the reasoning behind this design decision is that the entry and exit trees of a strategy must be perfectly attuned to one another which is unlikely to result from slicing two different strategies. Future work may examine the sexual exchange of the GP trees themselves.

3.3 Additional Considerations

It is useful to place a number of extra restrictions on the behaviour of the system. For example, a minimum of 30 trades must be specified by a GA bit string for it to be considered valid; otherwise the solution is given the worst possible dominance count. This restriction avoids GA being able to create seemingly very predictable strategies who's task requires only spotting a handful of trades. A minimum of 30 trades ensures that there can be some degree of statistical confidence in the predictability metric. A second restriction disallows solutions onto the Pareto front who have a negative cumulative return over the testing period. Unlike the trade restriction we do not kill off unprofitable solutions as they may have

superior predictability performance. This genetic material could appear in future offspring who may find a more profitable use for their prediction abilities.

The system is not allowed to hold multiple positions or engage in *short* trades where a financial instrument is sold before being bought. A pair of trees with a higher cumulative return are always assumed to be the better forecasters.

After the GA population has evolved for a number of generations an interesting collection of models that cannot be beaten usually emerges along the Pareto front. At one extreme of this set there are solutions that have a very high predictability but low profitability and at the other extreme we find the opposite. It is typical practice in multi-object Pareto optimization to select a single solution from the middle of this set (the "knee") that has a reasonable trade off between all fitness traits. We choose instead to hold onto all solutions that are found on the Pareto front as a means of analysing the out of sample consistency between the models. If it were found that the out of sample validating performance varied greatly between all of the Pareto optimal solutions that claim profitability during testing then it may be an indication that the HFS approach does not work well for the particular financial instrument or time frame being used.

4 Experimental Setup

To test the HFS, ten well known equity stocks are selected from the NYSE and NASDAQ trading exchanges. Each data set was run through the system with the same functions as listed in Table 1 and parameters as listed in Table 2.

Table 1. GP function set. Both entry and exit trees use the same set of functions. For safety the *range* and t parameter inputs are always rounded off and made positive. The Division operator is also protected.

Standard GP Functions		**Technical Analysis Functions**	
$+, -, *, /$	Arithmetic	Diff(t)	Price diff at t
$>=, <$	Logic	Avg(x, y), MA(*range*)	Averages
$1000, 112, 56, 28, 1, 0.5$	Constants	Max(*range*), Min(*range*)	Extremes
		Band(*threshold*, *range*)	Threshold test

4.1 Data preparation

The system is fed on the raw price differences between 10,000 consecutive minutes that occurred during open US market hours. This quantity equates to about ten months of daily market action, though the exact calendar time varies from stock to stock due to data provider inconsistencies. In the tests presented in this paper a different start date is used for each stock; this is intentionally done to exposes the system to a large array of market conditions and not simply the ones observed most recently. The start date for each data set is picked at random within the confines of the available data (January 2004 to December 2007). Each block of 10,000 price values for each stock is divided into training, testing and validating

Table 2. List of the GA and GP parameters used for all experiments

GA Parameters		**GP Parameters**	
population	100	population	100
generations	20	generations	20
pCrossover	0.9	pCrossover	0.5
pMutation	0.1	pMutation	0.5
GA type	Steady State	selection	tournament, size 2
selection	Roulette Wheel	initialization	ramped half & half
type of crossover	standard two-point	maximum tree depth	5
type of mutation	custom (trade shift)	minimum tree depth	3

segments in the ratios of 40%, 40% and 20% respectively. The first 2000 points in each segment is reserved as a look back period to ensure that even the first case presented to the signal generators is guaranteed to have the full complement of prior data points for use as input.

The costs that one would expect to find trading in real markets such as spreads, commissions, margin calls or price slippage are not taken into account in this paper. Our concern here is solely focused on forecasting; the use of returns merely serves as a more intuitive tool for gauging forecasting success rather than as a guide to real market performance. This issue is addressed further in section 5.

4.2 Comparison Approaches

To ascertain the abilities of the HFS, its performance is compared against the returns of three comparison methods.

The first hurdle of comparative performance (and probably the most important) is **Buy & Hold**. It is the return received from a single position that lasts the entire duration of the validating period. The logic behind such *Buffet* [4] style investing is that the share prices moves around but ultimately increase in value over time.

The second comparison solution is called the **Maximum Random Strategy** and it is designed to assess the likelihood of arriving at a given level of forecasting performance simply by making a random sequence of trades. The average number of trades engaged in by the optimal solutions of the HFS is calculated for each stock. This average is then used as the number of trades randomly placed over the validating period to create a random strategy. Unique random seeds are used to create 30 of these random runs and the cumulative returns are recorded for each. The random run with the **highest** return is used as the comparison value. Of course, selecting the best performing random strategy in hindsight is not something that can be used as a real time strategy, it does however serve as an extremely difficult benchmark to compare the HFS strategies against. It is very probable that there will be at least one highly profitable sequence amongst the 30 random runs.

The final comparison solution is the **Greedy Predictor**. It is similar to the HFS but the GA part has the simple task of finding the sequence of trades that would of made the most money over the training and testing period (in hindsight). Then, just like the HFS, two GP signal trees are evolved to attempt to learn from this *greedy* sequence and the winning pair of signal trees are applied to the out of sample validating data. The Greedy Predictor approach is akin to a novice trader studying historic market *peaks* and *troughs* in the hopes of being able to forecast them in the future (see trades A and D in Figure 1 for an example).

5 Results

Table 3 shows the experimental results. The first thing to consider is the variation of the out of sample conditions. The **Buy & Hold** returns range from −9.26% to +7.35%, this alleviates any concerns we might have had about the period (2004 - 2007) only containing *bullish* (positive) conditions.

The **Greedy Forecaster** resulted in a large degree of variation from one run to the next, indicating poor predictive ability. Nevertheless the results were compared by averaging over 30 greedy runs instead of just one.

The **Maximum Random** strategy results are very high as expected, always resulting in positive returns (irrespective of the underlying stock's situation).

The percentage returns for the **HFS** models are very promising; they are always positive and always beat Buy & Hold (Table 3). In all experiments except **Boe** and **Cat** the HFS also beats the highest of the 30 random runs.

It is worth noting that the winning solutions created by the HFS all employ a *scalping* trading style, frequently engaging in positions that last only a few minutes. The strategy evolved for Microsoft (Msft), as an example, achieves a return of 35% over the validating period by making 6082 trades lasting on average 1.8 minutes each (Figure 3).

Table 3. Experimental results. Standard Deviations and number of models on the front are shown for HFS. All figures are for out of sample validating performance only, training and testing scores are not shown for brevity.

Stock	Period	**B&H**	**Greedy**	**MaxR**	**HFS**	std HFS	#HFS
Boe	2005-08-01→2005-10-04	3.53%	2.21%	**9.72%**	5.16%	0.39%	2
Cat	2007-08-07→2007-10-10	-0.11%	2.18%	**5.27%**	4.67%	6.77%	5
Coke	2006-03-31→2006-06-06	3.53%	0.85%	5.43%	**8.95%**	12.97%	3
GE	2007-08-27→2007-10-26	5.66%	1.61%	10.96%	**14.91%**	7.18%	6
Intel	2004-08-18→2004-10-08	-4.00%	-3.36%	16.49%	**58.74%**	1.82%	2
J&J	2004-01-11→2005-01-05	7.35%	4.86%	6.22%	**12.89%**	6.93%	7
Msft	2005-08-11→2005-10-06	-8.77%	-4.06%	4.68%	**35.85%**	19.28%	3
Nc	2006-05-11→2006-07-19	-9.26%	-3.74%	3.52%	**29.76%**	14.21%	3
Pfi	2006-10-30→2007-01-03	-4.14%	-4.07%	8.24%	**12.99%**	15.40%	10
Wmt	2007-04-03→2007-06-07	3.78%	1.49%	6.43%	**9.32%**	6.70%	2

```
Entry Tree:  (GTET  (Diff  (Avg  1  (MA  28 ) ) )  (MA  1 ) ) >= 1
Exit  Tree:  (GTET  (MA  (Avg  0.5  1 ) )  (MA  (-  0.5  28 ) ) ) >= 1
```

Fig. 3. A trading strategy evolved for Microsoft. The entry tree generates a buy signal under two circumstances: if the general price trend is rising and the immediate price difference is lower than the previous one, or, if the general trend is falling with an immediate price difference decline larger than the previous one. The exit tree gives a sell signal when the recent price differences are greater than the general trend. These trees are deceptively parsimonious due to their innovative use of the input protection mechanisms within the Moving Average (MA) function.

The highest average performance (**Intel**) achieved by the HFS is an astonishing **58.74**% return over about a two month period. Even in the worst case (**Boe**) the HFS model's average is 1.63% higher than buy & hold. Such results clearly demonstrate high predictive ability although it must be reiterated that having no cost penalty within the fitness function ultimately means the figures reported here cannot be viewed as real market returns. The strategies have been trained to trade as frequently as possible to maximize every last bit of profit. The magnitude of variation in a stock's price that is generally observed over a one minute period is not large enough to justify the *cost spread* that would be incurred during such trades. Of course, the alternative approaches (B&H, MaxR, Greedy) also benefit from the lack of costs, meaning the comparisons are fair ones. The primary concern for this work is to assess whether using selective training cases improves the predictive efforts of a trading system. The challenge of implementing the HFS approach under realistic market conditions is left to future work.

6 Conclusions

This paper demonstrates the idea of improving the performance of a financial forecasting system by finding a subset of market moves that correspond to predictable or inefficient times in that market. To implement this idea, the HFS attempts to first address the question *"what are realistic objectives for this data"* before it attempts to tackle those objective. Such an approach is counter to that of existing forecasting methods, where a system is given all of the training data and expected to solve the *ideal objectives* without regard to the feasibility of those objectives.

The performance of any forecasting methodology could potentially be improved by *training case selection*. It would be relatively easy to retrofit another forecasting methodology with an outer GA layer similar to the one described here, so long as the underlying algorithm uses discretized input cases and can produce a metric for its performance.

The results from HFS experiments seem to run contrary to two common notions about market prediction. First, it is generally accepted that *the more training examples the better*. The optimal solutions found by the HFS were all

trained on only a very small fraction of the total training cases available. The second counter intuitive result is that strategies trained on more ambitious training sequences did not result in higher out of sample profits. In fact, the opposite was the case. The **Greedy Predictor**, which is trained on the perfect sequence of trades (as calculated in hindsight) produced very inconsistent results, while the strategies found by the HFS were trained on a much more modest sequence of trades. For the purposes of financial forecasting it would seem that modesty is the best policy.

References

1. Aranha, C., Iba, H.: Using memetic algorithms to improve portfolio performance in static and dynamic trading scenarios. In: GECCO 2009, pp. 1427–1434 (2009)
2. Becker, Y., Fei, P., Lester, A.: Stock selection - an innovative application of genetic programming methodology. In: GPTP IV. Springer, Heidelberg (2007)
3. Cui, W., Brabazon, A., O'Neill, M.: Efficient trade execution using a GA in an order book based artificial stock market. In: GECCO 2009, pp. 2023–2028 (2009)
4. Cunningham, L.: The Essays of Warren Buffett: Lessons for Corporate America. The Cunningham Group, Orlando (1998)
5. Deb, K.D., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: Nsga-ii. IEEE Evolutionary Computation 6(2), 182–197 (2002)
6. Fama, E.: Efficient capital markets: A review of theory and empirical work. Journal of Finance 25, 383–417 (1970)
7. Gorgulho, A., Neves, R., Horta, N.: Using gas to balance technical indicators on stock picking for financial portfolio composition. In: GECCO 2009. pp. 2041–2046 (2009)
8. Hirabayashi, A., Aranha, C., Lba, H.: Optimization of the trading rule in foreign exchange using genetic algorithm. In: GECCO 2009, pp. 1529–1536 (2009)
9. Kaboudan, M.A.: Genetically evolved models and normality of their fitted residuals. Journal of Economic Dynamics and Control 25(11), 1719–1749 (2001)
10. Karabuluta, K., Alkanb, A., Yilmazb, A.S.: Long term energy consumption forecasting using gp. Mathematical and Computational Applications 13, 71–80 (2008)
11. Larkin, F., Ryan, C.: Good news: Using news feeds with genetic programming to predict stock prices. In: O'Neill, M., Vanneschi, L., Gustafson, S., Esparcia Alcázar, A.I., De Falco, I., Della Cioppa, A., Tarantino, E. (eds.) EuroGP 2008. LNCS, vol. 4971, pp. 49–60. Springer, Heidelberg (2008)
12. Nassar, D.: Rules of the Trade: Indispensable Insight for Online Profits. McGraw Hill, Columbus (2001)
13. Nevmyvaka, Y., Feng, Y., Kearns, M.: Reinforcement learning for optimized trade execution. In: ICML 2006, pp. 673–680 (2006)
14. O'Neill, M., Brabazon, A., Ryan, C., Collins, J.J.: Evolving market index trading rules using grammatical evolution. In: Boers, E.J.W., et al. (eds.) EvoWorkshops 2001. LNCS, vol. 2037, pp. 343–352. Springer, Heidelberg (2001)

Threshold Recurrent Reinforcement Learning Model for Automated Trading

Dietmar Maringer and Tikesh Ramtohul

Universität Basel, CH-4002 Basel, Switzerland
{Dietmar.Maringer,Tikesh.Ramtohul}@unibas.ch

Abstract. This paper presents the threshold recurrent reinforcement learning (TRRL) model and describes its application in a simple automated trading system. The TRRL is a regime-switching extension of the recurrent reinforcement learning (RRL) algorithm. The basic RRL model was proposed by Moody and Wu (1997) and used for uncovering trading strategies. We argue that the RRL is not sufficiently equipped to capture the non-linearities and structural breaks present in financial data, and propose the TRRL model as a more suitable algorithm for such environments. This paper gives a detailed description of the TRRL and compares its performance with that of the basic RRL model in a simple automated trading framework using daily data from four well-known European indices. We assume a frictionless setting and use volatility as an indicator variable for switching between regimes. We find that the TRRL produces better trading strategies in all the cases studied, and demonstrate that it is more apt at finding structure in non-linear financial time series than the standard RRL.

Keywords: regime-switching, automated trading, reinforcement learning, differential Sharpe ratio.

1 Introduction

Advances in artificial intelligence techniques and computational power have opened up exciting avenues of research in the world of finance and economics, especially in the field of algorithmic trading. One such technique, known as recurrent reinforcement learning (RRL) was proposed by [1]. It has an autoregressive outlook and can be likened to a single-layer recurrent neural network. Previous work has already shown that the RRL offers good promise in finding profitable strategies in financial markets. [1] and [2] used the RRL methodology for training trading systems and portfolios by optimizing risk-adjusted objective functions such as the differential Sharpe ratio (DSR) and the differential double deviation ratio (see [3]). The authors used both artificial data and real-world FX data for their experiments and found out that the RRL-traders led to mostly profitable situations and outperformed traders trained to maximise profits or minimise error functions. [4] extended the model to a two-layer neural network and subsequently drew comparisons between the effectiveness of this variant with

C. Di Chio et al. (Eds.): EvoApplications 2010, Part II, LNCS 6025, pp. 212–221, 2010.

the original single-layer network: FX data was used for this purpose and it was found that the two-layer version underperformed the original RRL. Moreover, for some markets, neither set of mechanical traders could come up with profitable strategies. [5] used the RRL as part of a system with a layered structure for trading in FX markets, and talked about the ability of the system to efficiently exploit structure in past returns time series. More recently, [6] used an objective function that corresponds to the ratio of the cumulative positive trading returns to the cumulative negative trading returns for their RRL-trading system. They used daily data and reported profitable outcomes in all but one case.

Despite the relative success of the single-layer RRL model, it can be argued that its linear outlook makes it ill-suited to capture all the intricate aspects of financial data. An approach with a higher degree of non-linearity could very much aid in increasing its predictive capabilities. One straightforward way of accounting for the non-linearities is to incorporate hidden layers in the network. But, [4] noted a decline in performance when he introduced a hidden layer in the RRL topology. Indeed, multi-layer models are prone to overfitting, especially with noisy financial data, and are quite often unable to generalise properly. Moreover, such black-box approaches render inference about the input-output relationship difficult, if not impossible. A certain degree of transparency ensures that automated trading systems are more tractable, thereby allowing the human expert to adopt remedial measures or perform fine-tuning more efficiently whenever performance starts to degenerate. Thus there is a need for non-linear models that can perform well out-of-sample and that can shed some light on how economic variables affect financial markets. Regime-switching models provide an elegant solution to this kind of problem. These models define different states of the world (regimes), and assume that the dynamic behaviour of economic variables depends on the regime that occurs at any given point in time. This implies that certain properties of the time series, such as its mean, variance, autocorrelation, etc., are different in different regimes. Such models offer a great deal of transparency and the concept of regimes helps to capture non-linearities. Moreover, the regime-switching framework is more adapted for modelling dramatic changes in behaviour in economic time series, as a consequence of events such as financial crises or major changes in government policy [7].

We propose a new model, called Threshold Recurrent Reinforcement Learning (TRRL), that augments the existing RRL with regime-switching properties. We argue that the TRRL is more suited at dealing with the non-linear properties of financial data and therefore can lead to more profitable trading strategies. The principal goal of this paper is to describe the main building blocks of this model, and compare its performance with the basic RRL in an automated trading setting using daily financial data from four well-known European stock indices.

The outline of the paper is as follows: in Section 2, we briefly review the basic RRL model and focus on how it can be repackaged to formulate the regime-switching RRL model, as well as present the main equations related to the online learning procedure via maximisation of the differential Sharpe ratio. This is followed by a section on the experiments carried out to compare the two

methodologies. Section 4 provides concluding remarks and discusses possibilities for future work.

2 Model Description

2.1 Threshold Recurrent Reinforcement Learning

Recurrent reinforcement learning, proposed by [1], is an adaptive policy search algorithm which tries to maximise a certain performance criterion in order to learn profitable investment strategies. As its name suggests, the system is recurrent, meaning that the current investment decision has a say in shaping future decisions. In the presence of transaction costs, investment performance depends on sequences of interdependent decisions; the recurrent nature of the algorithm takes this path-dependency into account [2]. For a single-asset, two-position trader who can take only long or short positions of constant magnitude, the trading function is

$$F_t = \tanh\left(\sum_{i=0}^{m} w_i r_{t-i} + w_{m+1} F_{t-1} + w_{m+2} v\right). \tag{1}$$

F_t is the output of the network at time t. A long position is adopted when $F_t > 0$; the trader buys an asset at time t and makes a profit if the price goes up in the next time step. If $F_t < 0$, the trader short sells an asset at time t and makes a profit if the price goes down at time $t + 1$. The price return r_t corresponds to the difference in value of the asset between the previous period and the current period, i.e. $r_t = p_t - p_{t-1}$. The term v is the familiar bias present in neural network models, typically having a value of 1. The w_i's denote the system parameters or network weights that need to be optimised. Note that the time indexation of the weights has been dropped for clarity. The term F_{t-1}, i.e. the trade position at the previous time step, induces recurrence and hence some kind of internal memory. The RRL model is not restricted to taking only lagged price returns as input. It can easily accommodate technical indicators or other economic variables that might have an impact on the security.

It was argued earlier that such a system might fail in capturing all the intricacies of financial markets, and a regime-switching version could potentially be more adapted to such highly non-linear environments. There exists some well-established regime-switching methods that have gained prominence in econometrics. These include the threshold model, initially proposed by [8], the Markov-Switching model of [9], the artificial neural network model of [10], and the smooth transition model (see [11]), the latter being a more general version of the threshold model. While all of these could potentially be used to extend the basic RRL, the threshold model is more appealing for this particular problem because of its simplicity and the degree of transparency that it offers. It assumes that a regime is determined by the value of an indicator variable q_t relative to a threshold value c. Suppose that we have a 2-regime model for some dependent variable y_t, and

each regime is characterised by an autoregressive process of order 1, i.e. $AR(1)$ process. The threshold model can be expressed as

$$y_t = (\phi_{0,1} + \phi_{1,1} y_{t-1})\,(1 - I\,[q_t > c]) + (\phi_{0,2} + \phi_{1,2} y_{t-1})\,(I\,[q_t > c]) + z_t \tag{2}$$

where z_t denotes an i.i.d white noise process, and $I[X]$ is an indicator function with $I[X] = 1$ if event X occurs and $I[X] = 0$ otherwise. This model offers great flexibility and can easily be modified to account for higher-order ARMA models in the different regimes.

The threshold version of the recurrent reinforcement algorithm can be formulated by considering (1) and (2). A two-regime system having indicator variable and threshold denoted by q_t and c respectively, can be described as

$$F_t = y_{t,1}\,(I\,[q_t > c]) + y_{t,2}\,(1 - I\,[q_t > c]) \tag{3a}$$

$$y_{t,j} = \tanh\left(\sum_{i=0}^{m} w_{i,j} r_{t-i} + w_{m+1,j} F_{t-1} + w_{m+2,j} v\right) \quad \text{for } j = \{1, 2\} \tag{3b}$$

The TRRL system can be thought of having two RRL networks, each one corresponding to a particular regime and having a distinct set of weights. The overall output F_t of the system is the weighted sum of the the outputs y_1 and y_2 of the individual networks. The weighting factor is actually the value of the indicator variable, and is determined by the prevailing regime. Initially, both networks have the same set of weights. During training, the model promotes selective learning and this leads to each network developing a unique set of weights. If the system is in a particular regime, the network associated with that regime is exposed to higher weight updates than the other. Each network learns a distinct mapping that corresponds to a specific region in the space spanned by the indicator variable. The latter effectively acts as a switch or gating device that selects the appropriate network at each time step.

The threshold model can be easily modified to represent a more gradual (smooth) transition between regimes or extended to account for multiple regimes and/or multiple indicator variables (see [12] for more details). These regime-switching models can be thought of as a teacher-directed learning variant of the mixtures of experts (MoE) neural network model developed by [13]. In the MoE models, a gating network learns to adaptively partition the input space and assign one expert for each partition. In the regime-switching RRL models, the partitioning is not part of the actual learning process. The input space is broken down in an arbitrary manner, based on the knowledge and beliefs of the designer, and this teacher information is then used to update the expert networks.

2.2 Differential Sharpe Ratio for Online Learning

The learning process of the TRRL is in essence similar to that of the RRL described in [1]. It involves maximising a certain performance criterion to obtain a set of network weights that can lead to profitable strategies. However, in the

case of the TRRL, the indicator variable has a great say in directing the outcome of the learning phase. The selection of the appropriate indicator variable is of critical importance. The very nature of the algorithm calls for an indicator variable that has certain desirable characteristics. First and foremost it must have an impact on the serial correlation of the price returns process. Next, for proper learning, frequency of switching between regimes should be reasonable. Excessive switching tends to destabilise the learning process, while unreasonably low switching frequencies lead to situations where the system is reliant on very old information; this is not desirable since financial markets are dynamic, meaning that patterns that were present a long time ago might not be present now. In addition, as is always the case for regime-switching models, it is required that each regime contains a minimum percentage of the observations. Finally, for reliability purposes, it should preferably be an observable variable; otherwise, if a latent variable is to be chosen, the estimation techniques should be fast and relatively simple.

Another important aspect concerns the choice of objective function for optimisation. [2] showed that RRL systems trained by maximising risk-adjusted performance criteria perform better than those trained by minimizing error functions. They used stochastic gradient ascent to maximise the differential Sharpe ratio (DSR), a variant of the well-known Sharpe ratio introduced by [14]. The DSR is derived by making use of exponential moving average estimates of the first and second moments (A_t and B_t respectively in (5), as described in [2]) of the trading returns distributions. The same approach has been adopted in this paper. The trading return R_t, as defined by [2], is expressed as

$$R_t = r_t^f + \text{sign}\,(F_{t-1})\left(r_t - r_t^f\right) - \delta\left|\,\text{sign}\,(F_t) - \text{sign}\,(F_{t-1})\,\right| \tag{4}$$

where r_t^f is the risk-free rate of interest and δ is the transaction cost rate per share traded . Note that both of these terms have been assumed to be zero in this study. The exponential moving average Sharpe ratio can be expressed in terms of the trading return R_t. It is given by

$$S_t = \frac{A_t}{\sqrt{B_t - A_t^2}} \tag{5}$$

where

$$A_t = A_{t-1} + \eta\,(R_t - A_{t-1})$$
$$B_t = B_{t-1} + \eta\left(R_t^2 - B_{t-1}\right)$$

The DSR is obtained by expanding the exponential moving average version to first order in the adaptation rate η [2]. It is defined as

$$D_t = \frac{B_{t-1}\,(R_t - A_{t-1}) - \frac{1}{2}A_{t-1}\left(R_t^2 - B_{t-1}\right)}{\left(B_{t-1} - A_{t-1}^2\right)^{\frac{3}{2}}} \tag{6}$$

It can be optimised incrementally using gradient ascent. If ρ corresponds to the learning rate, the weight update equation is given by

$$w_{t,j} = w_{t-1,j} + \rho \Delta w_{t,j} \qquad \text{for } j = \{1, 2\} \tag{7}$$

where

$$\Delta w_{t,j} = \frac{dD_t}{dR_t} \left(\frac{dR_t}{dF_t} \frac{dF_t}{dw_{t,j}} + \frac{dR_t}{dF_{t-1}} \frac{dF_{t-1}}{dw_{t-1}, j} \right).$$

The derivative $\frac{dF_t}{dw_t}$ for online training can be computed using an approach similar to backpropagation through time (BPTT) introduced by [15] and discussed in [2],

$$\frac{dF_t}{dw_{t,j}} \approx \frac{\partial F_t}{\partial w_{t,j}} + \frac{\partial F_t}{\partial F_{t-1}} \frac{dF_{t-1}}{dw_{t-1,j}} \qquad \text{for } j = \{1, 2\} \tag{8}$$

where

$$\frac{\partial F_t}{\partial w_{t,j}} = \frac{\partial F_t}{\partial y_{t,j}} \times \frac{\partial y_{t,j}}{\partial w_{t,j}}$$

$$\frac{\partial F_t}{\partial F_{t-1}} = \sum_{j=1}^{2} \left(\frac{\partial F_t}{\partial y_{t,j}} \times \frac{\partial y_{t,j}}{\partial F_{t-1}} \right).$$

All the required derivatives can be computed using basic differentiation rules, and thus the weight update process turns out to be rather straightforward and relatively fast.

3 Experiments

3.1 Setup

This section describes the experiments carried out to gauge the efficiency of the TRRL relative to the basic RRL model. The simulation results are presented and an assessment of the main findings is given. The experiments aimed at comparing the abilities of both sets of traders to discover structure in real financial price series. The traders studied were of the $\{long, short\}$ type who could buy/short sell only 1 share at a time. Transaction costs were assumed to be zero. The training phase consisted of subjecting the traders to training data of length L_{tr} for a number of epochs n_e. The trades made during the training period allow the systems to update their weights in a bid to generalise properly when faced with novel data. The performance of the traders was assessed by considering the trades made during an ensuing period L_{te}. The static Sharpe ratio was used as the performance measure. Both types of traders were subjected to identical initial conditions and model parameters, and adapted using either RRL or TRRL to optimize the DSR. Initial conditions here refer to the initial set of weights

assigned to the networks. The weights were sampled from a uniform distribution such that $-0.1 \leq w_i \leq 0.1$. The model parameters include the learning rate ρ, the adaptation rate η, the number of lagged price return inputs m, the size of the training window L_{tr},the number of training epochs n_e, and the size of the test window L_{te}. The values used were $L_{tr} = 2000$, $L_{te} = 250$, $m = 5$, $n_e = 5$, $\rho = 0.01$, and $\eta = 0.01$, inspired from previous work by [3] and [4].

The indicator variable used in this paper is volatility. Empirical studies carried out by [16], [17], and [18] by and large indicated that an upsurge in volatility increases the likelihood of negative autocorrelation in returns (see [12] for more details). Morever, since volatility is known to have a persistent nature, there is little risk of the trading system suffering from excessive regime switching. Although volatility is a latent indicator variable, various standard techniques that ally speed and reliability exist for its estimation. For our experiments, a simple GARCH(1,1) framework was chosen to represent the volatility process. The choice was motivated by the parsimony and the widely-acknowledged reliability of this type of model. For each dataset, the respective GARCH parameters were estimated over the training data, and subsequently used to make one-day-ahead forecasts during the test period. The median of the volatility distribution spanning the training period was taken as the threshold for the regime-switching framework.

3.2 Data

The data consisted of daily closing prices of 4 well-known European stock indices, namely the French market index (CAC 40), the FTSE 100 index from the London Stock Exchange, the DAX from the Frankfurt Stock Exchange, and the Swiss market index (SMI). A nine-year period from December 1999 upto October 2008 was considered. It was divided into a training portion consisting of 2000 datapoints (roughly 8 years of data), and a test portion of 250 datapoints corresponding to the period between Oct. 2007 and Oct. 2008. The price return at time t is defined by $r_t = 100 \times [\ln(p_t) - \ln(p_{t-1})]$ where p_t is the price index at time t. The resulting return series was multiplied by a factor c ($c = 0.5$ for our experiments) before being input to the network to ensure that the lagged returns are on a scale comparable to the F_{t-1} values. Figure 1 illustrates the relevant financial time series for the SMI. The other indices have very similar profiles. The price series reflect chronologically the bursting of the dot-com bubble, the transition to the US housing bubble and the subsequent deflation of the latter which eventually led to the current crisis. The corresponding volatility profiles, and to some extent the return series, show how markets went from turbulent to tranquil and back to turbulent again. More specifically, the test period is predominantly in a high-volatility regime for all the datasets.

3.3 Results and Discussion

Figure 2 summarises the test results for an ensemble of 1000 cases for each index. It can be seen that both types of traders yield mostly positive out-of-sample

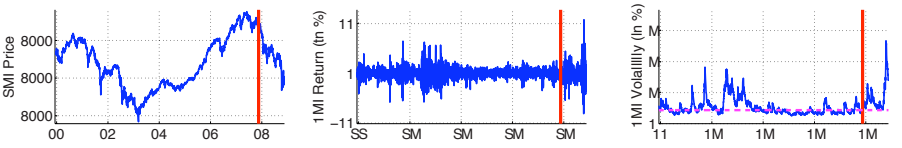


Fig. 1. The price series, return series and volatility estimates for the SMI dataset for the period December 1999 to October 2008. The vertical line separates in and out of sample periods, the horizontal dotted lines in the rightmost graph correspond to the median of the volatility distribution for the training period.

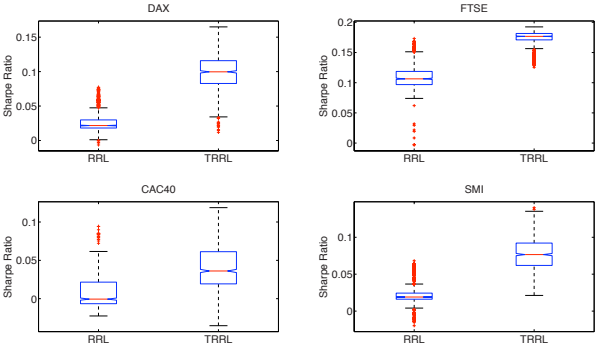


Fig. 2. Box plots comparing the out-of-sample performance of an ensemble of 1000 RRL-traders against the TRRL-traders

Sharpe ratios, with the TRRL-traders outperforming the RRL-traders in each case. The disparity in performance can be explained by looking at the volatility profile. There is a definite switch from a low-volatility to a high-volatility regime just before the test period begins. The RRL is unable to adjust to its new environment quickly enough, and therefore performs poorly. The TRRL however develops two sets of weights, one for each regime. The first set undergoes rigorous weight updates in the first half of the training period and very little change in the second half, and vice versa for the second set. The first set of weights is thus well-suited for highly volatile periods. The high volatility in the test period implies that the TRRL bases its trade decisions on the first set of weights rather than the second. Thus, the significant regime change just before the test period does not have a detrimental effect on the out-of-sample performance of the TRRL.

The superiority of the TRRL is also quite telling about the choice of volatility as an indicator variable, at least for the datasets considered. It is able to broadly describe the shifts from one economic regime to another experienced by financial markets over the last decade or so. And interestingly enough, analysis of the weights developed by the TRRL tend to support the empirical findings about the relationship of serial correlation with volatility. For the datasets studied, the set of weights developed for the high-volatility regime is typically more negative

than that developed for the low-volatility regime. Although the weights cannot be directly related to the coefficients of serial correlation, they do act as a good proxy, by virtue of the autoregressive construction of the algorithm.

Because of the frictionless settings assumed in this study, some degree of caution has to be preached while considering the relatively high out-of-sample Sharpe ratios achieved by the majority of the traders. With the inclusion of transaction costs, the out-of-sample performance of both types of traders would undoubtedly degrade, but it is realistic to assume that, for the datasets considered, the TRRL-traders would still outperform their RRL counterparts.

In [12], controlled experiments using artificial data showed that in general, the TRRL perform as well as the standard RRL if there is one prevailing regime, or if regimes are closely related to one another. However, in the presence of distinctly different regimes, as in the case with autocorrelation of different sign in each regime, the TRRL is more efficient. Thus, the TRRL does not undermine the basic working principles of the RRL, meaning that it can still capture temporal dependencies, just like the RRL, and on top of that, the added threshold non-linearity increases its predictive capabilities.

4 Conclusion

In this paper, we proposed the TRRL model, which is a regime-switching version of the RRL algorithm put forward by [1]. We compared the performance of both of these algorithms in an automated trading setting using daily return series of four well-known European indices. We also emphasized on the importance of correct identification of the regimes, and advocated the use of volatility as an indicator variable for the construction of the TRRL while dealing with real-world financial time series data. Results showed that the TRRL is better than the RRL in finding profitable strategies, especially in the aftermath of a drastic regime change. The majority of TRRL-traders yielded positive Sharpe Ratios out-of-sample for all 4 datasets, and significantly outperform the RRL-traders. The results thus back the notion of integrating regime-switching with the RRL methodology, and also demonstrate the viability of using volatility as an indicator variable. However, for more conclusive inferences, a more realistic setting must be used as well as additional datasets.

The flexibility of the model presented implies that it can be easily modified to suit the financial problem being investigated. For example, it could be easily extended to trade a portfolio of securities or for a simple asset allocation problem. It could be used for trading commodities or Forex instruments instead of stocks, or applied to other problems where investment decisions need to be taken. If the situation requires more than two regimes and/or more indicator variables, the model can easily be extended to account for that. Thus, the model presented can be customised and tailor-made to the needs of the problem, and the indicator variables hand-picked to best match the features of the application environment and the beliefs of the designer.

Acknowledgements. We are deeply grateful to the anonymous referees for their valuable comments and would like to thank the participants of the 36th Macromodels International Conference who provided some insightful remarks regarding this particular study.

References

1. Moody, J., Wu, L.: Optimization of trading systems and portfolios. In: Proceedings of the IEEE/IAFE Computational Intelligence for Financial Engineering (CIFEr), pp. 300–307 (1997)
2. Moody, J., Wu, L., Liao, Y., Saffell, M.: Performance functions and reinforcement learning for trading systems and portfolios. Journal of Forecasting 17(56), 441–470 (1998)
3. Moody, J., Saffell, M.: Learning to trade via direct reinforcement. IEEE Transactions on Neural Networks 12(4), 875–889 (2001)
4. Gold, C.: FX trading via recurrent reinforcement learning. In: IEEE International Conference on Computational Intelligence for Financial Engineering, pp. 363–370 (2003)
5. Dempster, M., Leemans, V.: An automated FX trading system using adaptive reinforcement learning. Expert Systems with Applications 30(3), 543–552 (2006); Intelligent Information Systems for Financial Engineering
6. Bertoluzzo, F., Corazza, M.: Making financial trading by recurrent reinforcement learning. In: Apolloni, B., Howlett, R.J., Jain, L. (eds.) KES 2007, Part II. LNCS (LNAI), vol. 4693, pp. 619–626. Springer, Heidelberg (2007)
7. Hamilton, J.: Regime-switching models. In: The New Palgrave Dictionary of Economics. Palgrave Macmillan, Basingstoke (2008)
8. Tong, H.: On a threshold model. In: Chen, C. (ed.) Pattern Recognition and Signal Processing, pp. 101–141. Sijthoff & Noordhoff (1978)
9. Hamilton, J.D.: A new approach to the economic analysis of nonstationary time series and the business cycle. Econometrica 57(2), 357–384 (1989)
10. White, H.: Some asymptotic results for learning in single hidden-layer feedforward network models. Journal of the American Statistical Association, 1003–1013 (1989)
11. Teräsvirta, T.: Specification, estimation, and evaluation of smooth transition autoregressive models. Journal of the American Statistical Association, 208–218 (1994)
12. Maringer, D., Ramtohul, T.: Regime-switching recurrent reinforcement learning. Mimeo University of Basel (2009)
13. Jacobs, R., Jordan, M., Nowlan, S., Hinton, G.: Adaptive mixtures of local experts. Neural computation 3(1), 79–87 (1991)
14. Sharpe, W.: Mutual fund performance. Journal of Business, 119–138 (1966)
15. Werbos, P.: Backpropagation through time: what it does and how to do it. Proceedings of the IEEE 78(10), 1550–1560 (1990)
16. LeBaron, B.: Some relations between volatility and serial correlations in stock market returns. Journal of Business 65(2), 199–219 (1992)
17. Sentana, E., Wadhwani, S.B.: Feedback traders and stock return autocorrelations: Evidence from a century of daily data. Economic Journal 102(411), 415–425 (1992)
18. Koutmos, G.: Feedback trading and the autocorrelation pattern of stock returns: further empirical evidence. Journal of International Money and Finance 16(4), 625–636 (1997)

Active Portfolio Management from a Fuzzy Multi-objective Programming Perspective

Nikos S. Thomaidis

Management and Engineering Laboratory, Department of Financial & Management Engineering, University of the Aegean, 8 Michalon Str., GR-821 00, Chios, Greece
Tel.: +30-2271-0-35454 (35483); Fax: +30-2271-0-35499
nthomaid@fme.aegean.gr
http://fidelity.fme.aegean.gr/decision

Abstract. We consider the problem of structuring a portfolio that outperforms a benchmark index, assuming restrictions on the total number of tradable assets. We experiment with non-standard formulations of active portfolio management, outside the mean-variance framework, incorporating approximate (fuzzy) investment targets and portfolio constraints. To deal with the inherent computational difficulties of cardinality-constrained active allocation problems, we apply three nature-inspired optimisation procedures: simulated annealing, genetic algorithms and particle swarm optimisation. Optimal portfolios derived from these methods are benchmarked against the Dow Jones Industrial Average index and two simpler heuristics for detecting good asset combinations, based on Monte-Carlo simulation and fundamental analysis.

Keywords: Active portfolio management, Fuzzy multi-objective programming, Cardinality constraints, Evolutionary computation, Simulated annealing, Genetic algorithms, Particle swarm optimisation.

1 Introduction

Active portfolio management, or *enhanced indexation*, is a collective investment scheme where the performance of the portfolio strategy is measured relatively to a benchmark market index. It is well known that many institutional investors follow this benchmarking procedure, by focusing e.g. on the course of a broad market index such as the S&P 500. In the recent years, active index trackers have become popular and in many cases have outperformed more passively placed funds that aspire to closely reproduce benchmark's returns.

In practice, a number of implementation issues hinder the task of setting-up an active strategy and generally reduce its investment performance. Active management typically involves frequent rebalancing of the portfolio, through careful stock-picking, which incurs high management and transaction costs. In the light of these issues, a natural decision problem is how to actively reproduce index performance whilst limiting the overall effect of market "frictions". This is achieved, in practice, by keeping the rebalancing rate low, or most importantly, imposing a limit on the maximum number of tradable assets, the so-called *cardinality constraint*.

C. Di Chio et al. (Eds.): EvoApplications 2010, Part II, LNCS 6025, pp. 222–231, 2010.

Nowadays, there exist a plethora of methodologies for structuring index tracking portfolios with cardinality constraints (some references are given in following sections; see also [1] for a comprehensive survey). Despite the numerous applications of passive portfolio selection, active tracking formulations - especially those incorporating cardinality constraints - have not yet attracted full attention. In [8] we consider three alternative formulations of active portfolio management, based on the traditional mean-variance view of portfolio selection. In this paper, we expand upon this issue by making an attempt to incorporate non-standard objectives related to portfolio performance. These focus, for example, on the probability that the index portfolio delivers positive return relatively to the benchmark or on restricting the total risk of the investment strategy. Such targets/constraints on portfolio performance can be effectively formulated within the framework of *fuzzy mathematical multi-objective programming*.

The rest of the article is structured as follows: enhanced index tracking with cardinality constraints is studied in section 2, while section 3 discusses the nature-inspired optimisation heuristics employed in our study to solve active portfolio formulations. Section 4 details an alternative, "fuzzy", conceptualisation of active portfolio management structured around approximate investment targets and portfolio constraints. In section 5 we evaluate the performance of fuzzy portfolios in terms of actively reproducing the American DJIA index. Section 6 summarises the main findings and proposes future research directions.

2 Index Tracking with Cardinality Constraints

Consider the following investment problem whereby a fund manager has to decide the best subset of K stocks ($K < N$) that can actively reproduce returns on a benchmark index I as well as the appropriate percentage of capital w_i that should be invested in each stock i. The active portfolio optimisation problem can be stated in its general form as $\underset{\mathbf{w}\in R^N,\, \mathbf{s}\in\{0,1\}^N}{\text{maximise}}\ f(\mathbf{w},\mathbf{s})$ subject to $\sum_{i=1}^{N} w_i = 1$, $w^l \leq w_i \leq w^u$, $i = 1, ..., N$ and $\sum_{i=1}^{N} s_i \leq K \leq N$, where $f(\mathbf{w},\mathbf{s})$ is the portfolio's active objective, which is a function of weights $\mathbf{w} = (w_1, w_2, ..., w_N)'$, a vector of binary variables $\mathbf{s} = (s_1, s_2, ..., s_N)'$ and sample data. Several choices for $f(.,.)$ are later discussed in section 4. Each s_i is an indicator function that takes the value 1 if asset i is included in the portfolio and 0 otherwise. All asset weights sum to one, implying that the initial capital is fully invested, and the maximum number of assets allowed in the portfolio should not exceed $K \leq N$, which is the *cardinality constraint*. Depending on the cardinality of the portfolio P, some of the w_i's may be zero and if an asset i is included then the fund manager may impose a lower or upper limit on its weight, w^l and w^u, respectively, the so-called *floor* and *ceiling constraint*.

The introduction of cardinality constraints significantly increases the computational effort associated with deriving optimal portfolio allocations. In fact, one ends up with a mixed nonlinear-integer programming problem, which even

for small values of N becomes a challenge for gradient-based optimisation techniques. In this paper, we adopt a solution strategy that overcomes the difficulties of handling integer constraints by introducing a transformation of the decision variables (see [8,6] for details). The underlying idea is to solve an unconstrained optimisation programme with decision variables $(x_1, ..., x_N) \in \mathbb{R}^N$ and use a deterministic function to map the values of $(x_1, ..., x_N)$ onto a feasible portfolio allocation $(w_1, w_2, ..., w_N)$, where K out of N weights are zero.

3 Nature-Inspired Optimisation Algorithms

Many real-life optimisation problems in finance are considered "hard" due to combinatorial explosion or the ruggedness of the optimisation landscape. Traditionally, practitioners approach these problems adopting techniques that make use of relaxation and decomposition principles, such as branch & bound, dynamic programming and quadratic line-search. In the last decades, a number of intelligent computational algorithms, such as simulated annealing, tabu search, genetic algorithms, ant colonies and particle swarms, have been developed to solve a wide range of practical optimisation problems.

In this study, we experiment with three popular nature-inspired computational schemes in the task of solving active portfolio optimisation problems. We first present a trajectory-based strategy, namely *simulated annealing*, and then introduce two evolutionary computational heuristics, *genetic algorithms* and *particle swarm optimisation*.

Simulated annealing (SA) took its name and inspiration from the annealing, a technique used in metallurgy involving heating and controlled cooling of a solid ([5]). The central idea is to start with some arbitrary solution and have it modified by randomly generating a number (or else *population*) of new solutions in its vicinity. To overcome the possibility of premature convergence to local optima, the algorithm also accepts solutions that come with impairment, yet with decreasing probability. *Genetic algorithms* (GAs) employ an alternative solution-search strategy by forcing a parallel exploration of the solution space by means of several agents that interact with each other. Inspired by the process of natural selection that drives biological evolution, a genetic algorithm repeatedly modifies a population of solutions until it "evolves" towards a "fit" generation ([7]). Among the numerous forms in which genetic algorithms appear in the literature, in this paper we adopt a simple version of the algorithm that encodes solutions into vectors of real numbers and uses three genetic operators to move towards fitter populations: elitist selection, crossover and mutation. *Particle swarm optimisation* (PSO) is another computational heuristic inspired by the behaviour of biological flocks or swarms [3,4]. Instead of applying genetic operators, PSO flows "particles" in the search space with a velocity and direction that are dynamically adjusted according to each particle's learning experience (the best solution detected by the particle in its own exploration) and the swarm's collective memory (the global best solution found by any particle in the swarm).

4 Traditional vs. Fuzzy Enhanced Indexation

Much of portfolio selection is about setting aspiration levels for performance indicators as well as constraints on the total risk of the investment strategy. This is often done, in practice, using mathematical programming formulations that require precise definition of objectives and an accurate expression of preference towards return and aversion towards risk. In the real-world, however, investment strategies are often structured around imprecise statements of investors about the required return or the risk profile of trading positions: "*Given current market conditions, I would appreciate an annual return of* ***not much more*** *than 5% in excess of the benchmark*" or "*The probability of shortfall should* ***not significantly*** *exceed 20%*". Fuzzy optimisation theory offers a very convenient framework for accommodating such approximate linguistic-type information, through the introduction of *fuzzy goals* and *constraints* [2].

To illustrate the idea, let us assume that the investor sets the objective for a performance indicator x to *significantly exceed* a certain level s_0. This imprecise goal can be modelled by means of the sigma-shaped function depicted in the upper plot of Fig. 1. The x-axis represents all possible values for x while the y-axis measures the overall degree of goal attainment. Note that the investor effectively rejects solutions for which the value of x falls below s_0, by assigning a degree of satisfaction equal to 0. As x increases, so does the goal fulfilment and the investor is practically indifferent between any solution for which the value of x exceeds s_1 (the upper aspiration level). In a similar fashion, one can formulate goals or constraints of the type "x should be *much less* than z_1" or "x should be *approximately equal* to p_2" using zeta- or pi-shaped functions depicted in Fig. 1.

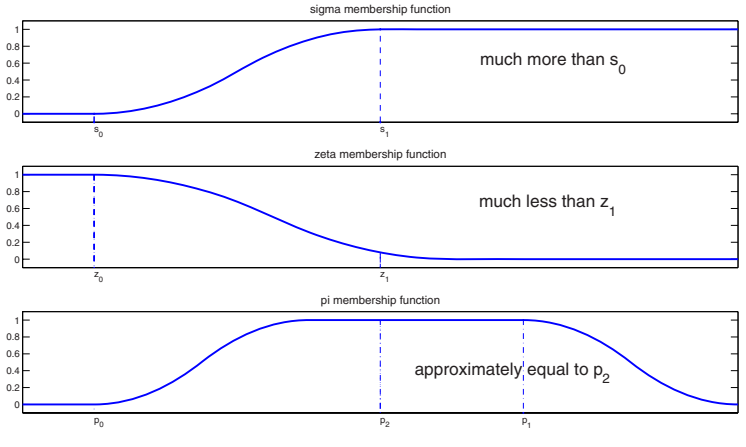


Fig. 1. Examples of fuzzy goals and constraints

The fuzzy linguistic framework presented above can be applied in enhanced indexation, by choosing a suitably defined objective function $f(.)$ for the portfolio-selection problem discussed in section 2. In our study, we experimented with portfolio objectives of the form: *restrict* the probability of under-performing the benchmark, while keeping the tracking error standard deviation *small*. This composite investment requirement is formulated by properly combining zeta-shaped functions, resulting in the following maximising function $f \equiv z(s_{TE}, 1\%, 10\%) \cdot z(P^-, 10\%, 50\%)$. In this equation, $z()$ denotes the zeta-shaped function, s_{TE} is the standard deviation of sample tracking error and P^- is the probability of under-performing the benchmark. As seen from the particular choice for the cutting points of f, the fund manager is unsatisfied with portfolio configurations whose (annualised) s_{TE} is above 10% and whose probability of going below the benchmark exceeds 50%. On the contrary, he is perfectly happy, and in fact indifferent between any active formulation that manages to jointly keep the tracking error standard deviation below 1% (on an annual basis) and the probability of under-performance below 10%.

5 Empirical Study: Actively Reproducing the Dow Jones Industrial Average Index

5.1 Sample Data and Experimental Design

We evaluate the performance of the methodology presented above in terms of actively reproducing the Dow Jones Industrial Average (DJIA) index. Our sample data includes daily closing prices of the DJIA index as well as its 30 constituent stocks covering the period from 21/01/2004 to 12/01/2006. This amounts to a total of 500 trading days, half of which were used for deriving optimal portfolio configurations and the remaining half for out-of-sample evaluation of trading performance.

To fill in the input list of the portfolio optimisation problems, we set the range of acceptable weight values to $[0.05, 0.8]$, excluding the possibility of short-selling a member stock. We also estimated the probability P^- by which a portfolio is likely to under-perform the benchmark using the frequency rate T^-/T, where T^- is the number of trading days for which the portfolio's return is below benchmark's and T is the total number of observations in the estimation sample.

Optimal portfolio allocations were computed in the estimation sample, by solving the active optimisation problem described in section 2 for both active objectives and a range of portfolio cardinalities $K = \{2, 5, 10, 15, 20, 25, 30\}$. All optimisation algorithms were run assuming a comparable level of computational resources, using default parameter values suggested in the literature. The population size was set equal to 100 and the maximum number of generations (iterations) was 200. We also made 500 independent runs of each algorithms from random initial populations. Due to space limitations, the parameter setting for each technique is not presented here but is available from the author upon request.

The performance of nature-inspired techniques was compared against simpler heuristics for detecting good asset combinations. The first one was a Monte-Carlo portfolio selection technique, in which the optimal portfolio was selected by generating 2000 random asset combinations of fixed cardinality K, computing optimal weights for each combination by means of a gradient-search technique and picking the asset combination that maximises the portfolio's objective. An alternative stock picking method was adopted which makes use of fundamental stock analysis, looking, in particular, at the market capitalisation and the beta of each member stock.

Fig. 2 shows the maximum (in-sample) degree of objective fulfillment achieved by each portfolio-selection technique versus the total number of assets allowed in the portfolio. The line segments under the label $MC^{99\%}_{\text{worst}}$, ($MC^{99\%}_{\text{best}}$) represent the worst (best)-performing asset allocation detected by the Monte-Carlo simulation with 99% confidence[1]. The upper curve, which signifies the frontier of the shaded region, corresponds to the global optimum portfolio configurations, which were all indicated by nature-inspired optimisation heuristics[2]. As observed, evolutionary algorithms were by far superior as they managed to deliver much more acceptable portfolios at all cardinalities. On the other hand, optimal allocations based on size and beta were generally unsuccessful in meeting the objectives set by the fund manager. Fig. 2 signifies the importance of carefully exploring the space of feasible solutions when actively reproducing a benchmark. A relatively small, yet carefully chosen, portfolio can closer meet active objectives than larger portfolios of arbitrary elements. Note that in the estimation sample period under consideration, an optimal portfolio of $K = 5$ DJIA stocks can increase the overall degree of fulfilment up to 0.16, which is higher than that achieved by any capital allocation indicated by the Monte-Carlo or the fundamental stock-picking technique.

Table 1 gives further insight into the relative performance of nature-inspired optimisation schemes. It reports the percentage of runs for which each algorithm detected a portfolio allocation that is at worse (20%, 40%, 60%) far from the *global* optimum (the best-ever solution reported for each cardinality). A general remark about the results of Table 1 is that the probability of closely reaching the optimal region in a single run diminishes with the cardinality of the portfolio, which is indicative of the increasing complexity of the optimisation problem. Hence, as cardinality increases more restarts are needed to detect a good solution with high confidence, due to the fact that algorithms find it harder to convergence to a near-optimum region. The success rate quickly builds up at all cardinalities with an increasing range; note that GA and PSO can detect with more than 95% confidence a solution which at worse deviates 60% from

[1] We also present the median curve, which in Fig. 2 is identified with the $MC^{99\%}_{\text{worst}}$.

[2] In almost all cases, except for very low cardinalities, these are the portfolios selected by PSO. GA and SA identified portfolios with lower value for the objective function, however these portfolios were superior to either MC or fundamental stock-picking at all cardinalities. A summary about the relative performance of each method is given in Table 1; further details are available upon request.

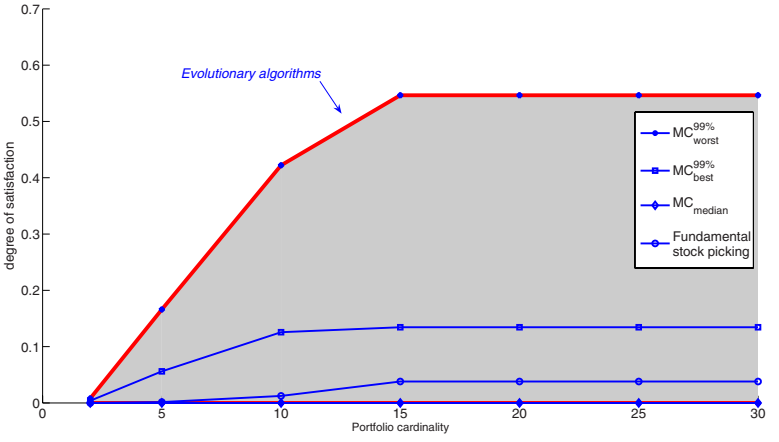


Fig. 2. The degree of overall goal attainment vs portfolio cardinality

Table 1. Empirical (%) probability of detecting a portfolio allocation within a range of (20%, 40%, 60%) of the global optimum

Cardinality	5	10	15	25	30
Simulated annealing	(6.4, 27.6, 78.4)	(0.6, 8.4, 46.8)	(0.0, 5.0, 38.0)	(0.0, 3.4, 39.2)	(0.0, 3.2, 37.2)
Genetic algorithm	(13.0, 67.8, 98.8)	(1.6, 50.6, 98.6)	(0.0, 20.4, 96.6)	(1.0, 23.4, 98.0)	(0.2, 24.0, 99.0)
Particle swarm optimisation	(30.4, 70.4, 98.8)	(12.2, 66.4, 98.8)	(3.0, 47.6, 97.8)	(3.0, 44.2, 98.0)	(3.2, 53.8, 98.8)

the global optimum. Among all heuristics considered in this study, PSO seems to be the most robust technique, as it manages to detect with higher frequency close-to-optimum allocations, independently of the initial state of the algorithm. SA has the lowest empirical success rate, which gets particulary worse with an increasing number of trading positions. This signifies the benefits from parallel exploration as opposed to trajectory-search techniques.

The financial performance of optimal active portfolios is analysed in Table 2. Each entry of the table is an average of the corresponding performance indicator over the whole range of cardinalities. The first (second) row associated with each method shows in-sample (out-of-sample) performance. All active strategies are evaluated in terms of the tracking error standard deviation (s_{TE}), the probability of under-performing the benchmark (P^-), the average portfolio return (m_P), the standard deviation of returns (s_P), the Sharpe ratio (SR), the Sortino ratio (SoR), the average excess return over the downside standard deviation (measured as the average of squared negative portfolio returns), and the cumulative return generated at the end of the investment period (CR). In the first three columns of the table, we report in addition the degree of fulfilment for the composite as well as for the two individual portfolio objectives (Obj1 refers to the constraint on s_{TE} and Obj2 to the restriction on the probability of shortfall). For comparison purposes, we also report in the last row of Table 2 performance measures for a

Table 2. Average (%) performance indicators of optimal portfolios

Portfolio selection method	Degree of fulfilment			s_{TE}	P^-	m_P	s_P	SR	SoR	CR
	Total	Obj1	Obj2							
Evolutionary	39.75	71.66	48.07	4.20	30.82	13.60	11.11	94.33	150.07	14.63
computation	1.28	65.44	1.59	4.61	46.98	8.58	11.09	49.96	86.11	8.99
Fundamental	2.37	46.41	3.90	6.17	44.99	11.07	9.92	79.82	11.76	
stock picking	0.01	55.11	0.43	5.33	49.37	2.54	9.96	-4.73	2.66	
Monte-Carlo	11.00	14.74	70.64	4.22	39.41	9.02	11.17	52.68	9.56	
	0.27	67.37	0.35	4.42	49.03	4.52	11.2	14.65	4.67	
Buy-and-hold	-	-	-	-	-	2.48	10.76	-4.82	-7.58	2.51
	-	-	-	-	-	4.28	10.05	12.70	20.89	4.37

buy-and-hold strategy placing equal amounts to all member stocks. To facilitate interpretation of results, we express figures as a percentage and on an annual basis (except for P^- which refers to the probability of shortfall between two consecutive trading days). All calculations of Sharpe ratio assume a constant risk-free rate of return equal to 3% per annum.

The overall picture of Table 2 indicates both in- and out-of-sample superiority of optimal portfolios suggested by evolutionary techniques. Such asset allocations manage to keep the tracking error std below 10%, the least desirable threshold, and also have a better control on the probability of shortfall, which in no case exceeds 50% as required by the fund manager. For these reasons, evolutionary algorithms attain the highest average degree of fulfilment for the investment objectives among all portfolio-selection techniques. Despite the fact that evolutionary portfolios are characterised by low tracking error, they manage to deliver above-market average and total return with similar-to-market risk, hence the major improvement in Sharpe and Sortino ratios. Note that although intelligent trading strategies managed to control the tracking error in both sample data sets (as seen by the s_{TE} column), they generally find it hard to outperform the market with high frequency in the last period (the probability of going below the market has on average increased from 30.82% in the estimation to 46.98% in the out-of-sample period). This is also evident from the shrinkage of the average spread between portfolio and market mean return, which ranged from 13.6-2.48=11.12% in the estimation to 8.58-4.28=4.30% in the out-of-sample period. The gradual deterioration in the investment performance - a common characteristic of all active strategies - could be attributed to the fact that portfolios are rebalanced only once per year, which makes it difficult to closely meet trading objectives.

6 Discussion - Further Research

In this paper, we analyse the construction of optimal portfolios for an investor who benchmarks his trading strategy against a market index. We deviate from the main trend in active asset management by considering non-standard objective functions, focusing on the risk faced by the investor when his portfolio

under-performs the market. These objectives can be effectively handled by combining properly defined fuzzy goals and constraints. We also examine the situation of index tracking using a subset of the assets available in the market. These special active formulations lead to optimisation problems of particular computational complexities, which can be effectively handled by means of intelligent heuristics, such as simulated annealing, particle swarm optimisation and genetic algorithms.

The results of our empirical study have important implications both for the computational properties of optimisation algorithms as well as the trading performance of the derived asset allocations. We show that in the sample period under consideration, passively holding DJIA stocks is an inefficient trading strategy and that an investor can benefit from straying away from the market. However, the ultimate success of the active investment scheme depends both on careful asset selection and optimal capital allocation. Evolutionary heuristics that drive the exploration of the solution space in both directions are more likely to detect optimal portfolio allocations with improved in- and out-of-sample performance. These tend to outperform both Monte-Carlo and simple expert rules of thumb based on fundamental stock analysis. Computation methods, such as genetic algorithms and particle swarm optimisation, are shown to be an effective tool for handling the inherent complexities of cardinality-constrained portfolio optimisation problems. The success of these methods can be attributed to their parallel exploration procedures and the stochastic elements embedded into the solution-search process. However, an undesired feature of the introduction of randomness is the large diversity in reported solutions (especially in complex landscapes), which calls for repeated runs before a near-optimum solution is reached with high confidence. An objective of this study was to bring out the inherent complexities of cardinality-constrained active formulations and also provide an empirical analysis of the stochastic properties of optimisation heuristics.

There are many directions in which the presented methodology could be developed in the future. First of all, it would be interesting to experiment with alternative active formulations, utilising different definitions of reward and risk. Searching for an optimal parameter setting that would boost algorithmic performance would also be a major step towards understanding how best to operate each heuristic and what to expect from it. Extensive comparisons with other optimisation techniques or simple rules of thumb for detecting good asset combinations, would shed further light on the complexities of the active optimisation programmes. In our future research plans is to benchmark active portfolios derived from intelligent heuristics against indexes with a much broader market coverage, such as the S&P 500 or Russell 3000.

References

1. Canakgoza, N., Beasley, J.: Mixed-integer programming approaches for index tracking and enhanced indexation. European Journal of Operational Research 196(1), 384–399 (2008)

2. Fang, Y., Lai, K.K., Wang, S.: Fuzzy Portfolio Optimization: Theory and Methods. Lecture Notes in Economics and Mathematical Systems, vol. 609. Springer, Heidelberg (2008)
3. Kennedy, J., Eberhart, R.C.: Particle swarm optimization. In: Proceedings of the IEEE International Conference on Neural Networks, Perth, Australia, pp. 12–13. IEEE Service Center, Piscataway (1995)
4. Kennedy, J., Eberhart, R.C., Shi, Y.: Swarm Intelligence. Morgan Kaufmann Publishers, San Francisco (2001)
5. Kirkpatrick, S., Gelatt, C.D., Vecchi, M.P.: Optimization by simulated annealing. Science 220, 671–680 (1983)
6. Maringer, D., Oyewumi, O.: Index tracking with constrained portfolios. Intelligent Systems in Accounting, Finance and Management 15, 57–71 (2007)
7. Mitchell, M.: An Introduction to Genetic Algorithms. MIT Press, Cambridge (1996)
8. Thomaidis, N.S., Angelidis, T., Vassiliadis, V., Dounias, G.: Active portfolio management with cardinality constraints: An application of particle swarm optimization. Journal of New Mathematical and Natural Computation 5(3) (2009); Special Issue on New Computational Methods for Financial Engineering

Evolutionary Monte Carlo Based Techniques for First Passage Time Problems in Credit Risk and Other Applications in Finance

Olena Tsviliuk[1], Roderick Melnik[2], and Di Zhang[2]

[1] OJSC Rodovid Bank, 1-3 Pivnychno-Syretska St, Kyiv 04136, Ukraine
[2] M[2]NeT Lab, Wilfrid Laurier University,
75 University Ave W, Waterloo, ON, Canada N2L 3C5

Abstract. Evolutionary computation techniques are closely connected with Monte Carlo simulations via statistical mechanics. Most practical realizations of such a connection are based on Markov chain Monte Carlo procedures and Markov chain approximation methodologies. However, such realizations face challenges when we have to deal with multivariate situations. In this contribution, we consider the development of evolutionary type Monte Carlo based algorithms for dealing with jump-diffusion stochastic processes. In particular, we focus on the first passage time problems for multivariate correlated jump-diffusion processes in the context of credit risk and the analysis of default correlations. The developed technique can be useful in option pricing as well as in other areas of complex systems analysis.

Keywords: Evolutionary Monte Carlo based techniques; Credit risk; Default correlations; Brownian bridge simulations; Complex systems; Multivariate jump-diffusion processes.

1 Introduction

Evolutionary computation techniques are common in dealing with various optimization, integration, and sampling problems in science and engineering where we attempt to explore the search space with a population that is with a multisubset of such a subset. The connection between evolutionary algorithms and Monte Carlo methods has been analyzed by a number of authors (e.g., [3,4,8]). It is quite natural to utilize this connection in the multivariate stochastic models where we deal the analysis of more than one statistical variable.

Evolutionary computation techniques usually include both selection and variation where during the selection we replicate an individual in the population based on selection probabilities, while their stochastic perturbations during this process are viewed as variation. The selection mechanism can be based on different distributions and corresponding algorithms including Boltzmann-Gibbs, Tsallis-Stariolo, Metropolis-Hasting and others [6,8]. One of the ways of integrating evolutionary techniques and Monte Carlo is to apply Evolutionary Markov

C. Di Chio et al. (Eds.): EvoApplications 2010, Part II, LNCS 6025, pp. 232–241, 2010.

Chain Monte Carlo (EMCMC) methodologies [4], deeply rooted in the Markov chain approximations [15].

Several application areas in data mining for EMCMC methods have been recently explored [8,5] and several evolutionary strategy algorithms based on sequential Monte Carlo simulations have been proposed [10]. The interest in this new development has been largely stimulated by the fact that standard MCMC methods can be trapped in a local mode indefinitely [9]. Attempts were made in the past to remedy this difficulty by developing multiple MCMC that can be run in parallel, each of which can, in principle, be characterized by different (yet related) distributions [7]. This followed by more recent interest to interacting MCMC algorithms [2] and by a number of interesting works directed to solving problems with complex multi-modal probability density landscapes, to the analysis of average properties of complex systems, as well as to the development of generic learning strategies [13,8,15,14,18].

This latter set of works inspired us to develop these ideas in the context of credit risk problems. There are two aspects that are intrinsic to the problems we are dealing with in this paper: (a) we deal with the first passage times (FPT) of stochastic processes with jumps, and (b) we are interested in the multivariate (and correlated) case.

2 First Passage Time in Credit Risk Models

Many problems in finance require the information on the FPT of a stochastic process. Mathematically, such problems are often reduced to the evaluation of the probability density of the time for such a process to cross a certain level, a boundary, or to enter a certain region. While in other areas of applications the FPT problem can often be solved analytically, in finance we usually have to resort to the application of numerical procedures, in particular when we deal with jump-diffusion stochastic processes (JDP).

Credit risk can be defined as the possibility of a loss occurring due to the financial failure to meet contractual debt obligations. This is one of the measures of the likelihood that a party will default on a financial agreement.

In structural credit-risk models, a default occurs when a company cannot meet its financial obligations, or in other words, when the firm's value falls below a certain threshold. One of the major problems in credit risk analysis is when a default occurs within a given time period and what is the default rate during such a time period. This problem can be reduced to a FPT problem, that can be formulated mathematically as a certain stochastic differential equation (SDE). It concerns the estimation of the probability density of the time for a random process to cross a specified threshold level. Therefore, it is natural that the FPT problem occurs also frequently in other areas of applications, including many branches of science and engineering [17,19].

An important phenomenon that we account for in our discussion here lies with the fact that, in the market economy, individual companies are inevitably linked together via dynamically changing economic conditions. Therefore, the

default events of companies are often correlated, especially in the same industry. Authors of [21] and [11] were the first to incorporate default correlation into the Black-Cox first passage structural model, but they have not included the jumps. As pointed out in [22] and [12], the standard Brownian motion model for market behavior falls short of explaining empirical observations of market returns and their underlying derivative prices. In the meantime, jump-diffusion processes have established themselves as a sound alternative to the standard Brownian motion model [1]. Multivariate jump-diffusion models provide a convenient framework for investigating default correlations with jumps and become more readily accepted in the financial world as an efficient modeling tool.

However, as soon as jumps are incorporated in the model, except for very basic applications where analytical solutions are available, for most practical cases we have to resort to numerical procedures. Examples of known analytical solutions include problems where the jump sizes are doubly exponential or exponentially distributed [12] as well as the jumps can have only nonnegative values (assuming that the crossing boundary is below the process starting value). For other situations, Monte Carlo methods remain a primary candidate for applications.

In a structural model, a firm i defaults when it can not meet its financial obligations, or in other words, when the firm assets value $V_i(t)$ falls below a threshold level $D_{V_i}(t)$. Generally speaking, finding the threshold level $D_{V_i}(t)$ is one of the challenges in using the structural methodology in credit risk modeling, since in reality firms often rearrange their liability structure when they have credit problems. In this contribution, we use an exponential form defining the threshold level $D_{V_i}(t) = \kappa_i \exp(\gamma_i t)$ as proposed by [21], where γ_i can be interpreted as the growth rate of the firm's liabilities. Coefficient κ_i captures the liability structure of the firm and is usually defined as the firm's short-term liability plus 50% of the firm's long-term liability. If we set $X_i(t) = \ln[V_i(t)]$, then the threshold of $X_i(t)$ is $D_i(t) = \gamma_i t + \ln(\kappa_i)$. Our main interest is in the process $X_i(t)$.

Prior to moving further, we define a default correlation that measures the strength of the default relationship between different firms. Take two firms i and j as an example, whose probabilities of default are P_i and P_j, respectively. Then the default correlation can be defined as

$$\rho_{ij} = \frac{P_{ij} - P_i P_j}{\sqrt{P_i(1-P_i)P_j(1-P_j)}}, \tag{1}$$

where P_{ij} is the probability of joint default. From Eq. (1) we have $P_{ij} = P_i P_j + \rho_{ij}\sqrt{P_i(1-P_i)P_j(1-P_j)}$. Let us assume that $P_i = P_j = 5\%$. If these two firms are independent, i.e., the default correlation equals zero, then the probability of joint default is $P_{ij} = 0.25\%$. If the two firms are positively correlated, for example, $\rho_{ij} = 0.4$, then the probability that both firms default becomes $P_{ij} = 2.15\%$ which is almost 10 times higher than in the former case. Thus, the default correlation ρ_{ij} plays a key role in the joint default with important implications in the field of credit analysis.

3 Multivariate Jump-Diffusion Processes and Monte Carlo Simulations

Although for jump-diffusion processes, the closed form solutions are usually unavailable, yet between each two jumps the process is, generally speaking, a Brownian bridge for a univariate jump-diffusion process. Authors of [1] have deduced the one-dimensional first passage time distribution for time period $[0, T]$. In order to evaluate multiple processes, we obtain multi-dimensional formulas and reduce them to computable forms.

Let us consider N_{firm} firms $\boldsymbol{X}_t = [X_1, X_2, ..., X_{N_{\text{firm}}}]^T$, each X_i describes the process of individual firm i. We expect that each process X_i satisfies the following SDE:

$$\begin{aligned} dX_i &= \mu_i dt + \sum_j \sigma_{ij} dW_j + dZ_i \\ &= \mu_i dt + \sigma_i d\tilde{W}_i + dZ_i, \end{aligned} \tag{2}$$

where $\tilde{W}_i$ is a standard Brownian motion and σ_i is:

$$\sigma_i = \sqrt{\sum_j \sigma_{ij}^2}.$$

We assume that in the interval $[0, T]$, the total number of jumps for firm i is M_i. Let the jump instants be $T_1, T_2, \cdots, T_{M_i}$. Let $T_0 = 0$ and $T_{M_i+1} = T$. The quantities τ_j equal interjump times, which are $T_j - T_{j-1}$. Following the notation of [1], let $X_i(T_j^-)$ be the process value immediately before the jth jump, and $X_i(T_j^+)$ be the process value immediately after the jth jump. The jump-size is $X_i(T_j^+) - X_i(T_j^-)$, and we can use such jump-sizes to generate $X_i(T_j^+)$ sequentially.

Let $A_i(t)$ be the event consisting of process X_i crossing the threshold level $D_i(t)$ for the first time in the interval $[t, t + dt]$, then the conditional interjump first passage density is defined as [1]:

$$g_{ij}(t) = P(A_i(t) \in dt | X_i(T_{j-1}^+), X_i(T_j^-)). \tag{3}$$

If we only consider one interval $[T_{j-1}, T_j]$, we can obtain

$$\begin{aligned} g_{ij}(t) = &\frac{X_i(T_{j-1}^+) - D_i(t)}{2y_i \pi \sigma_i^2} (t - T_{j-1})^{-\frac{3}{2}} (T_j - t)^{-\frac{1}{2}} \\ &* \exp\left(- \frac{[X_i(T_j^-) - D_i(t) - \mu_i(T_j - t)]^2}{2(T_j - t)\sigma_i^2} \right) \\ &* \exp\left(- \frac{[X_i(T_{j-1}^+) - D_i(t) + \mu_i(t - T_{j-1})]^2}{2(t - T_{j-1})\sigma_i^2} \right), \end{aligned} \tag{4}$$

where

$$y_i = \frac{1}{\sigma_i \sqrt{2\pi\tau_j}} \exp\left(-\frac{[X_i(T_{j-1}^+) - X_i(T_j^-) + \mu_i \tau_j]^2}{2\tau_j \sigma_i^2}\right).$$

After getting a result in one interval, we combine the results to obtain the density for the whole interval $[0, T]$. In a slight generalization, the process X_i may be viewed as a Brownian bridge $B(s)$ with $B(T_{j-1}^+) = X_i(T_{j-1}^+)$ and $B(T_j^-) = X_i(T_j^-)$ in the interval $[T_{j-1}, T_j]$, i.e. between each of the two successive jumps. If $X_i(T_j^-) > D_i(t)$, then the probability that the minimum of $B(s_i)$ is always above the boundary level is

$$P_{ij} = 1 - \exp\left(-\frac{2[X_i(T_{j-1}^+) - D_i(t)][X_i(T_j^-) - D_i(t)]}{\tau_j \sigma_i^2}\right), \tag{5}$$

and zero otherwise. The event "$B(s_i)$ is below the threshold level" means that the default happens or already happened, and its probability is $1 - P_{ij}$. Let $L(s_i) \equiv L_i$ denote the index of the interjump period in which the time s_i (first passage time) falls in $[T_{L_i-1}, T_{L_i}]$. Also, let I_i represent the index of the first jump, which happened in the simulated jump instant,

$$\begin{aligned} I_i = \min(j : X_i(T_k^-) > D_i(t); k = 1, \ldots, j, \text{ and} \\ X_i(T_k^+) > D_i(t); k = 1, \ldots, j-1, \text{ and} \\ X_i(T_j^+) \le D_i(t)). \end{aligned} \tag{6}$$

If no such I_i exists, then we set $I_i = 0$. By combining Eq. (4), (5) and (6), we get the probability of X_i crossing the boundary level in the whole interval $[0, T]$ as

$$\begin{aligned} &P(A_i(s_i) \in ds | X_i(T_{j-1}^+), X_i(T_j^-), j = 1, \ldots, M_i + 1) \\ &= \begin{cases} g_{iL_i}(s_i) \prod_{k=1}^{L_i - 1} P_{ik} & \text{if } L_i < I_i \text{ or } I_i = 0, \\ g_{iL_i}(s_i) \prod_{k=1}^{L_i - 1} P_{ik} + \prod_{k=1}^{L_i} P_{ik} \delta(s_i - T_{I_i}) & \text{if } L_i = I_i, \\ 0 & \text{if } L_i > I_i, \end{cases} \end{aligned} \tag{7}$$

where δ is the Dirac's delta function.

For firm i, after generating a series of first passage times s_i, we use a kernel density estimator with Gaussian kernel to estimate the first passage time density (FPTD) f. The kernel density estimator is based on centering a kernel function of a bandwidth (similar to [16,1]).

Next, we develop a procedure for generating beforejump and postjump values $X_i(T_j^-)$ and $X_i(T_j^+)$, respectively. Here $j = 1, \cdots, M$ where M is the total number of jumps for all the firms. We compute P_{ij} according to Eq. (5). To recur the first passage time density $f_i(t)$, we have to consider three possible cases that may occur for each non-default firm i:

1. **First passage happens inside the interval.** We know that if $X_i(T_{j-1}^+) > D_i(T_{j-1})$ and $X_i(T_j^-) < D_i(T_j)$, then the first passage happened in the time

interval $[T_{j-1}, T_j]$. To evaluate when the first passage happened, we introduce a new variable b_{ij} as $b_{ij} = \frac{T_j - T_{j-1}}{1 - P_{ij}}$. We generate several correlated uniform numbers Y_i by using the Sum-Of-Uniforms (SOU) method, then compute $s_i = b_{ij} Y_i + T_{j-1}$. If s_i belongs to interval $[T_{j-1}, T_j]$, then the first passage time occurred in this interval. We set $\texttt{IsDefault}(i) = 1$ to indicate that firm i has defaulted and compute the conditional boundary crossing density $g_{ij}(s_i)$ according to Eq. (4). To get the density for the entire interval $[0, T]$, we use $\widehat{f}_{i,n}(t) = \left(\frac{T_j - T_{j-1}}{1 - P_{ij}}\right) g_{ij}(s_i) * K(h_{opt}, t - s_i)$, where n is the iteration number of the Monte Carlo cycle.

2. **First passage does not happen in this interval.** If s_i does not belong to interval $[T_{j-1}, T_j]$, then the first passage time has not yet occurred in this interval.
3. **First passage happens at the right boundary of the interval.** If $X_i(T_j^+) < D_i(T_j)$ and $X_i(T_j^-) > D_i(T_j)$ (see Eq. (6)), then T_{I_i} is the first passage time and $I_i = j$, we evaluate the density function using kernel function $\widehat{f}_{i,n}(t) = K(h_{opt}, t - T_{I_i})$, and set $\texttt{IsDefault}(i) = 1$.

Next, we increase j and examine the next interval and analyze the above three cases for each non-default firm again. After running N times the Monte Carlo cycle, we get the FPTD of firm i as $\widehat{f}_i(t) = \frac{1}{N} \sum_{n=1}^{N} \widehat{f}_{i,n}(t)$.

In order to provide a reasonable credit analysis, we need to calibrate the developed model or, in other words, to numerically choose or optimize the parameters, such as drift, volatility and jumps to fit the most liquid market data. We have used the historical default data to optimize the parameters in the model based on the least-square methodology.

After Monte Carlo simulation we obtain the estimated density $\hat{f}_i(t)$ by using the kernel estimator method (with Gaussian kernel). The cumulative default rates for firm i in our model is defined as:

$$P_i(t) = \int_0^t \widehat{f}_i(\tau) d\tau, \tag{8}$$

where the kernel density estimator here is chosen from centering a kernel function of a bandwidth [16,20]. Recall that EMCMC combine Evolutionary Computation techniques and (often parallel) MCMC algorithms in order to design new algorithms for sampling or optimizing complex distribution functions. In a sense, we pursue here the same goal as we minimize the difference between our model and historical default data $\widetilde{A}_i(t)$ to obtain the optimized parameters in the model (such as σ_{ij}, arrival intensity λ in Eq. (2)):

$$\text{argmin} \left(\sum_i \sqrt{\sum_{t_j} \left(\frac{P_i(t_j) - \widetilde{A}_i(t_j)}{t_j} \right)^2} \right). \tag{9}$$

For example, theoretical (taken as in [21]) and simulated default correlations of two A-rated firms (A,A) for one-, two-, five, and ten-year periods would be

(0.00, 0.00), (0.02, 2.47), (1.65, 6.58), (7.75, 9.28), respectively. Note that simulated results here are obtained by the UNIForm sampling (UNIF) method as explained in the next section.

4 Density Functions, Default Rates, and Correlated Default

First, we considered a set of historical default data on differently rated firms (including the one presented in [21]) and described the FPTD functions and default rates of these firms. The optimized parameters (including optimal bandwidths) were obtained according to the procedure described in Section 3. Historical, theoretical (obtained with closed form solutions as in [21]), and simulated cumulative default rates for differently rated firms were compared. These results will be published elsewhere and one example of such a comparison can be found in [20].

However, here we focus on an example concerning the default correlation of two firms. If we do not include jumps in the model, the default correlation can easily be calculated. In Tables 1 and 2 we present comparisons of our results with those based on closed form solutions provided by [21] with $\rho = 0.4$. Next, let us consider the default correlations under the multivariate jump-diffusion processes. We use the following conditions in our multivariate UNIF method:

1. Setting $X_i(0) = 2$ and $\ln(\kappa_i) = 0$ for all firms.
2. Setting the growth rate of debt value equivalent to the growth rate of the firm's value: $\gamma_i = \mu_i$ and $\mu_i = -0.001$ for all firms.
3. Since we are considering two correlated firms, we choose $\boldsymbol{\sigma}$ as,

$$\boldsymbol{\sigma} = \begin{bmatrix} \sigma_{11} & \sigma_{12} \\ \sigma_{21} & \sigma_{22} \end{bmatrix}, \tag{10}$$

where $\boldsymbol{\sigma}\boldsymbol{\sigma}^\top = \boldsymbol{H}_0$ such that,

$$\boldsymbol{\sigma}\boldsymbol{\sigma}^\top = \boldsymbol{H}_0 = \begin{bmatrix} \sigma_1^2 & \rho_{12}\sigma_1\sigma_2 \\ \rho_{12}\sigma_1\sigma_2 & \sigma_2^2 \end{bmatrix},$$

and

$$\begin{cases} \sigma_1^2 = \sigma_{11}^2 + \sigma_{12}^2, \\ \sigma_2^2 = \sigma_{21}^2 + \sigma_{22}^2, \\ \rho_{12} = \dfrac{\sigma_{11}\sigma_{21} + \sigma_{12}\sigma_{22}}{\sigma_1\sigma_2}. \end{cases} \tag{11}$$

In Eq. (11), ρ_{12} reflects the correlation of the diffusion parts of the processes of the two firms. In order to compare with the standard Brownian motion and to evaluate the default correlations between different firms, we set all the $\rho_{12} = 0.4$ as in [21]. Furthermore, we use the optimized σ_1 and σ_2 for firm 1 and 2, respectively. Assuming $\sigma_{12} = 0$, we get,

$$\begin{cases} \sigma_{11} = \sigma_1, \\ \sigma_{12} = 0, \\ \sigma_{21} = \rho_{12}\sigma_2, \\ \sigma_{22} = \sqrt{1-\rho_{12}^2}\sigma_2. \end{cases}$$

4. The arrival rate for jumps satisfies the Poisson distribution with intensity parameter $\lambda = 0.1$ for all firms. The jump size is a normal distribution $Z_i \sim N(\mu_{Z_i}, \sigma_{Z_i})$, where μ_{Z_i} and σ_{Z_i} can be different for different firms to reflect specifics of the jump process for each firm. We adopt the optimized parameters.
5. As before, we generate the same interjump times $(T_j - T_{j-1})$ that satisfy an exponential distribution with mean value equal to 1 for each of the two firms.

We carry out the UNIF method to evaluate the default correlations via the following formula:

$$\rho_{12}(t) = \frac{1}{N}\sum_{n=1}^{N} \frac{P_{12,n}(t) - P_{1,n}(t)P_{2,n}(t)}{\sqrt{P_{1,n}(t)(1-P_{1,n}(t))P_{2,n}(t)(1-P_{2,n}(t))}}, \tag{12}$$

where $P_{12,n}(t)$ is the probability of joint default for firms 1 and 2 in each Monte Carlo cycle, $P_{1,n}(t)$ and $P_{2,n}(t)$ are the cumulative default rates of firm 1 and 2, respectively, in each Monte Carlo cycle.

The simulated default correlations for one and ten year periods are given in Tables 1 and 2, respectively. All the simulations were performed with the Monte Carlo runs $N = 500,000$. Comparing simulated default correlations with the theoretical data for standard Brownian motions, we can conclude that

Table 1. One year default correlations (%). All the simulations are performed with Monte Carlo runs $N = 500,000$.

	UNIF				[21]			
	A	Baa	Ba	B	A	Baa	Ba	B
A	-0.01				0.00			
Baa	-0.02	3.69			0.00	0.00		
Ba	2.37	4.95	19.75		0.00	0.01	1.32	
B	2.80	6.63	22.57	26.40	0.00	0.00	2.47	12.46

Table 2. Ten year default correlations (%). All the simulations are performed with the Monte Carlo runs $N = 500,000$.

	UNIF				[21]			
	A	Baa	Ba	B	A	Baa	Ba	B
A	8.79				7.75			
Baa	10.51	13.80			9.63	13.12		
Ba	9.87	14.23	22.50		9.48	14.98	22.51	
B	8.50	12.54	20.49	24.98	7.21	12.28	21.80	24.37

1. Similarly to conclusions of [21], the default correlations of same rated firms are usually large compared to differently rated firms. Furthermore, the default correlations tend to increase over long time and may converge to a stable value.
2. In our simulations, the one year default correlations of (A,A) and (A,Baa) are negative. This is because they seldom default jointly during one year. Note, however, that the default correlations of other firms are positive and usually larger than in the results presented by [21].
3. For two and five years, the default correlations of different firms increase. This can be explained by the fact that their individual first passage time density functions increase during these time periods, hence the probability of joint default increases.
4. As for ten year default correlations, our simulated results are almost identical to the theoretical data for standard Brownian motions. The differences are that the default correlations of (Ba,Ba), (Ba,B) and (B,B) decrease from the fifth year to tenth year in our simulations. The reason is that the first passage time density functions of Ba- and B-rated firms begin to decrease from the fifth year, hence the probability of joint default may increase slowly.

5 Conclusion

In this contribution, we have analyzed the credit risk problems of multiple correlated firms in a structural model framework, where we incorporated jumps to reflect the external shocks or other unpredicted events. By combining the fast Monte-Carlo method for one-dimensional jump-diffusion processes and the generation of correlated multidimensional variates, we have developed a fast evolutionary type Monte-Carlo type procedure for the analysis of multivariate and correlated jump-diffusion processes. The developed approach generalizes previously discussed non-correlated jump-diffusion cases for multivariate and correlated jump-diffusion processes. Finally, we have applied the developed technique to analyze the default events of multiple correlated firms via a set of historical default data. The developed methodology provides an efficient computational technique that is applicable in other areas of credit risk and for the pricing of options.

References

1. Atiya, A.F., Metwally, S.A.K.: Efficient Estimation of First Passage Time Density Function for Jump-Diffusion Processes. SIAM Journal on Scientific Computing 26, 1760–1775 (2005)
2. Campillo, F., Rakotozafy, R., Rossi, V.: Parallel and interacting Markov chain Monte Carlo algorithm. Mathematics and Computers in Simulation 79, 3424–3433 (2009)
3. Cercueil, A., Francois, O.: Monte Carlo Simulation and Population-Based Optimization. In: Int. Congress on Evolutionary Computation, CEC 2001, pp. 191–198. IEEE Press, Los Alamitos (2001)

4. Drugan, M.M., Thierens, D.: Evolutionary Markov chain Monte Carlo. In: Liardet, P., Collet, P., Fonlupt, C., Lutton, E., Schoenauer, M. (eds.) EA 2003. LNCS, vol. 2936, pp. 63–76. Springer, Heidelberg (2004)
5. Dugan, N., Erkoc, S.: Genetic algorithm-Monte Carlo hybrid geometry optimization method for atomic clusters. Computational Materials Science 45(1), 127–132 (2009)
6. Dukkipati, A., Bhatnagar, S., Murty, M.N.: Gelfand-Yaglom-Perez theorem for generalized relative entropy functionals. Information Sciences 177(24), 5707–5714 (2007)
7. Geyer, C.J.: Practical Markov Chain Monte Carlo. Statistical Science 7, 473–483 (1992)
8. Goswami, G., Liu, J.S., Wong, W.H.: Evolutionary Monte Carlo methods for clustering. J. of Computational and Graphical Statistics 16(4), 855–876 (2007)
9. Goswami, G., Liu, J.S.: On learning strategies for evolutionary Monte Carlo. Statistics and Computing 17(1), 23–38 (2007)
10. Johansson, A.M., Lehmann, E.A.: Evolutionary Optimization of Dynamics Models in Sequential Monte Carlo Target Tracking. IEEE Trans. on Evolutionary Computation 13(4), 879–894 (2009)
11. Hull, J., White, A.: Valuing Credit Default Swaps II: Modeling Default Correlations. Journal of Derivatives 8, 12–22 (2001)
12. Kou, S.G., Wang, H.: First passage times of a jump diffusion process. Adv. Appl. Probab. 35, 504–531 (2003)
13. Laskey, K.B., Myers, J.W.: Population Markov Chain Monte Carlo. Machine Learning 50(1-2), 175–196 (2003)
14. Melnik, R.V.N.: Coupling control and human factors in mathematical models of complex systems. Engin. Appl. of Artificial Intelligence 22(3), 351–362 (2009)
15. Melnik, R.V.N.: Markov chain network training and conservation law approximations: Linking microscopic and macroscopic models for evolution. Applied Mathematics and Computation 199(1), 315–333 (2008)
16. Silverman, B.W.: Density Estimation for Statistics and Data Analysis. Chapman & Hall, London (1986)
17. Song, J.H., Kiureghian, A.D.: Joint First-Passage Probability and Reliability of Systems under Stochastic Excitation. Journal of Engineering Mechanics 132(1), 65–77 (2006)
18. Vrugt, J.A., et al.: Accelerating Markov Chain Monte Carlo Simulation by Differential Evolution with Self-Adaptive Randomized Subspace Sampling. Int. J. of Nonlinear Sciences and Numer. Simul. 10(3), 273–290 (2009)
19. Wang, J., Zhang, W.-J., Liang, J.-R., et al.: Fractional nonlinear diffusion equation and first passage time. Physica A 387(1), 764–772 (2008)
20. Zhang, D., Melnik, R.V.N.: First passage time for multivariate jump-diffusion processes in finance and other areas of applications. Applied Stochastic Models in Business and Industry 25(5), 565–582 (2009)
21. Zhou, C.: An analysis of default correlation and multiple defaults. Review of Financial Studies 14, 555–576 (2001)
22. Zhou, C.: The Term Structure of Credit Spreads with Jump Risk. Journal of Banking and Finance 25, 2015–2040 (2001)

Calibrating the Heston Model with Differential Evolution

Manfred Gilli and Enrico Schumann*

University of Geneva
Department of Econometrics

Abstract. Calibrating option pricing models to market prices often leads to optimisation problems to which standard methods (like such based on gradients) cannot be applied. We investigate one particular example, Heston's stochastic volatility model. We discuss how to price options under this model, and how to calibrate the parameters of the model with a heuristic technique, Differential Evolution.

1 Introduction

Implied volatilities obtained by inverting the Black–Scholes–Merton (BSM) model vary systematically with strike and maturity; this relationship is called the volatility surface. Different strategies are possible for incorporating this surface into a model. We can accept that volatility is not constant across strikes and maturities, and directly model the volatility surface and its evolution. This approach is not consistent since we assume that a single underlier has different volatilities, but still, it is the approach that is mostly used in practice. An alternative is to model the option prices such that the BSM-volatility surface is obtained, for instance by including jumps, locally varying volatility, or by making volatility stochastic; a well-known example for the latter approach is the Heston model [1]. It is the model that we look at in this paper.

As so often in finance, the success of the BSM-model stems not so much from its empirical quality, but from its computational convenience. This convenience comes in two flavours. Firstly, there are closed-form pricing equations (the Gaussian distribution function is not available analytically, but fast and precise approximations exist). Secondly, calibrating the model requires only one parameter to be determined, the volatility, which can be computed from market prices readily with Newton's method or another zero-finding technique. For the Heston-model, both tasks become more difficult: pricing requires numerical integration, and calibration requires to find (in a riskneutral world) five parameters instead of only one for BSM.

* This paper is short version of our paper "Calibrating Option Pricing Models with Heuristics". Both authors gratefully acknowledge financial support from the EU Commission through MRTN-CT-2006-034270 COMISEF; they would like to thank Benoît Guilleminot for many discussions on the subject.

C. Di Chio et al. (Eds.): EvoApplications 2010, Part II, LNCS 6025, pp. 242–250, 2010.

In this short paper, we will look into the calibration of the Heston model. Finding parameters that make the model consistent with market prices means solving a non-convex optimisation problem. We suggest to use optimisation heuristics, more specifically we show that Differential Evolution is able to give good solutions to the calibration problem. The paper is structured as follows: In Section 2 we discuss how to price options under the Heston model. Fast pricing routines are important since the suggested heuristic is computationally intensive; hence to obtain calibration results in a reasonable period of time, we need to be able to evaluate the objective function (which requires pricing) speedily. Section 3 details how to implement the heuristic for a calibration problem. Section 4 concludes.

2 Pricing with the Characteristic Function

There are several generic approaches to price options. The essence of BSM is a no-arbitrage argument which leads to a partial differential equation that can be solved numerically (in the particular case of BSM, even analytically). A more recent approach builds on the characteristic function of the (log) stock price. European options can be priced by the following equation ([3, 4]):

$$C_0 = \mathrm{e}^{-q\tau} S_0 \Pi_1 - \mathrm{e}^{-r\tau} X \Pi_2 \tag{1}$$

where C_0 is the call price today (time 0), S_0 is the spot price of the underlier, and X is the strike price; r and q are the riskfree rate and dividend yield; time to expiration is denoted τ. The Π_j are calculated as

$$\Pi_1 = \frac{1}{2} + \frac{1}{\pi} \int_0^\infty \mathsf{Re}\left(\frac{\mathrm{e}^{-\mathrm{i}\omega \log(X)} \phi(\omega - \mathrm{i})}{\mathrm{i}\omega \phi(-\mathrm{i})} \right) \mathrm{d}\omega \,, \tag{2a}$$

$$\Pi_2 = \frac{1}{2} + \frac{1}{\pi} \int_0^\infty \mathsf{Re}\left(\frac{\mathrm{e}^{-\mathrm{i}\omega \log(X)} \phi(\omega)}{\mathrm{i}\omega} \right) \mathrm{d}\omega \,. \tag{2b}$$

The symbol ϕ stands for the characteristic function of the log stock price; the function $\mathsf{Re}(\cdot)$ returns the real part of a complex number. For a given ϕ we can compute Π_1 and Π_2 by numerical integration, and hence obtain option prices from Equation (1).

2.1 Black–Scholes–Merton

In a BSM-world, the stock price S_t under the risk-neutral measure follows

$$\mathrm{d}S_t = rS_t \mathrm{d}t + \sqrt{v} S_t \mathrm{d}z$$

where r is the riskfree rate, and z is a Wiener process. The volatility $\sqrt{v}$ is constant. The well-known pricing formula for the BSM-call is given by

$$C_0 = \mathrm{e}^{-q\tau} S_0 \,\mathrm{N}(d_1) - X\mathrm{e}^{-r\tau} \,\mathrm{N}(d_2) \tag{3}$$

with

$$d_1 = \frac{1}{\sqrt{v\tau}}\left(\log\left(\frac{S_0}{X}\right) + \left(r - q + \frac{v}{2}\right)\tau\right) \tag{4a}$$

$$d_2 = \frac{1}{\sqrt{v\tau}}\left(\log\left(\frac{S_0}{X}\right) + \left(r - q - \frac{v}{2}\right)\tau\right) = d_1 - \sqrt{v\tau} \tag{4b}$$

and $\mathrm{N}(\cdot)$ the Gaussian distribution function.

Given the dynamics of S, the log-price $s_\tau = \log(S_\tau)$ follows a Gaussian distribution with $s_\tau \sim \mathcal{N}\big(s_0 + \tau(r - \frac{1}{2}v), \tau v\big)$, where s_0 is the natural logarithm of the current spot price. The characteristic function of s_τ is given by

$$\begin{aligned}\phi_{\mathrm{BSM}}(\omega) &= \mathsf{E}(\mathrm{e}^{\mathrm{i}\omega s_\tau}) \\ &= \mathrm{e}^{\mathrm{i}\omega\left(s_0 + \tau(r - \frac{1}{2}v)\right) + \frac{1}{2}\mathrm{i}^2\omega^2 v} \\ &= \mathrm{e}^{\mathrm{i}\omega s_0 + \mathrm{i}\omega\tau r - \frac{1}{2}(\mathrm{i}\omega + \omega^2)\tau v}\,. \end{aligned} \tag{5}$$

With a continuous dividend yield q, we replace r by $r - q$. With (5) inserted into (2), Equation (1) should, up to numerical precision, give the same result as Equation (3).

2.2 The Heston Model

Under the Heston model the stock price S and its variance v are described by

$$\begin{aligned}\mathrm{d}S_t &= rS_t\mathrm{d}t + \sqrt{v_t}S_t\mathrm{d}z^{(1)} \\ \mathrm{d}v_t &= \kappa(\theta - v_t)\mathrm{d}t + \sigma\sqrt{v_t}\mathrm{d}z^{(2)}\,.\end{aligned}$$

The long-run variance is denoted θ, mean reversion speed is κ and σ is the volatility of volatility. The Wiener processes have correlation ρ. For $\sigma \to 0$, the Heston dynamics approach those of BSM. The characteristic function of the log-price in the Heston model is as follows, see [5].

$$\phi_{\mathrm{Heston}} = \mathrm{e}^A \times \mathrm{e}^B \times \mathrm{e}^C \tag{6}$$

$$\begin{aligned}A &= \mathrm{i}\omega(\log S_0 + (r - q)\tau) \\ B &= \frac{\theta\kappa}{\sigma^2}\left((\kappa - \rho\sigma\mathrm{i}\omega - d)\tau - 2\log\left(\frac{1 - g_2\mathrm{e}^{-d\tau}}{1 - g_2}\right)\right) \\ C &= \frac{\frac{v_0}{\sigma^2}\Big(\kappa - \rho\sigma\mathrm{i}\omega - d\Big)\Big(1 - \mathrm{e}^{-d\tau}\Big)}{1 - g_2\mathrm{e}^{-d\tau}}\end{aligned}$$

$$\begin{aligned}d &= \sqrt{(\rho\sigma\mathrm{i}\omega - \kappa)^2 + \sigma^2(\mathrm{i}\omega + \omega^2)} \\ g_2 &= \frac{\kappa - \rho\sigma\mathrm{i}\omega - d}{\kappa - \rho\sigma\mathrm{i}\omega + d}\end{aligned}$$

2.3 Integration Schemes

We use a Gauss–Legendre rule, see [6], [7]; we experimented with alternatives like Gauss–Lobatto as well, but no integration scheme was clearly dominant over another. We do not use adaptive algorithms, but compute a fixed number of nodes and weights, and evaluate the integrals in Equations (2). To test our pricing algorithms, we first investigate the BSM-case. Here we compare results from Equation (3) with results from Equation (1). The cutoff point for the integrals in Equations (2) is set to 200. Figure 1 shows the relative pricing errors as compared with `blsprice` (from Matlab's Financial Toolbox), with 20 nodes (left) and 100 nodes (right). Note that here we are already pricing a whole matrix of options (different strikes, different maturities). This matrix is taken from the experiments described in the next section. Already with 100 nodes the pricing errors are in the range of 10^{-13}, ie, practically zero. Several Matlab programs can be downloaded from `http://comisef.eu`.

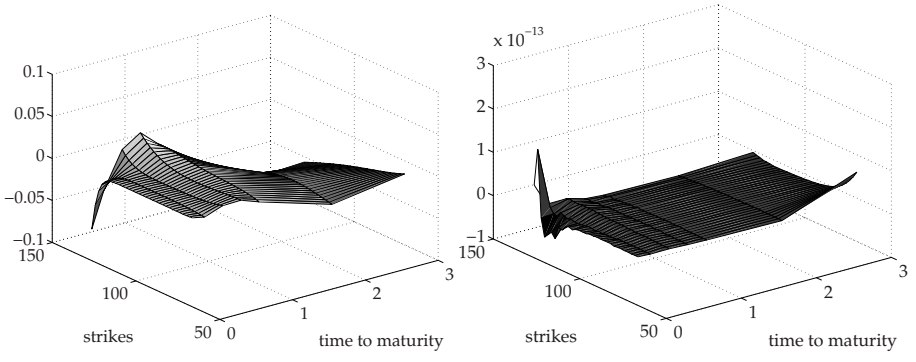


Fig. 1. Relative pricing errors compared with analytical BSM: a Gauss-rule with 20 nodes (left) and 100 nodes (right)

3 Calibrating the Model Parameters

Calibrating an option pricing model means to find parameters such that the model's prices are consistent with market prices, leading to an optimisation problem of the form

$$\min \sum_{i=1}^{M} \frac{\left|C_i^{\text{model}} - C_i^{\text{market}}\right|}{C_i^{\text{market}}} \tag{7}$$

where M is the number of market prices. Alternatively, we could specify relative deviations, use squares instead of absolute values, or introduce weighting schemes. The choice of the objective function depends on the application at hand; ultimately, it is an empirical question to determine a good objective function.

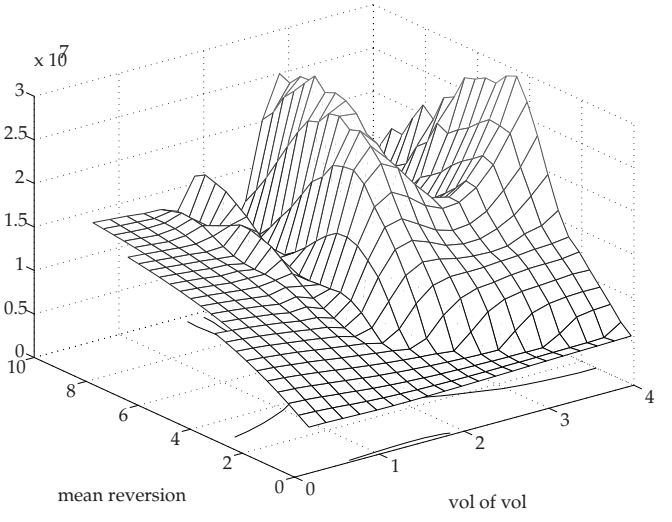


Fig. 2. A search space for the Heston model

Since here we are interested in numerical aspects, we will use specification (7). The problem is not convex, and hence standard methods (eg, based on derivatives of the objective function) may fail. Figure 2 shows the objective function when varying two parameters (mean reversion κ and volatility-of-volatility σ) while holding the others fixed. Hence, we deploy a heuristic method, Differential Evolution, to solve problem (7).

When we evaluate (7), we price not just one option, but a whole array (different strikes, different maturities). But for a given set of parameters that describe the underlying process of the model, the characteristic function ϕ only depends on the time to maturity, not on the strike price. This suggests that speed improvements can be achieved by preprocessing those terms of ϕ that are constant for a given maturity, and then compute the prices for all strikes for this maturity, see [8] for a discussion, see Algorithm 1 for a summary.

Algorithm 1. Computing the prices for a given surface.

```
set parameters, set T = maturities, set X = strikes
for τ ∈ T do
    compute characteristic function φ
    for X ∈ X do
        compute price for strike X, maturity τ
    end for
end for
compute objective function
```

3.1 Differential Evolution

Differential Evolution (DE) is described in detail in [9]; Algorithm 2 summarises the technique. DE evolves a population P of n_P solutions, stored in real-valued vectors of length p (where $p = 5$ for the Heston model). In every iteration (or 'generation'), the algorithm creates a new candidate solution (a rival) for each existing solution. This candidate solution is constructed by taking the difference between two other solutions, weighting this difference by a factor F, and adding the weighted difference to a third solution (Statement 7 in Algorithm 2). Then an element-wise crossover (with probability CR) takes place between this auxiliary solution and the original solution, see Statements 8–10 (ζ is a random number that is uniformly distributed between 0 and 1). This final candidate solution is then evaluated (the objective function is denoted Φ). If it is better than the original solution, it replaces it; if not, the old solution is kept. The search stops after n_G generations (other stopping criteria are possible).

Algorithm 2. Differential Evolution.

1: initialise parameters n_P, n_G, F and CR
2: initialise population $P^{(1)}_{j,i}$, $j = 1, \ldots, p$, $i = 1, \ldots, n_P$
3: **for** $k = 1$ to n_G **do**
4: $\quad P^{(0)} = P^{(1)}$
5: $\quad$ **for** $i = 1$ to n_P **do**
6: $\quad\quad$ generate ℓ_1, ℓ_2, $\ell_3 \in \{1, \ldots, n_P\}$, $\ell_1 \neq \ell_2 \neq \ell_3 \neq i$
7: $\quad\quad$ compute $P^{(v)}_{\cdot,i} = P^{(0)}_{\cdot,\ell_1} + \mathtt{F} \times (P^{(0)}_{\cdot,\ell_2} - P^{(0)}_{\cdot,\ell_3})$
8: $\quad\quad$ **for** $j = 1$ to p **do**
9: $\quad\quad\quad$ **if** $\zeta_j < \mathtt{CR}$ **then** $P^{(u)}_{j,i} = P^{(v)}_{j,i}$ **else** $P^{(u)}_{j,i} = P^{(0)}_{j,i}$
10: $\quad\quad$ **end for**
11: $\quad\quad$ **if** $\Phi(P^{(u)}_{\cdot,i}) < \Phi(P^{(0)}_{\cdot,i})$ **then** $P^{(1)}_{\cdot,i} = P^{(u)}_{\cdot,i}$ **else** $P^{(1)}_{\cdot,i} = P^{(0)}_{\cdot,i}$
12: $\quad$ **end for**
13: **end for**

3.2 Calibrating the Heston Model

We create artificial data sets to test DE. The spot price S_0 is 100, the riskfree rate r is 2%, there are no dividends. We compute prices for strikes X from 70 to 130 (in steps of size 2), and maturities τ of 1/12, 3/12, 6/12, 9/12, 1, 2 and 3 years. Hence our surface comprises $31 \times 7 = 217$ prices. The parameters for the Heston model come from the following table:

$\sqrt{v_0}$	0.3	0.3	0.3	0.3	0.4	0.2	0.5	0.6	0.7	0.8
$\sqrt{\theta}$	0.3	0.3	0.2	0.2	0.2	0.4	0.5	0.3	0.3	0.3
ρ	-0.3	-0.7	-0.9	0.0	-0.5	-0.5	0.0	-0.5	-0.5	-0.5
κ	2.0	0.2	3.0	3.0	0.2	0.2	0.5	3.0	2.0	1.0
σ	1.5	1.0	0.5	0.5	0.8	0.8	3.0	1.0	1.0	1.0

Given a set of parameters, we compute option prices and store them as the true prices. Then we run DE 10 times to solve problem (7) and see if we can recover the parameters; the setup implies that a perfect fit is possible.

DE is coded in Matlab following Algorithm 2. We ran a number of preliminary experiments to find reasonable parameter values for these algorithms. The `F`-parameter should be set to around 0.3–0.5 (we use 0.5); very low or high values typically impaired performance. The `CR`-parameter had less influence, but levels close to unity worked best (ie, each new candidate solution is likely changed in many dimensions). The stopping criterion is a fixed number of function evaluations (population size × generations); we run three settings,

$$\begin{aligned} 5\,000 \quad & (50\times 100)\,, \\ 20\,000 \quad & (100\times 200)\,, \\ 45\,000 \quad & (150\times 300)\,. \end{aligned}$$

On an Intel P8700 single core at 2.53GHz with 2 GB of RAM, a single run takes about 20, 80, and 180 seconds, respectively. (Each evaluation of the objective function involves the pricing of 217 options, hence for the last setting, in one run we compute $217 \times 150 \times 300 = 9\,765\,000$ prices.)

We also run a gradient search (steepest descent) and a direct search (Nelder–Mead), using the Matlab implementations `fminunc` and `fminsearch`. Starting values were chosen randomly from the ranges:

	min	max
$\sqrt{v_0}$	0.05	1.0
$\sqrt{\theta}$	0.05	1.0
ρ	-1.00	1.00
κ	0.01	5.00
σ	0.01	5.00

(in the same way the initial population for DE was set up). Like for DE, we conduct 10 restarts for each set of parameters.

This is not an entirely fair comparison: steepest descent and direct search are not appropriate for non-convex problems . Neither did we spend much time tuning the algorithms; or finding out why they failed when they did. Accordingly, one reviewer of this paper argued that in particular `fminunc` would be 'a relatively weak benchmark'. Yes, but then the results still reflect what is done in both professional and academic practice.

For each run we store the value for the objective function (the mean percentage error; Equation (7)), and corresponding parameter estimates for which we compute absolute errors, ie,

$$\text{error} \;=\; |\;\text{estimated parameter} \;-\; \text{true parameter}\;|\,.$$

Results for unconstrained optimisation runs are given below in Figure 3. For each level of function evaluations, we have 100 results (10 parameters sets times 10 restarts). The panels show the distributions of the parameter errors and the objective function (pooled over all parameter sets); optimal values would be zero.

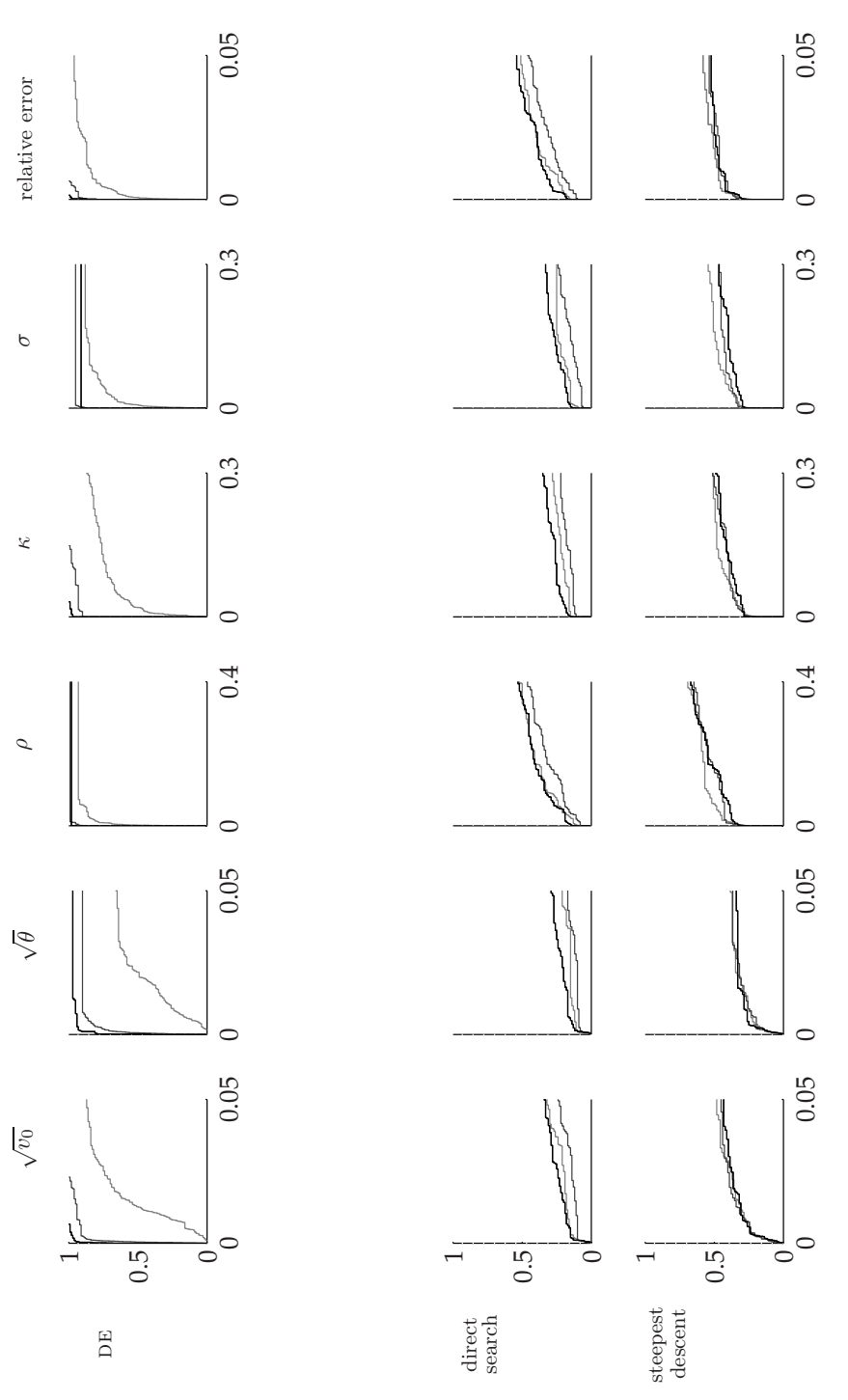


Fig. 3. Distributions of absolute errors

The increasing number of function evaluations is marked by an increasingly dark grey (ie, 5 000, 20 000, 45 000). DE works well, with error distributions converging to zero, even though there are cases where it fails to recover the true parameters. Both steepest descent and direct search fare worse; steepest descent in particular returned `NaN` in between 10 to 20% of cases. All results come from unconstrained optimisation runs: in some cases, steepest descent and direct search converged to economically meaningless or impossible values (eg, correlations outside the range from -1 to 1, or negative variances).

4 Conclusion

In this paper we have investigated option pricing under Heston's stochastic volatility model. We have shown how to calibrate the parameters of the model with a heuristic technique, Differential Evolution, and presented evidence that such techniques can provide very good solutions to the calibration problem.

References

[1] Heston, S.L.: A Closed-Form Solution for Options with Stochastic Volatility with Applications to Bonds and Currency options. Review of Financial Studies 6(2), 327–343 (1993)
[2] Carr, P., Madan, D.B.: Option Valuation Using the Fast Fourier Transform. Journal of Computational Finance 2(4), 61–73 (1999)
[3] Bakshi, G., Madan, D.B.: Spanning and Derivative-Security Valuation. Journal of Financial Economics 55(2), 205–238 (2000)
[4] Schoutens, W.: Lévy Processes in Finance: Pricing Financial Derivatives. Wiley, Chichester (2003)
[5] Albrecher, H., Mayer, P., Schoutens, W., Tistaert, J.: The Little Heston Trap., Wilmott, January 2007, pp. 83–92 (2007)
[6] Davis, P.J., Rabinowitz, P.: Methods of Numerical Integration, 2nd edn. Dover, New York (2007)
[7] Trefethen, L.N.: Is Gauss Quadrature Better than Clenshaw–Curtis? SIAM Review 50(1), 67–87 (2008)
[8] Kilin, F.: Accelerating the Calibration of Stochastic Volatility Models. Centre for Practical Quantitative Finance Working Paper Series, vol. 6 (2007)
[9] Storn, R.M., Price, K.V.: Differential Evolution – a Simple and Efficient Heuristic for Global Optimization over Continuous Spaces. Journal of Global Optimization 11(4), 341–359 (1997)

Evolving Trading Rule-Based Policies

Robert Gregory Bradley[1,2], Anthony Brabazon[1,2], and Michael O'Neill[1,3]

[1] Natural Computing Research and Applications Group
University College Dublin, Ireland
[2] School of Business, University College Dublin, Ireland
[3] School of Computer Science and Informatics, University College Dublin, Ireland
robert.bradley@ucdconnect.ie, anthony.brabazon@ucd.ie, m.oneill@ucd.ie

Abstract. Trading-rule representation is an important factor to consider when designing a quantitative trading system. This study implements a trading strategy as a rule-based policy. The result is an intuitive human-readable format which allows for seamless integration of domain knowledge. The components of a policy are specified and represented as a set of rewrite rules in a context-free grammar. These rewrite rules define how the components can be legally assembled. Thus, strategies derived from the grammar are well-formed, domain-specific, solutions. A grammar-based Evolutionary Algorithm, Grammatical Evolution (GE), is then employed to automatically evolve intra-day trading strategies for the U.S. Stock Market. The GE methodology managed to discover profitable rules with realistic transaction costs included. The paper concludes with a number of suggestions for future work.

1 Introduction

A rule-based trading system typically relies on a series of conditional rules to make trading decisions. This is in contrast to a discretionary system which primarily relies on human judgement. Discretionary traders function without explicitly quantified rules and instead act on mental rules which are developed through experience. An obvious problem with this approach is that the decision-making process can be skewed by human emotions such as fear and greed [7]. Although there are successful traders who function exclusively using these techniques, there are many advantages to quantifying one's rules. Quantifying a strategy and representing the logic in computer code allows for system automation, speeding up the decision-making process, and for the statistical analysis of the trading strategy via backtesting. In addition, it facilitates the application of advanced machine learning techniques in order to optimise system parameters. In this study we employ an evolutionary algorithm called Grammatical Evolution (GE) [8] to automatically evolve profitable trading models. GE has previously been applied to for-ex trading [1] and to stock market trading [2] . These studies evolve expressions which evaluate to real-numbers, and rules are formulated by comparing these values to threshold values in order to generate a trading entry signal. The exit strategies employed are static which is a simplification of

C. Di Chio et al. (Eds.): EvoApplications 2010, Part II, LNCS 6025, pp. 251–260, 2010.

the process of real-world trading. In contrast, this study structures a grammar such that a trading strategy is comprised of entry and exit rules which sensibly combine technical indictors resulting in a more intuitive and comprehensive representation of a trading strategy.

1.1 Structure of Paper

The remainder of this paper is structured as follows. Section 2 outlines the construction of a rule-based policy as applied to the problem of trading-rule design. Section 3 illustrates how a set of rewrite rules can govern the steps in creating a well-formed, domain-specific policy. In section 4, Grammatical Evolution is employed to heuristically navigate the search space of possible trading rules in order to find ones which are high-quality. The results of our experiments are presented in section 5. The last section outlines the conclusions of the study and suggests a number of avenues of future work.

2 Rule-Based Policies

Representation is a key factor when designing a trading strategy. One intuitive approach is to represent a strategy as a *rule-based policy*. This approach has previously been used to develop automated agents for a well known arcade game [6], and more generally provides a framework which encapsulates many real-world decision scenarios. A *rule-based policy* is a set of rules with the following structure

IF [Condition] holds, THEN do [Action]

The policy also includes logic to decide which rule in the set should be executed given a particular state of the environment. Rule-based policies are a particularly useful representation for financial trading rules as they are human readable and it is a straightforward task to embed domain knowledge. In applying these policies we need to specify four things:

1. What are the possible actions?
2. What are the possible conditions and how are they constructed from observations?
3. How to make rules from conditions and actions
4. How to combine the rules into policies

We will address each of these in turn.

For the purpose of our study we are limiting the system to five basic actions as outlined in Table 1 below: *EnterLong, ExitLong, EnterShort, ExitShort and DoNothing*. On the close of each time interval (one minute intervals in the case of our experiments), a trading agent is confronted with the problem of deciding which of these actions to take.

Table 1. The above table shows the list of actions which are used in the construction of trading rules

Action	Description
EnterLong	Open a long position
ExitLong	Close long position
EnterShort	Open a short position
ExitShort	Close short position
DoNothing	Take no action

An action is taken if the condition associated with the executed rule holds true. The market position is updated accordingly, as per Table 2. In the case where conditions belonging to multiple rules are satisfied simultaneously, the policy must include logic to prioritize one action over another.

Table 2. This table shows the market position state changes resulting from a sequence of actions

Action	DoNothing	EnterLong	ExitLong	EnterShort	ExitShort
Position	Flat	Long	Flat	Short	Flat

A trading rule condition is a Boolean expression of observations and comparison operators. The system designer decides on the set of observations, where members of the set could range from a simple closing price to a sophisticated statistical metric. In this study we have limited our observation set to four technical indicators (see Table 3). The length of this list is arbitrary, and can be extended with any number of variables, indicators, and analytics.

Table 3. This list of technical indicators serve as the building blocks in the construction of trading rule conditions

Analytic	Description
SMA	simple moving average which gauges momentum
WMA	weighted moving average which gauges momentum
STOC	Stochastic indicator which gauges overbought/oversold levels
ADX	ADX indicator gauges strength of trend

A trading strategy is then constructed by logically combining one or more rules of the form IF [Condition] holds, THEN do [Action]. Each of the actions listed in Table 1 have a rule which decides whether or not the action should be taken based on a boolean condition. The policy includes logic which decides which rule should be executed given the current market position [9]. For example, if we are already long then it makes no sense to execute the *ExitShort* rule. This feedback loop from the environment to the policy produces more logical trading decisions.

3 Grammatical Representation

A context-free grammar is a set of one or more rewrite rules of the form $NT- > T$, where NT is a nonterminal which maps to one or more terminals and/or nonterminals. This study employs a metasyntax called Backus Naur Form (BNF) to express the rewrite rules. We define a root nonterminal $< policy >$, see Fig. 1, which is mapped to the policy framework discussed in Section 2.

```
<Policy>      ::=          if(Long){
                                 if(<LongExitCondition>){
                                       ExitLong;
                                 }
                           }
                           else if(Short){
                                 if(ShortExitCondition){
                                       ExitShort;
                                 }
                           }
                           else if(Flat){
                                 if(<LongEntryCondition>){
                                       EnterLong;
                                 }
                                 else if(<ShortEntryCondition>){
                                       EnterShort;
                                 }
                           }
```

Fig. 1. The root nonterminal maps to the policy framework which includes terminals and non terminals

Rewrite rules are also added to define how conditions may be constructed from observations and operators. This allows us to incorporate our domain knowledge into the grammar, resulting in a set of rules that governs how a well-formed policy can be assembled.

```
<condition>        ::=  <ma><greatless><ma>
                      | <stochastic> <greatless> <threshold>
                      | (<condition>\&\&<condition>)
 <maindicator>     ::=  SMA(<number>)
                      | WMA(<number>)
<oscillator>       ::= STOC(<number>)
                      | ADX(<number>)
```

Fig. 2. A partial BNF grammar showing three production rules used to create well-formed conditions

We embed our domain knowledge in the grammar by defining rewrite rules which ensure that sensible conditions are created. For example, it would not be sensible to directly compare a moving average and a stochastic indicator, which oscillates between 0 and 100. Thus, we create separate nonterminals for each type of indicator. This inhibits ill-formed conditions such as $(SMA(10) > STOC(65))$ being created. Fig. 2 shows the production rules used to build conditions. A more

comprehensive set of observations might require additional domain knowledge to be included in the grammar.

Digit concatenation [3] can also be applied to the creation of constants within a strategy. Uses of constants in trading rules include for example, the parameterisation of technical indicators, and the creation of appropriate thresholds for (as an example) oscillator indicators. We define rewrite rules to control the range of values which a constant can take. The integer parameter passed to technical indicators specifies the number of intervals over which the indicator will be calculated. Domain knowledge is embedded in our rewrite rules so that this parameter is limited to an appropriate range of values. For example, in determining the number of periods over which a technical indication can 'look back' we need to consider the likely trading frequency. In this paper, we focus on high-frequency trading and we limit the range of this parameter to 2000, which is about 5 days of trading given a 1 minute frequency. Therefore, the production rules in Fig. 3 below allow for the generation of integers in the range [1,1999].

```
<number>        ::=   <1-9>
                    | <1-9><0-9>
                    | <1-9><0-9><0-9>
<1-9>           ::=   1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9
<0-9>           ::=   0 | <1-9>
```

Fig. 3. The production rules which define how digits can be concatenated to create an indicator look-back parameter constant

Similarly, a set of rewrite rules are defined to govern how digits may be concatenated to create the threshold constant. This threshold is compared to the oscillator indicators, for example ($STOC(55) > 80$). Domain knowledge is embedded in these production rules to limit this constant to be in the range [0,95] in increments of 5. A more complex grammar might define rewrite rules for a number of other constants which are used in a policy, with domain-specific knowledge applied to each constant if necessary.

4 Evolution of Trading Policies

Rule-based trading policies derived from our context-free grammar are guaranteed to be well-formed solutions. However, we are still faced with the challenge of deriving a successful trading strategy. A huge number of different strategies can be derived depending on the derivation sequence executed, and hence we need to traverse this search space in an efficient manner. To do this we adopt an evolutionary approach.

The GE algorithm was inspired by the genotype to phenotype mapping process in biology. This process involves the mapping of DNA to proteins. In the case of GE, integer strings drive the selection of rewrite rules from our grammar which results in a mapped policy.

4.1 Data Review

The dataset, see Fig. 4, used in this study is comprised of 200 trading days of 1 minute bars for the CAT stock which is listed on the NYSE. Normal trading hours on the NYSE are 9.30 to 16:00 EST, resulting in 390 minutes of trading per session, producing a dataset of 78,000 samples. Each bar contains a price for the open, high, low, and closing trades for that minute. The dataset is partitioned to produce an in-sample section from Monday 2007-01-22 to Friday 2007-04-20, and an out-of-sample section which ranges from Monday 2007-04-23 to Friday 2007-07-20. The first partition is used to train a population of policies, and the second to test the best individuals out-of-sample.

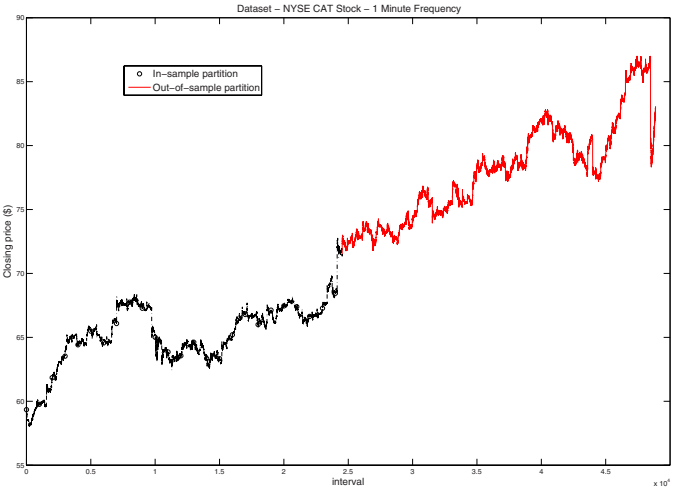


Fig. 4. Dataset of CAT high frequency data

4.2 Methodology

The results for the data-mining of trading policies using GE were averaged over 30 separately-seeded runs in order to allow us to assess the statistical significance of the performance metrics. A single run involves an in-sample training phase and an out-of-sample testing phase. Phase one trains a GE population of 300 individuals for 50 generations. On each generation of the algorithm, each integer string in the population is mapped to a trading policy. The in-sample dataset is then iterated from start to end. The trading policy is executed at the close of each interval and a trading action from the set listed in Table 1 is signaled. This action is then applied to the open of the next interval. For example if an *EnterLong* signal is generated on the close of interval 2007-01-22 09:56 a new trade is initiated on the open of the next interval 2007-01-22 09:57. Signals are only accepted up to and including the second last interval of the day. On the last interval of each day any open position (long or short) is closed and the market

position is set to flat. This ensures no positions are held overnight. The running return of the strategy is stored at each interval.

On reaching the end of the training set the vector of returns is analyzed to determine the performance of the individual. The performance metric used in this paper is an information ratio where the average daily return minus transaction costs is divided by the standard deviation. This metric is a risk-adjusted measure which favors strategies with stable daily returns. The calculated ratio is the fitness of the strategy being evaluated. On completion of a generation, each strategy in the population has a fitness score as calculated above. Roulette wheel selection is employed with a steady-state replacement strategy, see Goldberg [5]. The population is evolved for 50 generations and the best trading policy is then tested on the out-of-sample dataset to assess the robustness of the strategy on unseen data. The experimental parameters used for this study are listed in Table 4.

Table 4. Experimental Parameters

Parameter	Value
Pop size	300
Mutation	.001
Crossover	.9
Generations	50
Runs	30
Selection	Roulette wheel
Replacement	Steady-state

5 Results

The results of our experiments are now presented. A population of 300 individuals was trained on partition one of the dataset discussed in Section 4.1 for 50 generations. The best individual from the population was then tested out-of-sample on partition two. Thirty separately seeded runs were carried out.

Table 5 shows the performance of the best individual against a number of benchmarks, averaged over 30 runs. The average best policy, not including transaction costs, returned 57.83% annualized in-sample. The standard deviation of

Table 5. Statistics derived from the annualized percentage return of the best policy against a number of benchmarks, averaged over 30 runs

	In-sample			Out-of-sample		
	Mean	Std Dev	Info-ratio	Mean	Std Dev	Info-ratio
Best policy	57.83	32.85	1.76	13.85	30.96	0.45
Best policy - costs	51.56	29.55	1.75	9.41	30.96	0.30
Random agent	-6.07	35.32	-0.17	2.27	30.23	0.08
Random agent - costs	-354.21	35.13	-10.08	-287.05	29.37	-9.77
Buy only	0.19	NA	NA	19.94	NA	NA
Buy only - costs	-0.28	NA	NA	19.13	NA	NA

this level over the runs was 32.85%. The mean return dropped to 13.85% out-of-sample. When transaction costs were included the in-sample mean return dropped to 51.56%, while the out-of-sample performance fell to 9.41%.

Two benchmarks were employed. The first, a zero-intelligence agent, executes an action from the set{*Buy, Sell, Do Nothing*} with equal probability on the close of each interval. Due to the uniformity of the sampling distribution this random strategy will complete a trade every 4 intervals on average. This results in a large volume of trades making transaction costs a significant factor. The average best policy fails to significantly out-perform the random agent when costs are not included, however with transactions costs of 1 cent per share included the performance of the zero-intelligence strategy is drastically reduced. We note that a zero-intelligence agent with a high trading frequency (about 100 trades a day) might not be a realistic benchmark when compared to an evolved policy trading at a potentially much lower frequency.

The second benchmark in our experiment is an intra-day buy-and-hold strategy. Each day this agent initiates a long position at the market open and goes flat at the market close. The best evolved individual significantly outperforms this benchmark in-sample. The opposite is true out-of-sample. The benchmark's superior performance over the second partition is due to the aggressively bullish trend in the second half of our dataset, see Fig. 4. Although the benchmark has superior performance the risk profile is very different to that of the evolved strategies. The buy and hold strategy is exposed to the systematic risk of the market 390 minutes a day. A typical evolved policy is in the market a lot less than this. One such policy is described below.

5.1 Example Evolved Policy

This section takes a closer look at a profitable policy evolved using Grammatical Evolution. Fig. 1 shows the root nonterminal in our BNF grammar which maps to a basic template. The policy template is comprised of a number of rules, and the rules' conditions are represented by nonterminals in our grammar. GE is used to evolve these conditions using the rewrite rules defined in Fig. 2. An example evolved trading policy is shown in Fig. 5. The result is human-readable, and can be easily visualized in any standard technical indicator charting software.

The logic of the policy framework favors long positions over short as the *EnterLong* condition is checked before the *EnterShort* condition. The complete dataset used in this study is relatively bullish and the rule in Fig. 5 has essentially switched off short trades by evolving a condition for the *EnterShort* rule which will always evaluate to false as the ADX indicator is not likely to breach the 65 level. With the *EnterShort* rule out of the picture the *EnterLong* condition is checked at each interval to find a good entry point. This policy made 67 trades in-sample, and 44 out-of-sample, see Table 6. The average trade duration was approximately 60 minutes across the two partitions with a standard deviation of 65 minutes. The table also shows the mean return and the standard deviation of the return on these trades in basis points.

```
if(Long){
      if(((STOCHFD_TA(3,46)>40)&&((STOCHFD_TA(5,92)>55)&&(ADX_TA(92)<40)))){
            ExitLong;
      }
else if(Short){
      if(((ADX_TA(48)>85)&&(ADX_TA(719)>85))){
            ExitShort
      }
else if(Flat){
      if((((((STOCHFD_TA(63,69)<60)&&((STOCHFD_TA(1,321)>5)&&(ADX_TA(21)<80)))&&
      (((STOCHFD_TA(5,50)>5)&&(ADX_TA(95)<65))&&(STOCHFD_TA(759,8)<30)))&&
      ((WMA_TA(74)<SMA_TA(28))&&(SMA_TA(581)>SMA_TA(1))))&&(STOCHFD_TA(2,2)<30))){
            EnterLong;
      }
      else if((ADX_TA(578)>65)) {
            EnterShort;
      }
}
```

Fig. 5. Example policy evolved by Grammatical Evolution

Table 6. Trade duration (in minutes) and return (in basis points) statistics for the example evolved policy

	In-sample		**Out-of-sample**	
	Mean	Std Dev	Mean	Std DEv
Trade duration (mins)	45	52	75	80
Return (bps)	10.32	23.47	19.82	36.78
Return with costs (bps)	8.78	23.47	18.50	36.78

6 Conclusion and Future Work

In this study Grammatical Evolution was used to evolve well-formed trading rule-based policies for a large-cap U.S. stock. A policy is derived from a grammatical representation of the components which make up a policy. Despite the fact that we limited the system to a very simple set of building blocks, GE managed to uncover some profitable rules when realistic transaction costs were included.

In spite of the promise of these results, no absolutely definitive conclusions can be drawn from a set of experiments based on a single stock over a single time period. We intend to pursue a number of avenues to extend this work. For example, in this study the best policy from the population trained over a 3 month period is traded out-of-sample for the next 3 months. This is a conservative approach as it is ambitious to expect a simple technical trading rule to yield robust results over such a large window. A moving window approach, where the training set is incremented periodically and the population is trained for a number of generations at each increment, has been shown to yield superior returns [4] over a static approach, like the one adopted in this study. We intend to investigate this approach in more detail. We also intend to analyze the phenotypic characteristics of the system during evolution to give greater transparency into the distribution of intelligence inherent in the population.

References

1. Brabazon, A., O'Neill, M.: Evolving technical trading rules for spot foreign-exchange markets using grammatical evolution. Computational Management Science 1(3), 311–327 (2004)
2. Brabazon, A., O'Neill, M.: Intra-day trading using grammatical evolution. In: Brabazon, A., O'Neill, M. (eds.) Biologically Inspired Algorithms for Financial Modelling, pp. 203–210. Springer, Berlin (2006)
3. Dempsey, I., O'Neill, M., Brabazon, A.: Grammatical constant creation. In: Deb, K., et al. (eds.) GECCO 2004. LNCS, vol. 3103, pp. 447–458. Springer, Heidelberg (2004)
4. Dempsey, I., O'Neill, M., Brabazon, A.: Adaptive trading with grammatical evolution. In: Proceedings of the 2006 IEEE Congress on Evolutionary Computation, pp. 9137–9142. IEEE Press, Los Alamitos (2006), http://ncra.ucd.ie/papers/cec2006_adaptiveTrading.ps (Vancouver July 6-21, 2006)
5. Goldberg, D.E.: The Design of Innovation: Lessons from and for Competent Genetic Algorithms. Kluwer Academic Publishers, Dordrecht (2002)
6. Istvan, S., András, L.: Learning to play using low-complexity rule-based policies: Illustrations through ms. pac-man. J. Artif. Intell. Res. (JAIR) 30, 659–684 (2007), http://www.jair.org/media/2368/live-2368-3623-jair.pdf
7. Lo, A.W.: The adaptive markets hypothesis: Market efficiency from an evolutionary perspective. Journal of Portfolio Management (2004) (forthcoming)
8. O'Neill, M., Ryan, C.: Grammatical Evolution. Kluwer, Dordrecht (2003)
9. Saks, P., Maringer, D.: Evolutionary Money Management. In: Giacobini, M., et al. (eds.) EvoWorkshops 2009. LNCS, vol. 5484, pp. 162–171. Springer, Heidelberg (2009)

Evolving Artistic Styles through Visual Dialogues

Jae C. Oh[1] and Edward Zajec[2]

[1] Department of Electrical Engineering and Computer Science
[2] Department of Transmedia, Visual Performing Arts
Syracuse University, Syracuse, NY 13214, U.S.A.
{jcoh,ezajec}@syr.edu

Abstract. We describe an interactive art that can facilitate visual dialogues with a user and reveal the user's compositional proclivity. The system invites the user for a visual dialogue, initially presenting two compositions in distinct styles. Then, the user "converses" with the system by responding. As the interactions continue, the user starts to experience a tendency evolving in the compositions–much the same way as in a dialogue between two people. Throughout the interactions, the evolutionary process brings out a certain sense of self-reflection to the user. The product of the dialogue is an experience in self-reflection in visual proclivity. I3 was exhibited to the public in two international art forums.

1 Introduction

We hypothesize that each of us possesses an innate sense of how we prefer to organize things in the world, although we may not be fully aware of it. We describe a program that can guide the user through a search space of possible visual composition styles; in the end the user can be led to a set of compositional styles that represents the user's stylistic tendency. As Collomosse [4] considers painting a search process, we consider finding compositional styles to be search.

Our program is based on genetic algorithms (GAs) [8] and a novel composition analyzer based on gestalt theory [15] and perceptual grouping [12]. Each 'chromosome' in our system represents a visual composition style that can generate compositions in a certain artistic style. This system, called I3, can evolve chromosomes that capture the essence of a user's composition style through interactions with the user.

An interesting aspect of I3 is that the user is a part of the fitness evaluation function. The user gives feedback to the system by choosing the favorite among a set of compositions at each iteration. This is somewhat related to the idea of interactive fitness [3]. However, in I3, the system also evaluates compositions using a novel algorithm that computes an aesthetic measure. Therefore, the fitness function, $F(c)$, of chromosome, c, is defined as $F(c) = \alpha \times f_{system} + (1 - \alpha) \times f_{user}$, where f_{system} and f_{user} are feedbacks from the system and the user, respectively, and α is a weighting factor. Given the user is coherent in choosing compositions, in the end, the composition styles will converge to similar styles. Note that, however, the consistency is not forced because any dialogue can diverge to multiple topics and such a dialogue can be meaningful as well.

C. Di Chio et al. (Eds.): EvoApplications 2010, Part II, LNCS 6025, pp. 261–270, 2010.

A dialogue with I3 will bring the user an experience–a sense of self-reflection. Therefore, our intent is not to automatically generate art, or to validate an aesthetic theory on the basis of visual image processing [9], but to design a system of visual dialogues, which would provide users an aesthetic experience through self-reflection. It is important to note that the compositions are means to bring out such an experience of self-reflection and our primary goal is less in generating artwork but more in facilitating a unique user's experience.

GAs are often known to evolve unpredictable creative solutions. We see compositions generated by I3 often being quite artistically creative and unique while still exhibiting a coherent style. In many cases, the styles evolved seem different but a close observation reveals qualitative similarities.

I3 was shown at International Centre of Graphic Arts in Ljubljana and at the 14th International Festival of Computer Arts (MFRU) in Maribor, Slovenia. We have collected the sessions data of the gallery visitors for offline studies.

This paper is organized as follows: Section 2 presents a background and the motivation for I3. Section 3 describes I3. Section 3.2 presents the algorithms used in I3 and example interactions. In Section 3.3, we discuss an analyzer that discovers gestalt qualities from a composition. These qualities are used to compute the aesthetic measure for the composition. Section 4 discusses the experience obtained through I3 interactions and the data collected from the Ljubljana exhibit. Finally, Section 5 presents a summary and future work.

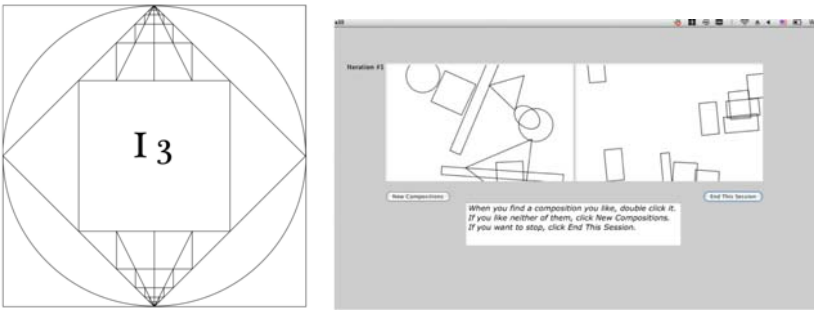
2 Background and Motivation

Can computers facilitate self-reflection? If so, how can we design a software system that can promote self-reflection? What are the roles of Fine Arts in promoting self-reflection? This project is an effort to answer these questions.

McCormack [10] classifies research in evolutionary music and art in two categories. One of them is research that explores creativity. Many researchers agree that creativity is one of the products of an increased self-awareness. Hofstadter argues that self-awareness is a prerequisite for creativity [7].

We attempt to bring self-awareness to the users by facilitating interactions with an evolutionary visual composition system. There are several efforts on generating aesthetic artworks; for example, in [11] and [13], the primary goals are mostly to generate artwork. Boden et. al. [2] discusses interactive generative art. We believe that an evolutionary process of compositional styles, combined with a proper user feedback mechanism, can foster self-awareness to the user through the creative exchanges between the user and the system. In this regard, our work is unique. Note that I3 still attempts to generate artistically pleasing compositions while keeping a sense of dialogical flow with the user.

I3 has two major components: the *generator* and the *analyzer* components. The generator component uses GAs, probabilistic methods, and interactions with the user to generate compositions. The analyzer component evaluates the compositions by using gestalt perceptual grouping and an aesthetic measure based on Birkhoff [1]. The interactions among the generator, the analyzer, and the user bring out an emergence of a sense of coherent experience to the user, and



(a) *Initial I3 Screen* **(b)** *An example of initial compositions*

Fig. 1. I3 intro screen and initiating a visual dialogue

Require: A set of compositional genes, C^j at generation j, $0 \leq j \in Integer$; A user response, $c_{selected}$
Ensure: A composition with the style specified by c_i.
  repeat
    Generate two chromosomes, c_0 and c_1 both in C^0;
    Produce_Compositions (c_0, c_1);
  until $c_{Selected} = \{c_0 \vee c_1 | c_i \in C^0 \wedge i \in Integer\}$
  repeat
    Apply GA($c_{Selected}$); Generate nine chromosomes, $c_0 \ldots c_9$;
    Produce_Compositions ($c_0, \ldots, c_9$);
    $c_{Selected} = \{c_0 \vee c_2 \vee \ldots \vee c_9 | c_i \in C^j \wedge i \in Integer\}$
    Evaluate each chromosome
  until User Quits

Algorithm 1. *The top-level algorithm of I3*

in some way to the system as well. A detailed discussion of various aesthetic measures is beyond the scope of the paper. Readers should consult Greenfield [5] and Hoenig [6] for good historical overviews on aesthetic measures.

3 A Description of I3

3.1 The Top-Level Behavior

Figure 1(a) shows the initial screen of I3. The design is inspired by the Socratic dialogue in Plato's Meno [14] between Socrates and a slave about finding the square that is exactly twice as large in its area of the small one. The screen shows that how a conversation between two people can be accomplished visually.

The user can click on any part of the intro screen to proceed to an instruction screen. After the instruction screen, the program runs Algorithm 1, the main-loop. The algorithm creates two initial chromosomes, c_0 and c_1, which represent distinct composition styles, using some gestalt rules (See Section 3.3). Then it

Fig. 2. Nine compositions are produced after the user selected the second composition in the first screen. The bottom of the screen is a history panel. The pop-up window gives information about each composition and it does not appear from the third iteration.

calls *Produce_Compositions*(c_0, c_1) to generate two compositions as in Figure 1(b). This function in turn calls Algorithm 2.

These two compositions are possible "topics" of the dialogue to ensue and the user is asked to choose one of them. Based on the choice, in the second repeat loop, the algorithm invokes an evolutionary algorithm to generate a set of nine chromosomes–representing nine compositional styles. Figure 2 shows the nine compositions generated after the user choosing the composition on the right. The user then is again to select one among the nine by focusing on the personal preference. This main-loop continues until the user stops. Given the user is reasonably coherent, nine chromosomes converge to the user's visual proclivity.

I3 generates compositions consisting three fundamental shapes–the triangle, the square, and the circle. All unnecessary details–shading, texture, and color–are stripped away, so that the overall structure, once discovered, can be more clearly apprehended. Each shape has properties such as *position, size, structure,* and *orientation.* The dialogue develops by having the users to rate a set of nine compositions for each interaction. The challenge for the users is to gain an ever deeper awareness of the interrelation, linking the type of compositions emerging in front of their eyes with the self-reflective dialogical process.

This is much like interviewing an unknown person. As the interview progresses, various subjects may be discussed, yet a coherent theme may arise. Throughout the interview, both the interviewer and the interviewee will gain an experience about their conversations. We attempt to capture such an experience with I3. The user will be more aware of their visual composition proclivities as the program reflects its own. As in any dialogues, digression can happen anytime. Focused conversationalist may be able to digress less from the main topic yet unfocused and casual attitudes may introduce more digressions. The program may digress and become less focused if the user is. On the other hand, if the user is focused, a clear theme of the dialogue emerges in front of the user's eyes. A certain amount of digression is allowed but if the conversation becomes

Require: A chromosome c, (i.e., style of composition) described above.
Ensure: A composition with the style specified by c.
totalShapes = $\mathcal{N}(nr, \sigma^2)$
for (i = 0; i < totalShapes; i++) **do**
shape = decideWhichShape(c);
position = decidePosition(shape, c)
shape = geometricTransformation (shape,c)
addShapeToCanvas (shape, position)
end for

Algorithm 2. *The Composition Algorithm; it is called by* `Produce_Compositions(C)`*, where* `C` *is a set of chromosomes (See Algorithm 1); $\mathcal{N}(nr, \sigma^2)$ is a normal distribution with $\mu = nr$ and σ is a standard deviation.*

too unfocused, it may come out of some kind confusing interactions–as it can happen in a conversation between two people.

3.2 The Generator Component

The knowledge in I3 is represented in chromosomes. Each chromosome represents a compositional style with a unique tendency of generating the three fundamental shapes discussed above. A chromosome consists of: the *nr* gene for how many shapes to be drawn; the *position* gene for where on the canvas the i^{th} shape to be drawn; and the *trans* gene for what geometric transformations will be applied to the shape. The position and transformation genes are associated with their own repetition probabilities, which decide the probabilities of repeating the tendencies of the i^{th} shape for drawing the $(i+1)^{th}$ shape.

`Produce_Compositions(C)` calls Algorithm 2, iteratively for each element c_i in `C`, to generate a composition for c_i, where `C` is a set of chromosomes. Given a chromosome, after deciding how many shapes to draw using the **nr** gene, the algorithm draws the first shape on the canvas based on the position gene. Then the i^{th} shape is generated based on the $(i-1)^{th}$ shape using the repetition probabilities of the chromosome in position, shape, and geometric transformations.

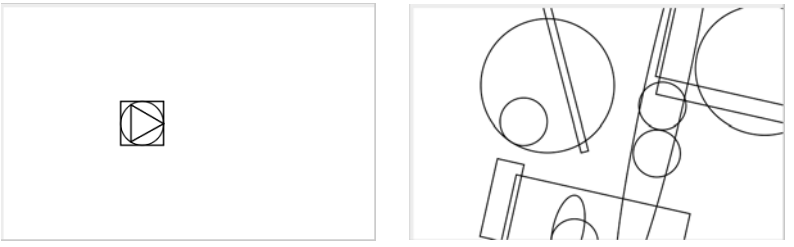
Figure 3(a) shows a rather contrived example but explains the concept clearly. This chromosome always draws three shapes at the same location with all three basic shapes. On the other hand, Figure 3(b) shows a typical composition generated by the algorithm.

We have designed the fitness function to consider both the user's feedback as well as the aesthetic measure computed by the system. The following function evaluates each chromosome c's fitness value:

$$f^c = \alpha \times f^c_{system} + (1 - \alpha) \times f^c_{\text{user}} \tag{1}$$

where f^c_{system} and f^c_{user} are feedbacks from the system and the user, respectively, and α is a weighting factor ranging between 0 and 1.

$f^c_{\text{user}} = 0$, if the composition created by c was not selected by the user in the current generation and $f^c_{user} = r$, where $r > 0$, if c's composition was



(a) *A contrived I3 composition* **(b)** *A typical composition*

Fig. 3. Example compositions generated

selected by the user. f^c_{system} is the fitness value computed by the system. The criteria for the system fitness is the aesthetics of the composition generated by c. f^c_{system} evaluates each composition based on the principle of *unity in variety* as in Birkhoff [1] and a perceptual grouping algorithm that finds gestalt tendencies [12]. Section 3.3 discusses f_{system} in more detail.

With the fitness for each chromosome, the evolutionary algorithms will generate the next generation of chromosomes. The new generation will be used for the next iteration of the repeat-loop. This process continues until the user quits.

3.3 The Analyzer Component

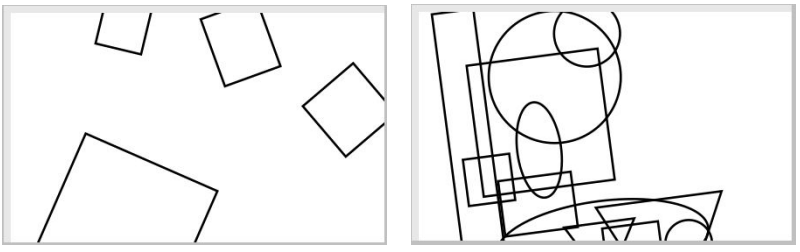
We designed an analyzer based on gestalt theory [15] and perceptual grouping [12] to evaluate the aesthetic quality of compositions generated.

The most basic law of gestalt is the law of proximity. It explains that, all things being equal, two objects that are close together will be grouped together. This law will interact with the law of similarity, that states that two objects that are similar will be grouped together. Of course two objects can be similar according to many dimensions including size, shape, color, and orientation. Proximity could also be argued to be the similarity of position, depending upon if we consider position to be an innate property like size and shape.

A more advanced law is the law of good-continuation, which states that objects that are arranged in linear or curve-linear formations are grouped together. The law of good continuation also says that our minds fill in gaps when objects are in lines or they can be connected by line-dashes to form an entire line.

Thorisson [12] developed an algorithm to group visual objects along the laws of proximity and similarity. We improved the Thorisson's algorithm to build a parse tree for a given composition after finding gestalt properties embedded in the composition. The parse tree is annotated with gestalt properties discovered from the composition.

Now we explain how f_{system} is computed. Birkhoff [1] defines aesthetic measure as a function of $\frac{Order(O)}{Complexity(C)}$. We define order O be the measure computed by the perceptual grouping. That is, order O is a function of proximity, similarity, and in turn, of good-continuation. Our algorithm, given a composition,



(a) *aesthetic measure = 0* **(b)** *aesthetic measure = 0.88*

Fig. 4. Two example compositions with their aesthetic measures

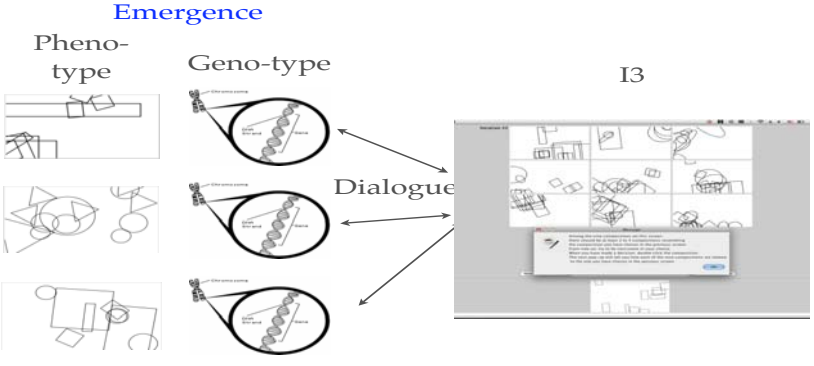


Fig. 5. Emergence of Styles via I3 interactions

finds the number of good-continuation properties in the composition and evaluates each good-continuation property based on proximity and similarity; then it returns a value between 0 and 1.

Complexity C is a function of the number of shapes, the number of different shape sizes, and the variety of different shapes in a composition. A composition with more different sizes and shapes has a higher C. C is also between 0 and 1.

Figures 4-(a) and (b) show two example compositions with their aesthetic measures 0 and 0.88, respectively. A higher aesthetic value indicates a good balance of order O and complexity C. The measure works acceptably in most cases but it can be improved by considering more gestalt properties in the analyzer and combining ideas from different aesthetic measures. We are studying a way to incorporate other gestalt properties to the aesthetic measure.

4 I3 Experience

In this section, we discuss I3's emergent behavior and the behavioral data collected from the I3 exhibit in Ljubljana.

Figure 5 shows the concept of emergence of compositional styles in I3. A genotype in I3 represents a compositional style and the corresponding phenotype

(a) *An example at i = 53* **(b)** *Another example at i = 14*

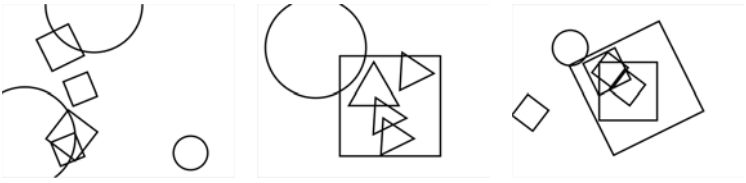
Fig. 6. Two separate I3 sessions converging to unique styles

is a visual composition, a manifestations of the genotype. It is important to remember that a genotype represents a style of composition not a composition itself. I3, using the same genotype, will generate various different compositions, yet they will be similar in styles. The user can start self-reflecting about his or her own compositional styles as a coherent artistic style emerges.

As shown in Figure 6, after enough interactions (i.e., generations), the nine compositions can converge to similar compositions that share common characteristics. The number of interactions required for a convergence depends on the user's coherence to a specific style of composition in selecting, as well as on the system's evolutionary process. If the user selects an arbitrary composition style at each interaction, the compositional style will most likely diverge. Figure 6(a) shows that all nine compositions share similar tendencies in orientations and in having a good mixture of medium- and small-sized shapes. Figure 6(b) shows a tendency of having many small shapes.

Figure 7 shows the behaviors of two anonymous users who visited the gallery in Ljubljana. There were 901 people interacted with I3 between November 15, 2007 to January 13, 2008. User J.Doe 1 had 68 iterative interactions with I3. Figures 7-(a),(b), and (c) are J.Doe 1's selections of compositions at iterations (i.e., generations) 46, 59, and 68, respectively. This user tends to like a number of smaller shapes enclosed within a larger shape. The user consistently selected such compositions from iteration 59. Note an interesting phenomenon: Figures 7-(b) and (c) look different at a glance. However, both compositions share a quite similar style–the same smaller shapes, triangles for (b) and rectangles for (c), are enclosed in a bigger shape–a big square. More interestingly, in (b), one large circle slightly overlaps with the big square. On the other hand, in (c), two small shapes, one circle and one square, seem to match the one large circle in (b), somehow giving a feeling of a "balanced weight." J.Doe 2 seems to like overlapping triangles and rectangles but no circles. We observed similar behaviors over the most of this user's interactions with I3.

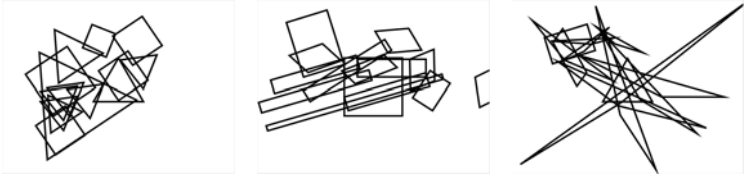
The above observations come from our manual "after-the-interactions" analysis. The analyzer currently is not advanced enough to capture higher-level semantic characteristics such as "balanced weight." We are currently developing a



(a) *User J.Doe 1, at i = 46*

(b) *User J.Doe 1, at i = 59*

(c) *User J.Doe 1, at i = 68*



(d) *User J.Doe 2, at i = 1*

(e) *User J.Doe 2, at i = 29*

(f) *User J.Doe 2, at i = 33*

Fig. 7. User samples from the Ljubljana Exhibit

more sophisticated analyzer that is capable of detecting such higher-level semantics. This new analyzer can give explicit feedbacks to the user, perhaps visually, aiding the process of self-reflection.

Many users among the 901 people have characteristic interactions with I3. However, we have noticed a significant percent of the people prematurely terminated sessions–many of them (about 75%) terminating within 15 iterations. This may be common to many art exhibits in general; however we believe that we can engage more users to I3 by addressing the following limitations of the current system. First, the system doesn't provide an explicit sense of progress to the user during the session. Focused users will not have problems in producing satisfying composition styles. However, casual users may loose interest prematurely due to the lack of goal-oriented guidance from the system. Without an explicit goal, users may terminate sessions too early. We plan to resolve this issue by making I3 interactions similar to a game play with a scoring system.

5 Summary and Future Work

We have presented and discussed an interactive visual composition system that can facilitate visual dialogues with a human user. The system invites the user for a dialogue based on compositions consisting of fundamental shapes and their geometric transformations. Evolutionary algorithms evolve compositional styles through interactions with the user. As the interactions continue, patterns emerge in the dialogue–much the same way as in a dialogue between two people. The result is an *experience* to the user, perhaps self-reflecting realization of the user,

about his or her own proclivities in visual arrangements of various objects. We have also developed an analyzer that can find gestalt tendencies (i.e., styles) in a visual composition and enumerate them in symbolic forms (i.e., a parse tree) that can be easily understood by humans. There are immediately important future works. First, we are working on improving the aesthetic measure by incorporating additional gestalt properties in the analyzer. Second, one of the main concerns that we heard from the users of I3 was a lack of sense of closure and progress over the interactions; often users are not sure about when is the time to finish a session. We are currently working on these issues.

References

1. Birkhoff, G.D.: Aesthetic Measure. Harvard University Press, Cambridge (1933)
2. Boden, M.A., Edmonds, E.A.: What is generative art? Digital Creativity 20(1-2), 21–46 (2009)
3. Collomosse, J.P.: Supervised genetic search for parameter selection in painterly rendering. In: Rothlauf, F., et al. (eds.) EvoWorkshops 2006. LNCS, vol. 3907, pp. 599–610. Springer, Heidelberg (2006)
4. Collomosse, J.P., Hall, P.M.: Genetic paint: A search for salient paintings. In: Rothlauf, F., Branke, J., Cagnoni, S., Corne, D.W., Drechsler, R., Jin, Y., Machado, P., Marchiori, E., Romero, J., Smith, G.D., Squillero, G. (eds.) EvoWorkshops 2005. LNCS, vol. 3449, pp. 437–447. Springer, Heidelberg (2005)
5. Gary, G.: On the origin of the term "computational aesthetics". In: Neumann, L., Sbert, M., Gooch, B., Purgathofer, W. (eds.) Computational Aesthetics 2005, Eurographics Workshop on Computational Aesthetics in Graphics, Visualization, and Imaging, Aire-la-Ville, Switzerland, May 2005, pp. 9–12. Eurographics Association (2005)
6. Hoenig, F.: Defining computational aesthetics. In: Neumann, L., Sbert, M., Gooch, B., Purgathofer, W. (eds.) Computational Aesthetics 2005, Eurographics Workshop on Computational Aesthetics in Graphics, Visualization, and Imaging, Aire-la-Ville, Switzerland, May 2005, pp. 13–18. Eurographics Association (2005)
7. Hofstadter, D.R.: Metamagical themas: Can inspiration be mechanized? Scientific American, 18–34 (September 1985)
8. Holland, J.H.: Adaptation in natural and artificial systems. The University of Michigan Press, Ann Arbor (1975)
9. Machado, P., Cardoso, A.: Computing aesthetics. In: de Oliveira, F.M. (ed.) SBIA 1998. LNCS (LNAI), vol. 1515, pp. 219–228. Springer, Heidelberg (1998)
10. McCormack, J.: Open problems in evolutionary music and art. In: Rothlauf, F., Branke, J., Cagnoni, S., Corne, D.W., Drechsler, R., Jin, Y., Machado, P., Marchiori, E., Romero, J., Smith, G.D., Squillero, G. (eds.) EvoWorkshops 2005. LNCS, vol. 3449, pp. 428–436. Springer, Heidelberg (2005)
11. Sims, K.: Artificial evolution for computer graphics. In: SIGGRAPH 1991: Proceedings of the 18th annual conference on Computer graphics and interactive techniques, pp. 319–328. ACM, New York (1991)
12. Thorisson, K.R.: Simulated perceptual grouping: An application to computer interaction. In: Proceedings of the Sixteenth Annual Conference of the Cognitive Science Society, pp. 876–881 (1994)
13. Todd, S.C., Latham, W.: Evolutionary Art and Computers. Academic Press, London (1992)
14. Jowett, B. (Translator): Plato: Meno. Digireads.com, Stilwell, KS (2006)
15. Wertheimer, M., King, D.: Max Wertheimer and Gestalt Theory. Transaction Publishers, Piscataway (2004)

Graph-Based Evolution of Visual Languages

Penousal Machado[1], Henrique Nunes[1], and Juan Romero[2]

[1] CISUC, Department of Informatics Engineering, University of Coimbra, 3030 Coimbra, Portugal
machado@dei.uc.pt
[2] Faculty of Computer Science, University of Coruña, Coruña, Spain

Abstract. We present a novel evolutionary engine for the evolution of context free grammars. The system relies on specially designed graph-based crossover and mutation operators. While in most evolutionary art systems each individual corresponds to a single artwork, in our approach each individual is a context free grammar that specifies a family of shapes following the same production rules. To assess the adequacy and completeness of the system we perform experiments using automated fitness assignment and user-guided evolution. The experimental results show that the system is able to create diverse and interesting families of shapes even when the initial population is composed of minimal grammars.

1 Introduction

The main inspiration of this research is the seminal work of Stiny and Gips [11] who introduced the concept of Shape Grammars and built, among others, shape grammars that capture the architectural "language" of Frank Lloyd Wright's prairie houses. Although the grammars were hand-built, their results show that: (i) it is possible to capture specific *visual languages* using a set of production rules; (ii) it is then possible to use this set of rules to automatically generate new objects that belong to the same visual language.

With the goal of developing a system that generates novel visual languages, we created an evolutionary engine where each individual is a Context Free Design Grammar (CFDG) [3] and built appropriate genetic operators for their manipulation. The use of CFDGs allows the specification of complex families of shapes through a compact set of rules, and has several potential advantages over several other Evolutionary Art (EA) representations. The development of a graph-based crossover operator was motivated by the need to take into account the underlying structure of the individuals while exchanging genetic code.

A thorough survey of EA systems is beyond the scope of this paper and can be found in [5]. To the best of our knowledge, there are two examples of the use of CFDG for EA. Unfortunately, neither of them allows the evolution of visual languages. *CFDG Mutate* [1] only allows the application of mutation operators and does not handle non-deterministic grammars, which means each individual represents a single shape (see Section 2). Saunders and Grace [10] present a parametric system that evolves parameters of specific CFDG hand-built grammars.

C. Di Chio et al. (Eds.): EvoApplications 2010, Part II, LNCS 6025, pp. 271–280, 2010.

Although it allows some degree of exploration, it has the same shortcomings as other parametric evolution approaches: there are strong constraints that limit the search space and define the type of imagery produced by the system.

A previous work [8] showed that our engine allows the evolution of interesting, complex and diverse visual languages when the initial population is composed of hand-built CFDGs. In this paper we focus on the description of the evolutionary engine, namely crossover and mutation operators, and on testing its generation abilities in the absence of hand-built grammars.

2 Context Free

Context Free [4] is a popular open-source application that renders images specified using a simple language entitled CFDG (for a full description of CFDG see [3]). In essence, and although the notation is different from the one used in formal language theory, a CFDG program is an augmented context free grammar, i.e., a 4-tuple: (V, Σ, R, S) where:

1. V is a set of nonterminal symbols
2. Σ is a set of terminal symbols
3. R is a set of production rules that map from V to $(V \cup \Sigma)^*$
4. S is the initial symbol

Fig. 1 presents a grammar and the image it generates. Programs are interpreted by starting with S (in this case $S = TREE$) and proceeding by the expansion of the production rules in breath-first fashion. Predefined Σ symbols call drawing primitives (e.g., CIRCLE). CFDG is an *augmented* context free grammar: it takes parameters that produce semantic operations (e.g., *size* produces a scale change). Program interpretation is terminated when there are no V symbols left to expand or the further expansion does not change the image [8].

The grammar of Fig. 1 is deterministic, there is exactly one rule for each V symbol, therefore its interpretation will always result in the same image. To specify languages of shapes we have to resort to non-determinism. In Fig. 2 we present a non-deterministic version of this grammar with two different production rules for the 'TREE' symbol. When several production rules are applicable one of them is selected randomly and the expansion proceeds. One can control the probability of selection by specifying a weight after the V symbol (0.8 and 0.2 in Fig. 2).

3 Evolutionary Context Free Art

In this section we describe our evolutionary engine. For the sake of parsimony we focus on the key components of the system.

Each genotype is a well-constructed CFDG grammar. Phenotypes are rendered using Context Free. To deal with nonterminating programs a maximum rendering time is set. The genotype is represented by a directed graph created as follows:

```
startshape TREE
rule TREE  { CIRCLE {}
             TREEA {size 0.95 y 1.6}}
rule TREEA { CIRCLE {}
             TREEB {size 0.95 y 1.6}}
rule TREEB { CIRCLE {}
             TREEC {size 0.95 y 1.6}}
rule TREEC { CIRCLE {}
             TREED {size 0.95 y 1.6}}
rule TREED { CIRCLE {}
             TREE {size 0.95 y 1.6 rotate 45}
             TREE {size 0.95 y 1.6 rotate -45}}
```

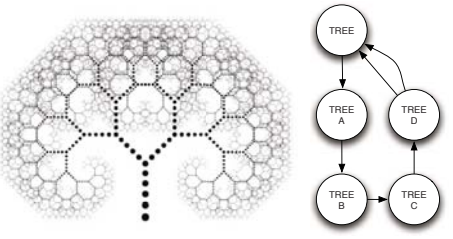


Fig. 1. A deterministic grammar, the tree-like shape it generates, and its graph representation. The labels of the edges have been omitted.

```
startshape TREE
rule TREE 0.80 {
     CIRCLE {}
     TREE {size 0.95 y 1.6}
}
rule TREE 0.20 {
     CIRCLE {}
     TREE {size 0.95 y 1.6
           rotate 45}
     TREE {size 0.95 y 1.6
           rotate -45}}
```

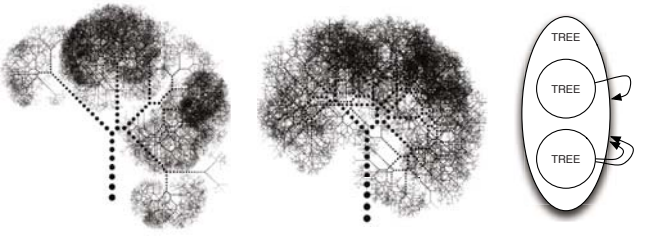


Fig. 2. A non-deterministic version of the grammar presented in Fig. 1, two instances of the family of tree-like shapes it defines and its graph representation. The labels of the edges have been omitted.

1. Create a node for each nonterminal symbol. In deterministic grammars each node represents a single production rule (see Fig. 1). In non-deterministic grammars each node encapsulates the set of all production rules associated with the nonterminal symbol, e.g., the grammar presented on Fig. 2 results in a directed graph composed of a single node that encapsulates the two rules associated with the nonterminal 'TREE'.
2. Create edges between each node and the nodes corresponding to the nonterminals appearing in its production rules (see Figs. 1 and 2).
3. Annotate each edge with the corresponding parameters (e.g. In Fig. 1 the edge connecting TREE with TREEA possesses the label 'size 0.95 y 1.6')

3.1 Crossover Operator

The design of genetic operators that are well-suited to the representation is vital for the success of an evolutionary algorithm. In our case the biggest challenge was to design a crossover operator that allows the meaningful exchange of genetic material between individuals. Given the nature of the representation, we developed a graph-based crossover operator based on the one presented by Pereira et al. [9]. In simple terms, this operator allows the exchange of subgraphs between individuals.

The crossover of the genetic code of two individuals, a and b, implies: (i) Selecting one subgraph from each parent; (ii) Swapping the nodes and *internal* edges of the subgraphs, i.e., edges that connect two subgraph nodes; (iii) Establishing a correspondence between nodes; (iv) Restoring the *outgoing* and *incoming* edges, i.e., respectively, edges from nodes of the subgraph to non-subgraph nodes and edges from non-subgraph nodes to nodes of the subgraph.

Subgraph selection. Randomly selects for each parent, a and b, one crossover node, v_a and v_b, and a subgraph radius, ra and rb. Subgraph s_{ra} is composed of all the nodes, and edges among them, that can be reached in a maximum of ra steps starting from node v_a. Subgraph s_{rb} is defined analogously.

Swapping the subgraphs. Swapping s_{ra} and s_{rb} consists in replacing s_{ra} by s_{rb} (and vice-versa). After this operation the outgoing and the incoming edges are destroyed. Establishing a correspondence between nodes repairs these connections.

Correspondence of Nodes. Let s_{ra+1} and s_{rb+1} be the subgraphs that would be obtained by considering a subgraph radius of $ra + 1$ and $rb + 1$ while performing the subgraph selection. Let mst_a and mst_b be the minimum spanning trees (MSTs) with root nodes v_a and v_b connecting all s_{ra+1} and s_{rb+1} nodes, respectively. For determining the MSTs all edges are considered to have unitary cost. When several MSTs exist, the first one found is the one considered. The correspondence between the nodes of s_{ra+1} and s_{ra+1} is established by transversing mst_a and mst_b, starting from their roots, as described in Algorithm 1.

Restoring outgoing and incoming edges. The edges from $a \notin s_{ra}$ to s_{ra} are replaced by edges from $a \notin s_{ra}$ to s_{rb} using the correspondence between the nodes established in the previous step (e.g. the incoming edges to v_a are redirected to v_b, and so on). Considering a radius of $ra + 1$ and $rb + 1$ instead of ra and rb in the previous step allows the restoration of the outgoing edges. By definition, all outgoing edges from s_a and s_b link to nodes that are at a minimum distance of $ra + 1$ and $rb + 1$, respectively. This allows us to redirect the edges from s_b to $b \notin s_b$ to $a \notin s_a$ using the correspondence list.

Figs. 3 and 4 present examples of the crossover operator at the genotype and phenotype level.

3.2 Mutation Operators

The development of the mutation operators was guided by the need to introduce new genetic code and to ensure that the search space is fully connected, i.e., that all of its points are reachable from any starting point by the successive application of genetic operators. This resulted in the use of ten operators, for which, due to space limitations, we only present a cursory description:

Startshape mutate – randomly selects a nonterminal as starting symbol;
Replace, Remove or Add symbol – when applied to a given production rule these operators: *replace* one of the present symbols by a randomly selected

Algorithm 1. transverse(a, b)

```
set_correspondence(a, b)
mark(a)
mark(b)
repeat
  if unmarked(a.descendants) ≠ NULL then
    next_a ← RandomlySelect(unmarked(a.descendants))
  else if a.descendants ≠ NULL then
    next_a ← RandomlySelect((a.descendants))
  else
    next_a ← a
  end if
  {**** do the same for next_b ****}
  transverse(next_a, next_b)
until unmarked(a.descendants) = unmarked(b.descendants) = NULL
```

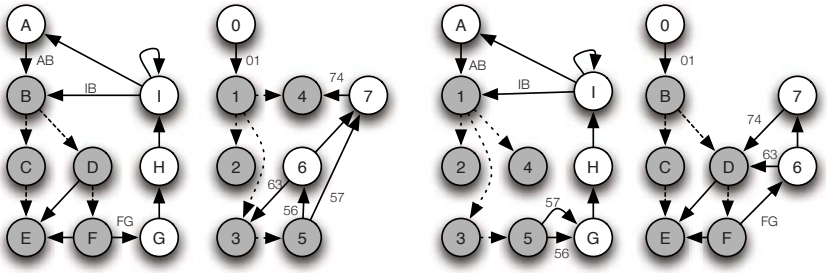


Fig. 3. On the left, the graphs of two parents. Considering $v_a = B$, $v_b = 1$ and $ra = rb = 2$, yields the subgraphs s_{ra} and s_{rb} whose nodes are depicted in grey. To establish the correspondence between nodes, the MSTs (here represented by dotted edges) are determined and Algorithm 1 applied. Considering that the algorithm returns the correspondence list {B-1, C-2, E-2, D-3, F-5, G-6, G-7, D-4}, the crossover operation results in the two descendants presented on the right. To help the reader, we added labels to the outgoing and incoming edges.

one; *remove* a symbol and associated parameters from the production rule; *add* a randomly selected symbol in a valid random position. Notice that these operators are applied to terminal and nonterminal symbols.

Duplicate, Remove or Copy&Rename – these operators: *duplicate* a production rule; *remove* a production rule, updating the remaining rules when necessary; *copy* a production rule, assigning a new randomly created name to the rule and thus introducing a new nonterminal; the copy&rename mutation will only affect the phenotype once the `Add symbol` mutation introduces a call to the new nonterminal in a production rule.

Change, Remove or Add parameter – as the name indicates these operators *add*, *remove* or *change* parameters and parameter values;

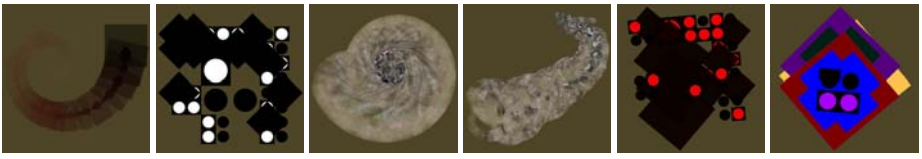


Fig. 4. The two leftmost images are the parents, the others are results of their crossover

4 Experimentation

In a previous study [8] we showed the adequacy of the engine to evolve families of shapes when the initial population included hand-built grammars. In the current work we are interested in determining if the system is self-sufficient and if it can generate interesting and novel images and shape families without resorting to hand-built grammars. For this purpose we conducted experiments using automated-fitness assignment and user-guided evolution. Using an initial population of randomly created grammars could hide possible shortcoming of the genetic operators. As such, we chose to use as starting population a single individual whose genotype consists of a minimal grammar: a startshape directive and a single production rule composed of a call to the SQUARE terminal (the list of CFDG predefined terminals can be found in [3]).

We use a generational non-elitist approach and roulette wheel selection. In all experiments presented here: population size = 50, crossover probability = 0.7, maximum crossover radius = 3, mutation probability = 0.01 per individual (for each of the ten mutation operators). In the automated fitness runs the max number of generations = 100, while in the user-guided ones the stopping criteria were determined by the users.

4.1 Fitness Functions

In user-guided runs the fitness is supplied by the user who may assign to each individual a value in the $[0, 9]$ interval. For automated fitness runs we experimented three different fitness formulas:

RMSE – The fitness of an individual, a, is calculated by determining the root mean square error between its phenotype, $i(a)$, and a target image, t, as follows: $fitness(a) = 1/(1 + rmse(i(a), t))$

Fractal Dimension – Following Taylor et al. [12], among others, fitness is assigned by estimating the fractal dimension (FD) of $i(a)$ and comparing it with a target value, v, as follows: $fitness(a) = 1/(1 + (FD(i(a)) - v)^2)$. FD is estimated by the box counting method using the *chaos* library (`http://www.math.uic.edu/~culler/chaos/`). In the experiments presented here two arbitrarily chosen target v values, 1.3 and 0.8, are considered.

JPEG – We use JPEG compression to estimate image complexity [7]. Fitness is proportional to the file size of the JPEG encoding of $i(a)$: $fitness(a) = max(0, (size(jpeg(i(a)) - const)/1500)$. To increase the evolutionary pressure, a subtraction by a constant is performed ($const = 1000$ in all runs, a

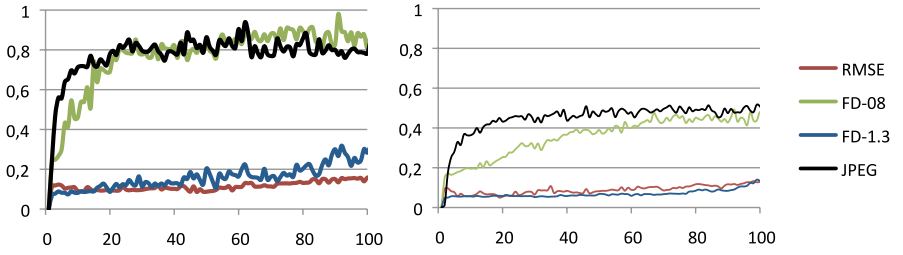


Fig. 5. Evolution of the best (left) and average (right) fitness throughout the automated fitness assignment runs. Results are averages of 10 runs.

value that was set taking into account the file size, 1100, of the image of the first population).

When the genotype encodes a non-deterministic grammar the phenotype may, and does, change from one interpretation to the other. As such, the fitness of an individual may vary from generation to generation. In other words, although an individual is a visual grammar, we assign fitness based on a single sample of this grammar. This design option poses an additional difficulty for the EC algorithm. Nevertheless, in theory, it should eventually converge to grammars predominantly composed of highly fit images.

4.2 Experimental Results

The analysis of the experimental results attained by EA systems typically implies a high degree of subjectivity. The main goals of the automated fitness runs were: (i) determine if the evolution of fitness values corresponds to the expectable behaviour of an EC engine; (ii) determine if the system is able to evolve complex grammars starting from a minimal one. In other words, the goals are validating the engine, the adequacy of the genetic operators, and test their ability to introduce novelty and promote complexification. The possible discovery of interesting shapes is not a goal.

The charts of Fig. 5 show the evolution of fitness throughout the automated runs. As can be observed, these charts display the typical behaviour of EC approaches. Fig. 6 presents some of the individuals evolved in these runs. Generally speaking, and although it is subjective to say it, the runs using RMSE fitness produced the least interesting results. This was expected and consistent with the results attained by researchers using RMSE fitness in expression-based EA (see, e.g., [2]). The individuals evolved using JPEG fitness created the most complex shapes, a result that was also expected. Interestingly, in several of these runs the system tended to evolve star-like shapes. This can be explained by two factors: (i) it is relatively easy to find a compact grammar that generates star-like shapes; (ii) the resulting images often result in a relatively large JPEG files.

The goal of the user-guided runs was to show that it was possible to generate diverse, interesting and novel shapes starting from "scratch". Typically the user guided runs had 20 to 40 generations, and the first half of the run was spent on

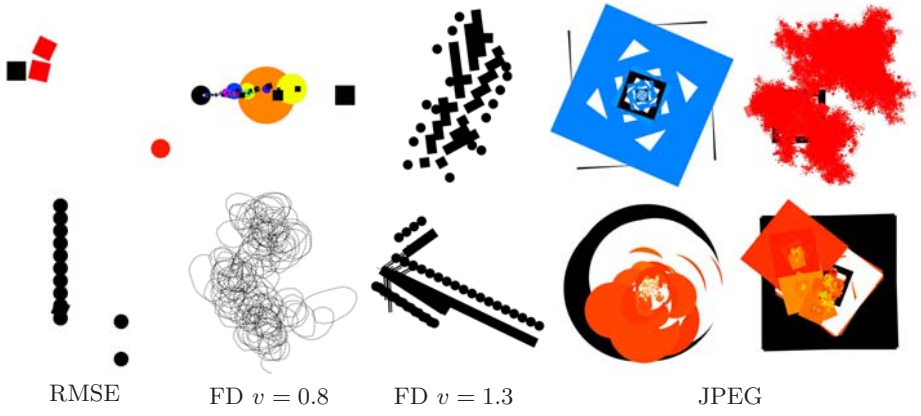


Fig. 6. Examples of images created in automated fitness runs

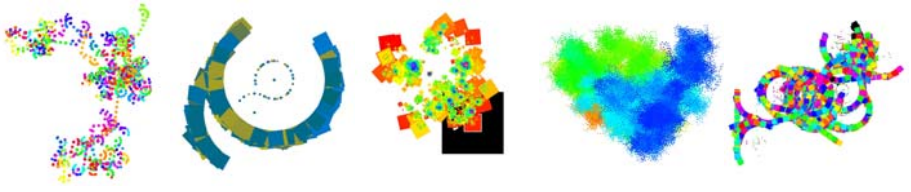


Fig. 7. Examples of images created in user-guided evolution runs

finding a grammar that draws more than a single shape. Thus, the use of a single and minimal grammar as first population makes the first generations of these runs quite boring for the user and, in normal circumstances, we would not recommend this initialization approach. Nevertheless, for the purposes of testing the system, we found it adequate. Fig. 7 presents examples of the images evolved in different user guided runs, showing what can typically be expected in these circumstances.

Although it would probably be advisable to increase the mutation probability during the first generations of these runs, the mutation operators proved valid, producing outcomes that are conceptually similar to the effects of mutation in expression-based EA. That is, the effects of mutation range from minor visual alterations to dramatic changes in appearance induced by small changes of the genetic code, with the later occurring less often [6].

In subsequent experiments, we used some of the individuals evolved in these runs as initial populations of other user-guided runs. Fig. 8 presents examples of images evolved in this fashion. The higher population diversity allowed us to get a better grasp of the behaviour of the crossover operator. In general terms, the outcome of the crossover operator appears to depend on the structural similarity of the genotypes and on their size. Additionally, when the parents are unrelated the visual appearance of each descendent tends to be mostly determined by one of the parents (see Fig. 4). An effect that is more visible with small genotypes.

As stated previously, non-deterministic grammars allow the definition of a family of shapes. The potential for evolving families of shapes instead of single

Fig. 8. Examples of images created in user-guided runs initiated with images from previous runs

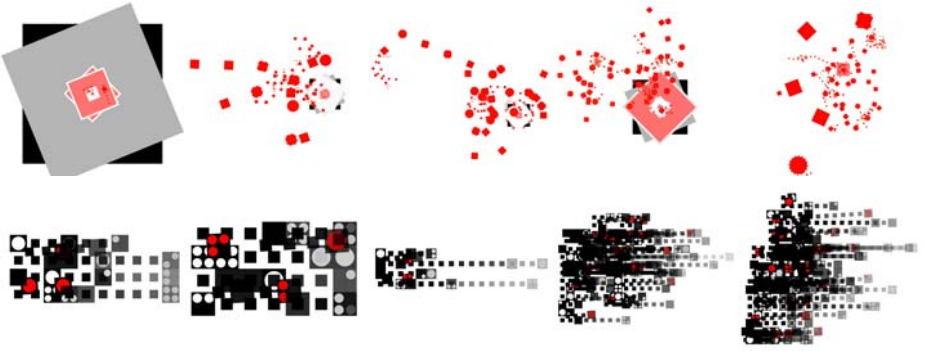


Fig. 9. Instances of the language of shapes defined by two individuals

images is one of the main motivations for the use of CFDGs. However, in the runs presented here, while assessing the individuals the user only has access to one instance of the images an individual may generate. This means that the quality, diversity and consistency of the language of shapes encoded by the individual is not directly assessed. Nevertheless, and somewhat surprisingly, non-trivial and interesting families of shapes were still evolved. This is arguably explained by the following factors: (i) The experimental settings, namely starting conditions and genetic operators, naturally lead to non-deterministic grammars, which is confirmed by the analysis of the individuals generated in automated fitness runs; (ii) User Fatigue – the user eventually grows tired of individuals that systematically generate the same image, therefore the evolutionary process indirectly favors non-deterministic grammars; (iii) individuals that fail to reliably generate images valued by the user will eventually be discarded by evolution, in other words consistency is also favored. Fig. 9 presents instances of the visual languages defined by two of the evolved individuals.

5 Conclusions and Future Work

We presented a novel evolutionary engine that allows the evolution of CFDGs, is able to cope with non-deterministic grammars, and allows their recombination through a graph-based crossover operator. Due to these abilities, it successfully overcomes the limitations of previous EC approaches where CFDGs are used. When compared with typical expression-based and parametric evolution models, our approach presents several advantages, including the abilities: to evolve visual

languages instead of individual images; to use hand-coded grammars; and to allow the direct editing of the genotypes by the user.

Although the interpretation of the results is subjective, they provide evidence of the adequacy of the developed crossover and mutation operators. They also indicate that further experimentation is required to fully explore the potential of the approach for the creation of visual languages. Nevertheless, we consider this to be an important step in that direction.

In terms of future work, redesigning the user interface, exploring automatic image fitness assignment schemes, and developing approaches to automatically assess a language of shapes in terms of consistency, diversity and aesthetic qualities of the generated images are our top priorities.

Acknowledgments. We would like to thank the anonymous reviewers for their thorough and insightful comments. This research is partially funded by the Spanish Ministry for Science and Technology, research project $TIN2008-06562/TIN$.

References

1. Borrell, A.: CFDG Mutate, http://www.wickedbean.co.uk/cfdg/index.html (last accessed in September 2009)
2. Colton, S., Torres, P.: Evolving approximate image filters. In: Giacobini, M., et al. (eds.) EvoWorkshops 2009. LNCS, vol. 5484, pp. 467–477. Springer, Heidelberg (2009)
3. Coyne, C.: Context Free Design Grammar, http://www.chriscoyne.com/cfdg/ (last accessed in September 2009)
4. Horigan, J., Lentczner, M.: Context Free, http://www.contextfreeart.org/ (last accessed in September 2009)
5. Lewis, M.: Evolutionary visual art and design. In: Romero, J., Machado, P. (eds.) The Art of Artificial Evolution: A Handbook on Evolutionary Art and Music, pp. 3–37. Springer, Heidelberg (2007)
6. Machado, P., Cardoso, A.: All the truth about NEvAr. Applied Intelligence. Special Issue on Creative Systems 16(2), 101–119 (2002)
7. Machado, P., Cardoso, A.: Computing aesthetics. In: de Oliveira, F.M. (ed.) SBIA 1998. LNCS (LNAI), vol. 1515, pp. 219–228. Springer, Heidelberg (1998)
8. Machado, P., Nunes, H.: A step towards the evolution of visual languages. In: First International Conference on Computational Creativity, Lisbon, Portugal (2010)
9. Pereira, F.B., Machado, P., Costa, E., Cardoso, A.: Graph based crossover — A case study with the busy beaver problem. In: Proceedings of the Genetic and Evolutionary Computation Conference, Orlando, Florida, USA, vol. 2, pp. 1149–1155. Morgan Kaufmann, San Francisco (1999)
10. Saunders, R., Grace, K.: Teaching evolutionary design systems by extending "Context Free". In: Giacobini, M., et al. (eds.) EvoWorkshops 2009. LNCS, vol. 5484, pp. 591–596. Springer, Heidelberg (2009)
11. Stiny, G., Gips, J.: Shape grammars and the generative specification of paintings and sculpture. In: Freiman, C.V. (ed.) Information Processing, vol. 71, pp. 1460–1465. North Holland Publishing Co., Amsterdam (1971)
12. Taylor, R.P., Micolich, A.P., Jonas, D.: Fractal analysis of Pollock's drip paintings. Nature 399, 422 (1999)

Refinement Techniques for Animated Evolutionary Photomosaics Using Limited Tile Collections

Shahrul Badariah Mat Sah, Vic Ciesielski, and Daryl D'Souza

School of Computer Science and Information Technology,
RMIT University, GPO Box 2476V Melbourne Victoria 3001, Australia
smatsah@cs.rmit.edu.au,
{vic.ciesielski,daryl.dsouza}@rmit.edu.au

Abstract. An *animated evolutionary photomosaic* is produced from a sequence of still or static photomosaics to evolve a near match to a given target image. A static photomosaic is composed of small digital images or tiles, each having their own aesthetic value. If the tiles are prepared manually, the tile collections are typically small. This potentially limits the visual quality of a photomosaic as there may not be sufficient options for matching tiles. We investigate the use of colour adjustment and tile size variation techniques via genetic programming to improve the animated photomosaics. The results show that colour adjustment improved both visual quality and fitness. However, it can produce strange looking tiles. Tile size variation was able to focus on details in the target image but produced slightly worse fitness values than an equal-sized tiles approach. Combining these techniques revealed that, regardless of the size of tiles, colour adjustment was the dominant refinement. In conclusion, each of these techniques is able to produce an aesthetically different animation effect and presents a better mechanism for generating photomosaics when only a limited number of tiles is available.

Keywords: Non-photorealistic rendering, animated evolved photomosaic, evolved art, genetic art, genetic programming, digital art.

1 Introduction

An animated evolutionary photomosaic is a movie comprising a sequence of photomosaics, each generated by an evolutionary algorithm, which evolves into a close match to a selected target image. A photomosaic is a composite digital image made of smaller images or digital *tiles*. When viewed close up the content of each tile is visible and is potentially a significant digital image. When viewed from a distance the subject of the entire photomosaic becomes the central feature.

Due to the differences in resolution between paper rendering and screen display, and tiling strategy considerations, aesthetic criteria for the two photomosaic output options need to be different. A tile needs to be large enough to be able to show its content on the screen and yet small enough to closely match the

C. Di Chio et al. (Eds.): EvoApplications 2010, Part II, LNCS 6025, pp. 281–290, 2010.

target image. A common way to generate a photomosaic is through placement of the tiles onto a two-dimensional grid in an arrangement that best matches the target image. The grid consists of N cells, where a cell is a space on the grid. Any selected tile must fit into a cell on the grid. Consequently, the size of the cells will also determine the size of tiles. A large number of tiles, in the order of hundreds of tiles or more, provides a high probability of closely matching tiles, and hence, potentially generating an accurate photomosaic. However, when tile collections are limited by size (in our experiments collection size is no more than 50 tiles) or do not serve desired thematic requirements (tiles are thematically incompatible with the target image) visual quality may be compromised. For photomosaics involving limited collections of tiles the best static photomosaic may be effectively constructed through exhaustive enumeration. However, our interest is in animations in which the target image emerges gradually from an initial random allocation of tiles on a canvas. Such animations cannot be generated via exhaustive search.

This paper reports our efforts to generate better animated photomosaics in relation to these limitations. We propose two techniques to improve animated colour photomosaics: colour adjustment and tile size variation. These, respectively, refer to amendments to the tile colour and tile size during photomosaic generation. Specifically, the following research question is addressed:

Do (a)colour adjustment, (b)tile size variation, separately, and in combination improve the animated photomosaics when only a limited number of tiles is available?

Our criteria for *improved* animated photomosaics involve two metrics. One is visual quality, is a combination of target image accuracy in the form of subject resemblances, rate of target visibility and tile clarity. The other is fitness based on pixel difference measurements.

The remainder of this paper is structured as follows. Section 2 presents work by others in improving photomosaics. Section 3 describes our colour adjustment and tile size variation refinements. Section 4 presents our experiments and reports results thereof, and finally, Section 5 reports on findings and discusses future work.

2 Related Work

Computer generated photomosaics were introduced by Robert Silvers in 1997 [2]. Silvers adopted what we denote the straightforward approach, which uses equal-sized tiles and exhaustive search, with large tile collections in the generation process and without modification to the tiles. Since then, much research has been carried out into photomosaics, ranging from improvements to the generation process itself [1,8,10] to producing refined or new aesthetic styles of photomosaic [5,6,7,9,11]. This work is primarily concerned with generation of static photomosaics, potentially for printing.

We review related work involving refinements to photomosaics with limited tile collections. One such technique is to use arbitrary shaped tiles instead of rectangular tiles, and also allow some degree of deformation. Jigsaw Image Mosaic

(JIM) [5] allows for tile deformation to reduce the gaps between tiles. Another system, Cut-out Mosaic [6], uses portions from images as tiles to generate a photomosaic. Although these approaches are able to more closely match the target image, source tile originality is compromised. Without seeing the source image of the cutout tile, it is no longer easy to identify the actual subject of the tile. Di Blasi et al. [7] explore multi-resolution tiles by using a quadtree image representation. In this work small-sized tiles are used to represent areas that consist of many colour variations, while areas of uniform colour are represented by single, large image tiles. This approach preserves the tile content and also potentially produces a close match to the target image.

Another approach for improving the visual quality of a mosaic is through colour adjustment. In the generation of static photomosaics, Finkelstein and Range [8] apply colour adjustment as an optional operation, *after* a photomosaic has been generated. Orchard and Kaplan [6] allow colour or luminance adjustment to the cut-out mosaic. Although these techniques work well with static photomosaics generation, they have yet to be tested in the generation of animated photomosaics, as the generation involves many intermediate frames of static photomosaics.

Prompted by these contributions, we incorporate colour adjustment and tile size variation into our animated evolutionary photomosaic generation system. Previous research in animated photomosaic generation has focused on equal-sized digital tiles, both for gray level [3] and in colour [15] animations. Our work continues the research from the perspective of animated colour photomosaics generation. The motivation for our work is to generate evolved animated photomosaics which have close resemblance to the target, through colour adjustment and tile size variation.

3 Refinement Strategies in Photomosaic Generation

The straightforward approach to photomosaic generation uses equal-sized tiles and large tile collections. When generating photomosaics with limited tile collections, such a tiling strategy is limited by the choice and number of matching tiles. As a result, some photomosaics retain abstract forms derived earlier in the evolution, and do not resemble the target image. This calls for refinements and we propose two possible approaches to address this problem: colour adjustment and tile size variation. We introduce both refinements in the context of Genetic Programming (GP) evolution, and describe the refinements in the next two sub-sections.

3.1 Colour Adjustment

We define colour adjustment as changing the average colour of a tile to the average colour of a region in the target image, which changes the overall colour of the tile while retaining its geometrical information. For colour adjustment, we construct the photomosaic using equal-sized tiles. As photomosaic generation

Table 1. GP configuration for Colour Adjustment and tile size variation Techniques

Parameter	Colour Adjustment	tile size variation
Population Size	10	10
Max Generations	50,000	40,000
Crossover Rate	0.70	0.70
Mutation Rate	0.20	0.20
Elitism Rate	0.10	0.10
Max depth	10	10
Min depth	2	2
Terminal	TileIndex	TileIndex
Function	DrawTile, TileJoint	DrawTile,TileJoint
Target Size	800 x 600 pixels	600 x 600 pixels
Tile Size	20 x 20 pixels	100 x 60 pixels
Selection	Proportional to fitness	Proportional to fitness
Replacement	Generational replacement	Generational replacement
Termination	Number of generations	Number of generations

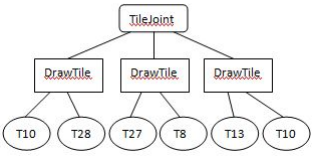


Fig. 1. Example of an individual in a population

divides a target image into a finite number of regions and uses a set of tiles to choose the best possible matches, the problem may be viewed as an arrangement problem, having some of the characteristic of the traveling salesperson problem [12] and the stock cutting problem [13]. In our GP implementation, an individual (photomosaic) in a population represents a non-overlap arrangement of equal-sized, juxtaposed tiles which completely cover the grid. The process of generating a photomosaic begins by dividing the target image into a grid of N cells for tile placement. Tiles to be placed on the grid are initially selected randomly by the evolutionary program.

The evolution parameters, terminal and functions sets are presented in Table 1. Figure 1 shows an example of an evolved individual in a population. The i^{th} terminal corresponds to the i^{th} cell in the grid. Thus tile number 28 (Figure 1) will occupy cell number 2. As individuals in GP programs are represented by parse trees of varying height, only the first N terminals traversed from a parse tree will be used to generate a photomosaic. Parse trees with fewer than N terminals are penalised and discarded from the evolutionary process. The GP representation was selected based on the results of previous work that compared GP and GA representations for photomosaic generation [14]. Our colour adjustment occurs during the process of arranging tiles onto the canvas. We explain the colour adjustment approach with reference to the placement of a single tile

in a grid cell, using Equation 1. The calculation is done separately on each R, G and B channel.

$$C(x) = x + w(ar - at) \tag{1}$$

$$w = CurrentGeneration/MaximumGenerations \tag{2}$$

Parameter x is a value of a colour channel of a pixel (e.g. Blue channel value), ar is the average colour of a target image region, at is the average colour of a selected tile and $(ar - at) \in \mathbb{R}$. The calculated amount is then thresholded as follows:

$$C(x) = \begin{cases} 255 & \text{if } C(x) > 255 \\ 0 & \text{if } C(x) < 0 \\ x + w(ar - at) & \text{otherwise} \end{cases}$$

The thresholding process is to ensure that the calculated value is within the valid range of 0 to 255, since each channel is represented using eight bits. The weight (w) is applied in order to create a gradual effect of colour adjustment for the evolved photomosaic, as w will be 1 at the final generation. This gradual change in colour, rather than an abrupt colour change, produces a more aesthetically pleasing animation effect.

3.2 Tile Size Variation

In the straightforward approach the size of the tiles influences the fidelity of the generated photomosaic to the target image. It is difficult to arrive at a balance between target image accuracy and tile clarity. In our second refinement, we vary tile sizes via quadtree area division to overcome this problem. Quadtree area division is carried out by partitioning the target image into *regions* of varying size based on colour [16].

The process of division begins with dividing the target image into four regions. For each quadrant, the largest pixel difference between each pixel and the average colour of the region is calculated. The value is than compared to a threshold value to determine further division. The threshold value is important as it allows the quadtree division to capture details of the image. Large threshold values will create uniform distribution of regions while small threshold values will detect the occurrences of edges. A threshold value of 0.2, based on pilot experiments, was used in the implementation. The division process is stopped when it reaches either of two conditions: the calculated pixel difference value is smaller than the threshold value or the size of a region reaches a minimum area size, specified as 10×10 pixels.

To retain the relationship between the entire image and the divided regions, the results of the quadtree division are organised in a tree-based structure with the entire image as the root. The regions with small pixel value differences or with minimum size are consider as the leaf nodes, which is the number of *cells*, N, for tile placement. Each cell contains information about its size, the coordinates of the top-left corner in the actual target image, and the average colour value. The GP implementation uses this cell information to evolve the photomosaics. The same parameter settings were employed as for colour adjustment (Table 1).

3.3 Fitness Evaluation

Our aim is to minimise the *sum-of-pixel-differences* between the candidate photomosaic and the target image, as the fitness calculation, and is formally expressed by the Equation 3.

$$\frac{\sum_{i=1}^{K}\sum_{j=1}^{L}|target(i,j) - individual(i,j)|}{K \times L} \tag{3}$$

Here, i and j correspond to a pixel's position, so $|target(i,j) - individual(i,j)|$ is the actual difference between corresponding pixel positions in the target and individual (generated photomosaic). The sum computes the total value of differences from R, G and B channel for each pixel pair. The sum of these differences is normalised to a value in the range of [0,1].

4 Results and Discussion

For colour adjustment, a target image of 800x600 pixels (Figure 2) was selected, with a tile collection of 50 images of 20x20 pixels. For tile size variation, we used a target image with a resolution of 600x600 pixels (Figure 3) with 29 tiles of 100 × 60 pixels. The images were chosen based on colour complexity and level of detail in the images. The tile collection was curated on the basis of the desired thematic scheme of the target images[1]. We present the results for colour adjustment and tile size variation in the following subsections.

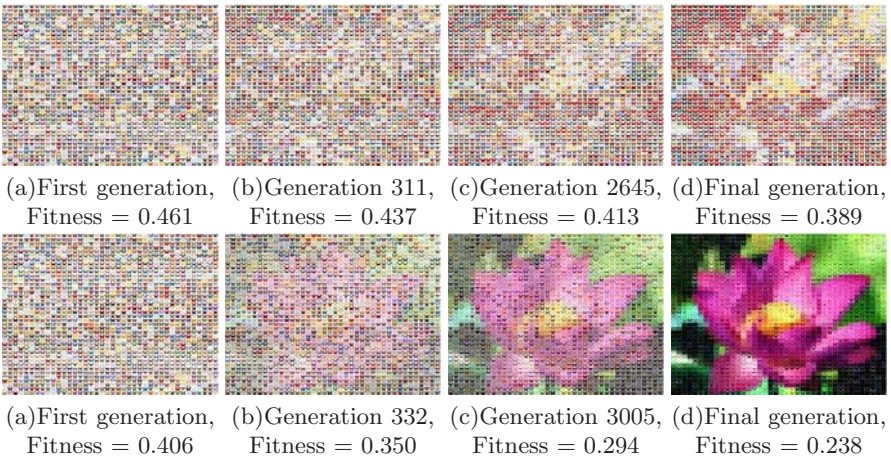
4.1 Colour Adjustment

Quantitative and qualitative assessments were used to measure the performance of this technique. Pixel differences between the final photomosaic and the target were used as the quantitative measures. Qualitative assessments were carried out with the help of two artists, via focus group interviews. Figure 2 shows the generated photomosaic of a lotus flower image with and without colour adjustments. A close-up of the butterfly tiles is shown in Figure 4.

Figure 2 shows that fitness values for colour adjustment are smaller than those for the equal-sized tiles approach. These values indicate that at the same number of generations (in our experiment it was 50,000 generations), colour adjustment was able to generate more acurate photomosaics.

This finding is also supported by the visual qualities of the generated photomosaics, which can be clearly seen in Figure 2. Without colour adjustment, the tile collection was unable to produce a likeness to the target image. This was due to the limited colour variation in the tile collection. For the qualitative assessment, the colour adjustment technique allows the subject of the target image to be visible at the early stage of the animation, when compared to the animation

[1] The evolved photomosaic and the animations can be found at http://evolart.cs.rmit.edu.au/research/evomusart10/



(a)First generation, Fitness = 0.461 (b)Generation 311, Fitness = 0.437 (c)Generation 2645, Fitness = 0.413 (d)Final generation, Fitness = 0.389

(a)First generation, Fitness = 0.406 (b)Generation 332, Fitness = 0.350 (c)Generation 3005, Fitness = 0.294 (d)Final generation, Fitness = 0.238

Fig. 2. This figure presents the photomosaics for Lotus target image. The first row shows selected photomosaics without colour adjustment while the second row displays selected outputs of colour corrected photomosaic.

produced by the straightforward photomosaic. This creates the illusion of the subject *emerging* from the tile arrangement, which is different from the *card shuffling* animation effect produced by the non-colour corrected photomosaic. The subject also remains for a longer period in the animation. As the colour adjustments were applied gradually for this tile collection of butterflies, the earlier part of the animation showed a *butterflies fluttering* effect which then revealed the flower, as it was fully played.

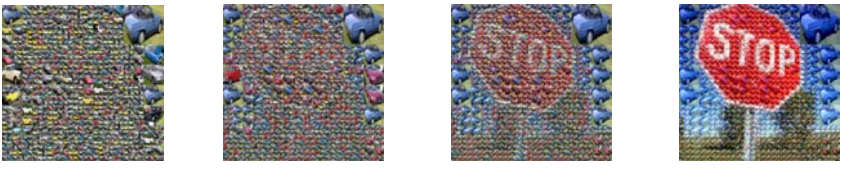
We also observed that our colour adjustment technique can produce negative outcomes, where it might produces strange coloured tiles. However, for this set of tiles and target image, this has not been a problem. The artists agreed that each animation effect has different aesthetic qualities, and is suitable for different sets of tiles and target images.

4.2 Tile Size Variation

For tile size variation we used a target image of a stop sign (Figure 3) and a tile collection of car images (Figure 4). The generated photomosaics were able to capture the details of the target image, which are shown in row 2 of Figure 3. The shape of the road sign and the word STOP are more visible in the final static photomosaic as compared to the final static photomosaic of the straightforward approach (row 1 of Figure 3). This is because the quadtree division was able to represent the edges in the images using smaller sized tiles (about 10x10 pixels), as compared to 20x20 pixels for the straightforward approach. Not surprisingly the technique produced slightly worse fitness values than the straightforward approach due to the simplicity of the fitness measurement.

(a)First generation, Fitness = 0.303 (b)Generation 110, Fitness = 0.281 (c)Generation 312, Fitness = 0.258 (d)Final generation, Fitness = 0.236

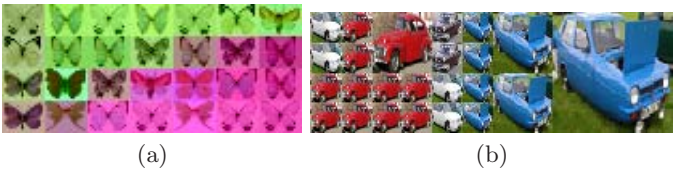
(a)First generation, Fitness = 0.310 (b)Generattion 640, Fitness = 0.288 (c)Generation 3765, Fitness = 0.265 (d)Final generation, Fitness = 0.243



(a)First generation, Fitness = 0.212 (b)Generation 5028, Fitness = 0.201 (c)Generation 11220, Fitness = 0.190 (d)Final generation, Fitness = 0.180

Fig. 3. This figure presents the photomosaics for the stop sign target image. The first row shows selected individuals from equal-sized tiles, the second row shows individuals from the tile size variation technique and the third row shows individuals from a combination of the techniques.

Further investigation of both the colour adjustment and the tile size variation techniques were carried out by combining the techniques in the photomosaic generation. Visually, the generated photomosaics showed a further improvement, over each of the single technique approaches. This is evident in row 3 of Figure 3. However, the best fitness values of the combined technique did not differ from the fitness of the colour adjustment technique. Figure 5 presents the best fitness graphs for each technique. The tile size variation technique has the worst performance in comparison to other techniques. The colour adjustment and combination techniques showed the most improvement in the fitness value.



(a) (b)

Fig. 4. This figure shows close-ups of the tiles from the colour corrected photomosaics: (a) Lotus image and (b) Stop sign image

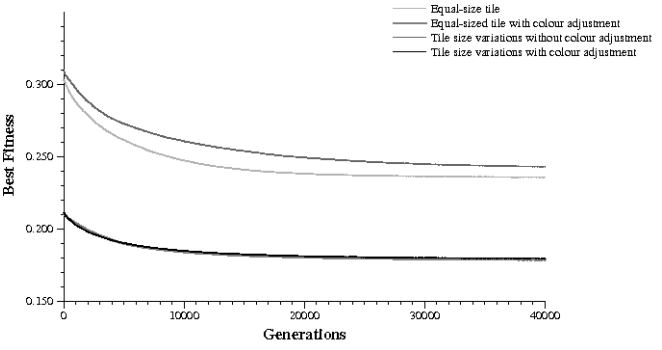


Fig. 5. Best fitness values over 40,000 generations. The curve for equal-sized tile with colour adjustment coincides with the curve for tile size variation with colour adjustment.

5 Conclusions

Based on the two test target images, we observed that colour adjustment generates better photomosaics, in terms of visual quality and fitness, than the equal-sized tile approach. The animation also produces engaging results in which the target subject gradually emerges with a smooth change of colour. This technique could help artist users by easing the burden of having to carefully and tediously select suitable tiles to match a target image. However, strange coloured tiles can be created, as the technique is applied without considering the original colour of the tile.

The tile size variation technique produces different aesthetic values to a photomosaic, as it is able to produce a balance between tile discernibility and subject clarity. Larger tiles easily reveal the subject of tiles in the collection while small tiles are able to accentuate the detail in the target image. The low fitness performance of this technique is not surprising as the fitness function only emphasizes the colour differences with no regard to visual fidelity of a photomosaic to the target image. Use of a combination of our refinement techniques suggests that regardless of tile size, the colour adjustment is the more dominant refinement.

In conclusion, each of these techniques is able to produce an aesthetically different animation effect and represents a better mechanism for generating photomosaics when only a small number of tiles is available. However further investigation is needed to develop the ideas. In future work we hope to improve the fitness function and to generate the quadtree area division dynamically during the evolutionary process. The latter improvement is likely to be able to produce different animation effects.

Acknowledgments

We would like to thank Karen Trist and Marsha Berry for providing the qualitative assessments of the animations and Erik Edespong for help in the implementation of the colour adjustment approach. The target images and tiles for the experiments are from various websites and from the AndreaMosaic collection.

References

1. Tran, N.: Generating photomosaics: An empirical study. In: Proceedings of the 1999 ACM symposium on Applied computing (SAC 1999), pp. 105–109. ACM, New York (1999)
2. Silvers, R., Hawley, M.: Photomosaics. Henry Holt and Company, Inc., New York (1997)
3. Ciesielski, V., Berry, M., Trist, K., D'Souza, D.: Evolution of animated photomosaics. In: Giacobini, M., et al. (eds.) EvoWorkshops 2007. LNCS, vol. 4448, pp. 498–507. Springer, Heidelberg (2007)
4. Wijesinghe, G., Mat Sah, S.B., Ciesielski, V.: Grid vs. arbitrary placement of tiles for generating animated photomosaics. In: Proceeding of Congress of 2008 Evolutionary Computation (CEC 2008). IEEE Service Center, Piscataway (2008) (NJ 08855-1331)
5. Kim, J., Pellacini, F.: Jigsaw image mosaics. ACM Transactions on Graphics (TOG) 21, 657–664 (2006)
6. Orchard, J., Kaplan, C.S.: Cut-out image mosaics. In: Proceedings of the 6th international symposium on Non-photorealistic animation and rendering (NPAR 2008), pp. 79–87. ACM, New York (2008)
7. Di Blasi, G., Gallo, G., Maria, P.: Smart ideas for photomosaic rendering. In: Proceedings of Eurographics Italian Chapter Conference 2006, Eurographic Association, Catania, Italy (2006)
8. Finkelstein, A., Range, M.: Image Mosaic. In: Hersch, R.D., André, J., Brown, H. (eds.) RIDT 1998 and EPub 1998. LNCS, vol. 1375, pp. 11–22. Springer, Heidelberg (1998)
9. Kim, J., Nah, H., Yoon, K.: The Decorative PixMosaics: using directional photo tiles. In: Proceedings of the Computer Graphics, Imaging and Vision: New Trends (CGIV 2005). IEEE Computer Society, China (2005)
10. Zhang, Y., Nascimento, M.A., Zaiane, O.R.: Building Image Mosaics: An application of content-based image retrieval. In: ICME 2003: Proceedings of the 2003 International Conference on Multimedia and Expo (ICME 2003), vol. 3. IEEE Computer Society, Washington (2003)
11. Park, J.W.: Artistic Depiction: Mosaic for Stacktable Objects. In: ACM SIGGRAPH 2004 Sketches SIGGRAPH 2004. ACM, New York (2004)
12. Yan, X., Liu, H., Yan, J., Wu, Q.: A Fast Evolutionary Algorithm for Traveling Salesman Problem. In: Proceedings of the 3rd International Conference on Natural Computation (ICNC 2007), pp. 85–90. IEEE, New York (2007)
13. Liang, K., Yao, X., Newton, C., Hoffman, D.: A New Evolutionary Approach to Cutting Stock Problems With and Without Contiguity. Computers and Operations Research 29(12), 1641–1659 (2002)
14. Mat Sah, S.B., Ciesielski, V., D'Souza, D., Berry, M.: Comparison between genetic algorithm and genetic programming performance for photomosaic generation. In: Li, X., Kirley, M., Zhang, M., Green, D., Ciesielski, V., Abbass, H.A., Michalewicz, Z., Hendtlass, T., Deb, K., Tan, K.C., Branke, J., Shi, Y. (eds.) SEAL 2008. LNCS, vol. 5361, pp. 259–268. Springer, Heidelberg (2008)
15. Trist, K.: Untitled 2008 animated mosaic (still), Project Space Spare Room, RMIT, Melbourne (2008), http://schoolofartgalleries.dsc.rmit.edu.au/PSSR/exhibitions/2008/a_house_a_home.html (accessed November 2, 2009)
16. Woodwark, J.R.: The Explicit quad tree as a structure for computer graphics. The Computer Journal 25, 383–390 (1982)

Generative Art and Evolutionary Refinement

Gary Greenfield

University of Richmond, Richmond VA 23173, USA
ggreenfi@richmond.edu
http://www.mathcs.richmond.edu/~ggreenfi/

Abstract. In considering a case study, we examine the process of promoting emergence and creativity within an evolutionary art system using the technique of evolutionary refinement. That is, for the complex, difficult to predict generative scheme based on a model for simulating cellular morphogenesis that forms the generative component of an evolutionary art system, we discuss how we proceed in stages to develop, analyze, focus, and re-target evolved genetic material for aesthetic purposes — in this instance aesthetic pattern formation.

1 Introduction

We consider the *process* of using a generative art system to evolve images with certain aesthetic characteristics. We emphasize the word *process* because of the ongoing debate about what the term emergence should really mean for a generative art system. Starting with Galanter's definition [8] of generative art as "art practices where the artist creates a process that acts with some degree of autonomy to create all or part of an artwork," Monro [15] summarizes the controversy about the meaning of the word emergence by considering: (1) categorical emergence, (2) simple-to-complex emergence (3) many-agent emergence, (4) difficulty-of-prediction emergence, (5) emergence relative to a model, (6) surprising emergence, and (7) Frankenstinean emergence before providing his own definition of emergence as it pertains to generative art to be a combination of the "difficult-to-predict" notion together with the notion of evoking "surprise-wonder-mystery-autonomy". By allowing the word "autonomy" to reappear, and by coupling it with "surprise", Monro not only comes full circle, but further confounds matters by encroaching on the discipline of [artificial] creativity (compare Burns [1], Dorin and Korb [5], Jacobsen [12], Ross et al. [16], Saunders and Gero [17], and Schmidhuber [18]).

If automated fitness is used as an integral part of a generative art system, we concur that emergence and creativity are goals, but we feel that formulating a universal criterion for either of these concepts is probably a hopeless cause. Therefore, in this paper, we consider a more pragmatic approach to automated fitness — evolutionary refinement, by which we mean breeding strains of genetic material that can be analyzed from different points of view, so that we might better gauge what evolutionary paths to pursue. More generally, given a complex, difficult-to-predict generative component of an evolutionary art system, we

C. Di Chio et al. (Eds.): EvoApplications 2010, Part II, LNCS 6025, pp. 291–300, 2010.

consider what strategies one might adopt to coax aesthetic output from the system. In adopting an evolutionary refinement approach, we therefore champion "emergence in context" after McCormack and Dorin [14] rather than emergence by "surprise" as expounded by Edmond et al. [6]. By necessity, our discussion is tied to a specific example.

Because our generative component is both intricate and complex, we shall only give a cursory overview of it. It was chosen because we have reason to believe that only by carefully selecting values for a large number of nonintuitive parameters will it yield interesting imagery. Moreover, for our generative method it appears to be difficult to extract meaningful measurement variables to help identify aesthetic content. Figure 1 gives three examples that are typical of the imagery we find uninteresting that our system tends to evolve.

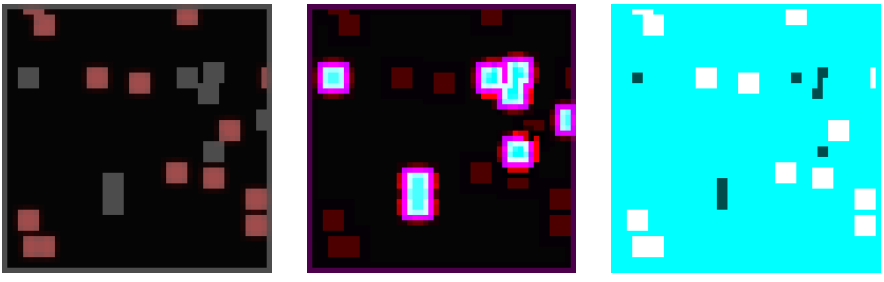


Fig. 1. Three lackluster images evolved from our generative art system based on cellular morphogenesis simulation

2 Our Cellular Morphogenesis Evolutionary Art System

Our generative scheme is based on simulating cellular morphogenesis. The model is due to Eggenberger [7]. Its adaptation to evolutionary art is due to Greenfield [9] [10] [11]. An image obtained from the scheme is a visualization of a matrix of simulated cells. The cells are initialized with concentrations of four simulated substances that represent transcription factors or TF's. The visualization is realized by designating three TF's to serve as components for RGB colors. The remaining colorless TF plays an important but indirect role. The scheme is complicated. The basic idea is that the concentrations of the TF's determine how the genes are expressed which in turn determines how the concentrations of the TF's change. Simulating cell development requires an update cycle consisting of a gene expression calculation followed by a TF concentration calculation. This update cycle is performed for all cells for a fixed number of iterations.

A cell genome (see Figure 2) consists of four gene *units*. A gene unit consists of two regulatory genes and a structural gene. Formally, we define a *gene* to be a string of eight digits $g_0 g_1 \ldots g_7$. The last digit g_7 is designated as the *marker* for the gene. Marker digits may assume any of the values zero through

six while the values for the remaining digits are constrained to lie in the range zero through four. When examining the digits of a gene, either we extract an *offset*, $o = g_0 + g_1 \bmod 3$, a *diffusion coefficient*, $d = (g_2 + g_3)/9$, and a *type*, $t = g_4 + g_5 + g_6 \bmod 5$, or we extract the following three base five constants $k_0 = (g_0 g_1 g_2 g_3 g_4)_5$, $k_1 = (g_1 g_2 g_3 g_4 g_5)_5$, and $k_2 = (g_2 g_3 g_4 g_5 g_6)_5$.

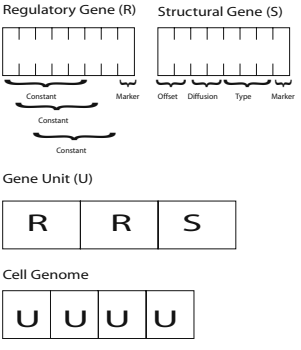


Fig. 2. The genome structure

In order to determine whether a structural gene S within a gene unit is currently expressed, or *active*, we begin by considering the role played by one of its regulatory genes R_j located in the regulatory segment of the gene unit together with the cell's i-th TF. Using the offset o of S, we extract the five digit string from R_j beginning at position o, perform a base five conversion to obtain the constant k_o, and then subtract the result from the *weight* w_i associated with the i-th TF. This gives the *affinity* f_i of R_j for the i-th TF. The weight w_i is an environmental attribute that is held constant for all cells. Note that by an appropriate choice for the weight w_i, f_i will be a *signed* quantity. We multiply the affinity f_i by the concentration c_i of the i-th TF. Keeping R_j fixed, we sum over all TF's to obtain the activity level r_j for R_j. Thus

$$r_j = \Sigma_i f_i c_i. \tag{1}$$

Next, by letting j vary, we sum r_j over all regulatory genes to associate to the structural gene S the activation measurement,

$$a = \frac{1}{1 + \exp(\Sigma_j r_j)}. \tag{2}$$

Finally, we threshold this result to obtain an activity level γ for S, by letting

$$\gamma = \begin{cases} -1.0, \text{ if } a < 0.2 \\ +1.0, \text{ if } a > 0.8 \\ \quad 0.0, \text{ otherwise.} \end{cases} \tag{3}$$

For convenience, we say the structural gene S is activated in an *excitory* state if $\gamma = +1$, activated in an *inhibitory* state if $\gamma = -1$, and not activated otherwise.

When activated, the concentration of the TF specified by the structural gene's type is increased (respectively decreased) by a fixed amount Δ. Moreover, if the concentration of the colorless TF is sufficiently high then the diffusion coefficient will be used to distribute that change over neighboring cells. Note that by assigning meta-functionality to gene expression, researchers have used variations of this formulation to model 2-D and 3-D cellular growth and differentiation (see Eggenberger [7] and Chavoya and Duthen [2]).

3 Evolutionary Exploration Using Fitness Functions

We say a cell is dormant if its TF_0, TF_1, and TF_2) concentrations (i.e. its RGB color components) are all at, or below, trace levels whence the cell will appear black in the final cell pattern visualization. Let μ_i and σ_i denote the mean and standard deviation calculated over all non-dormant cells in the matrix of the concentration of transcription factor TF_i. Let N_a denote the number of cells for which at least one structural gene underwent a change in activation status during the final development cycle.

For initial exploration purposes we experimented with the fitness functions listed in Table 1. For all of these functions we maximize fitness. The matrix template we used placed a substrate of cells in a 61×61 matrix together with a mix of 22 small blocks of two other strains of cells. The genomes for all three cells were initialized by being pseudorandomly generated, and the template also fixed the 22 pseudorandom location for the blocks. By randomizing the underlying template pattern for each evolutionary run, we were able to promote explorations of various kinds of cell interactions. The development period lasted for 400 cell update cycles. The 22 blocks of cells and the perimeter cells of the substrate were all given trace concentrations of all of the TF's to initiate the development. A population of size eight was allowed to evolve for five generations with crossover and mutation operators for the three cell genomes specifying each individual implemented by treating each of the four gene units defining the cell strain as linear arrays. None of these simulation paymasters were tuned or optimized. For the most part they were carried over from [10].

The first four fitness functions in Table 1 produced undistinguished imagery similar to that shown in Figure 1. The use of σ_i in the first three caused many

Table 1. Table of fitness functions used during the exploratory phase

Function	Formula
f_1	$\mu_1\sigma_1 + \mu_2\sigma_2$
f_2	$\mu_1 + \mu_0\sigma_0$
f_3	$\mu_1\sigma_3$
f_4	$\lvert\mu_2 - \mu_1\rvert$
f_5	$N_a\lvert\mu_2 - \mu_0\rvert$

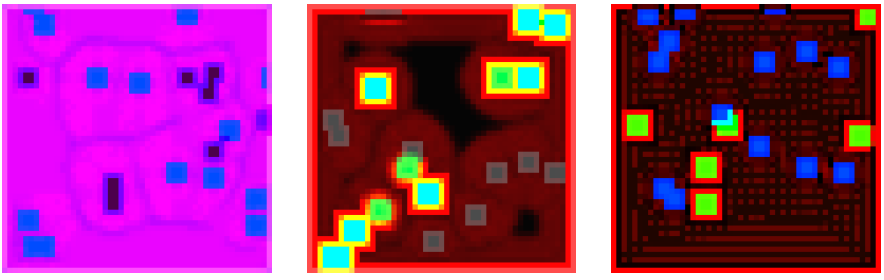


Fig. 3. Three evolved examples using fitness function f_5 from Table 1 that provide source genetic material for further study

cells to remain inactive so that the gradient of the i-th TF would be maintained. In the case of f_4, the black from never activated cells predisposed one of the μ_i to remain close to zero. However the last fitness function, $N_a|\mu_2 - \mu_0|$, which tries to maximize the *difference* between the averages of the red and blue components over all active cells while simultaneously maximizing the number of active cells, evolved the three examples shown in Figure 3. These three examples furnished the promising source genetic material we chose to analyze further.

4 Analysis of the Source Material

Figure 4 shows visualizations corresponding to each of the examples of Figure 3 where the substrate cells alone were used in the matrix, and the development period was extended to 800 update cycles. We observe that the substrate of the second example develops slowly, and the substrate of the third example exhibits unusual activation and inhibition feedback.

Corresponding to the first two examples of Figure 3, Figure 5 shows the development that results when only one of the two available overlay cell strains is used as a large cental block and the development period is extended to 1000 update cycles. In fact, the other available cell strain also exhibits a similar interference

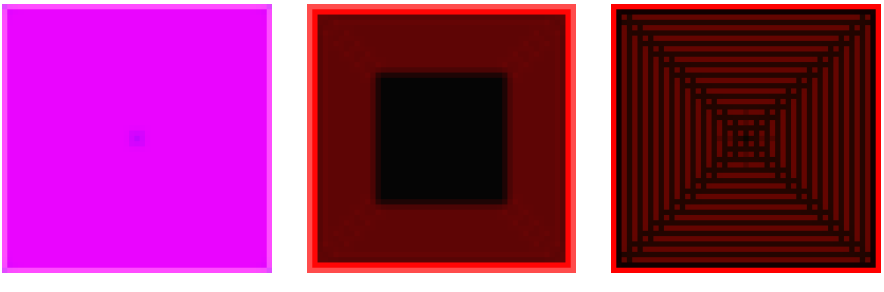


Fig. 4. Visualizations of the cell substrates for each of the three source material examples

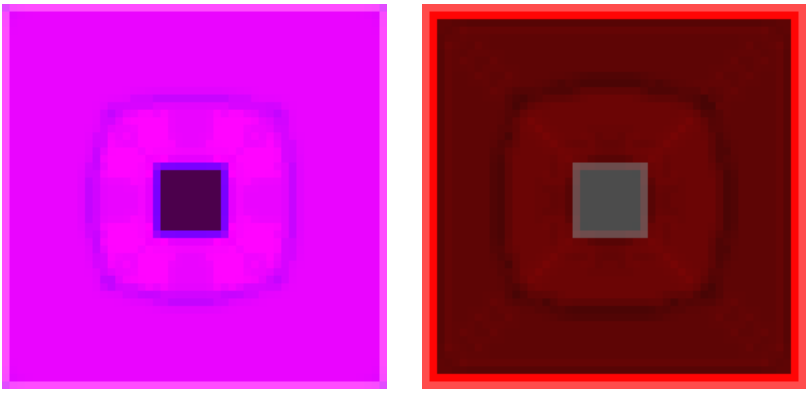


Fig. 5. The interference pattern interaction between the substrate and one of the two available cell strains is highlighted

pattern. On the other hand, as Figure 6 shows, for the third example only one of the two available cell strains interacts with the substrate. We also tried interchanging the roles played by the substrate cells and overlay cells and mixing cell strains from different examples, but no useful visualizations or information was obtained.

For completeness, we remark that our evolved phenomena are heritable. Using our source genetic material to seed the entire population and using high mutation rates but almost no crossover, Figure 7 shows examples corresponding to Figure 3 that were evolved during three separate evolutionary runs. Although the substrates did not exhibit any visually significant differences, new cell strains for the interior were successfully evolved.

5 Evolutionary Refinement of the Source Material

Researchers have always been keenly interested in models for formulating textures as well as skin patterns found in nature. Because the middle example from Figure 3 reminded us of the examples in Young's seminal paper [19] for simulating skin patterns based on a theory of Turing, in view of the nature of the interference boundary found in Figure 5, we elected to re-target evolution to exploit this interference phenomenon for aesthetic purposes.

To accomplish this, besides activating the perimeter of the substrate, in order to anchor the pattern we also activated four small blocks of the substrate at the vertices of a centered square one-fourth the area of the template. Our task now was to find good *locations* for ten activated blocks of the overlay cell strain that would yield complex interference patterns. Because of the color gradients involved, it is difficult to identify the interference boundary, and because we chose not to introduce target matching (see below), we evolved interesting interference patterns my maximizing (respectively minimizing) the total concentration

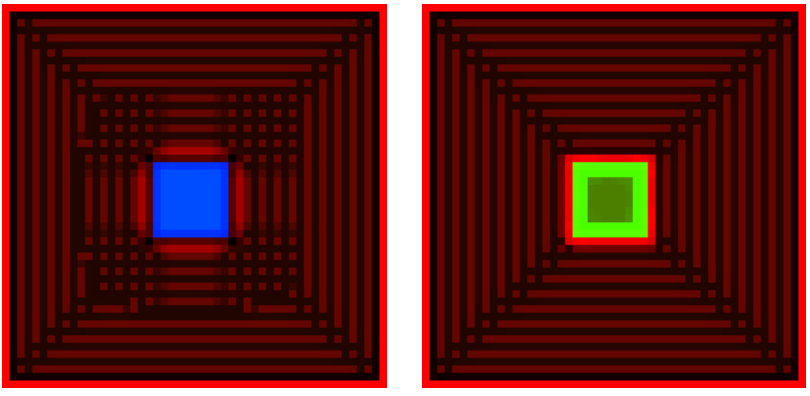


Fig. 6. For the third source example, visualizations showing that only one of the two available cell strains interacts with the substrate

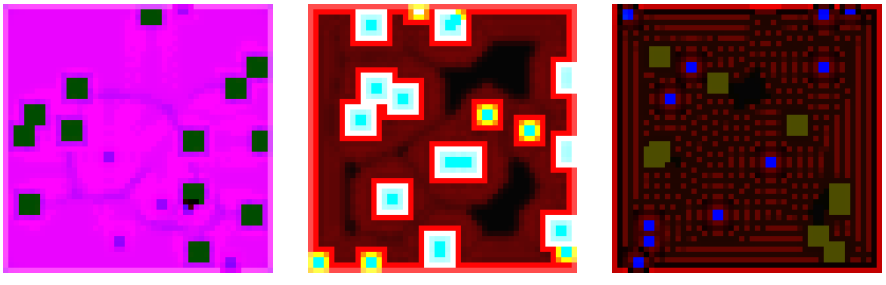


Fig. 7. Evolutionary runs showing that the genetic source material evolved is heritable

of the red transcription factor TF_0 in a centered square one-half the area of the template. Figure 8 show the results from two small test runs lasting only five generations using a population of size eight. The maximization example reveals an artifact exploited by simulated evolution that occurred by embedding activated overlay cells inside activated substrate cells. The minimization example gives rise to better definition of the interference boundary. Thus we increased the population size to twenty and ran the evolution for fifteen generations to obtain the evolved, refined, red-minimized image shown in Figure 9.

A natural question that arises is: At the refinement stage, why didn't we opt to target evolution by comparing our evolved patterns to existing patterns in, say, an image database? Certainly such image targeting has become popular. The reason we chose not to do this is because this approach is not very enlightening. Consider, for example, the work of Machado, Romero and Manaris [13] where sophisticated tools such as neural nets and classifier systems are used to anti-target images and the result that is reported is evolution of styles; or the work of Colton and Torres [4], where clever data management allows image targeting to be used to evolve image filters. Although they have slightly different

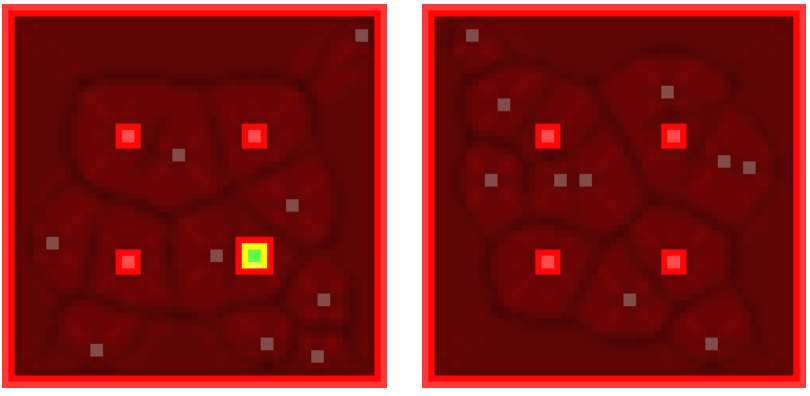


Fig. 8. Creating an interference pattern by maximizing (left) and minimizing (right) the concentration of red in a centered square of half the total area

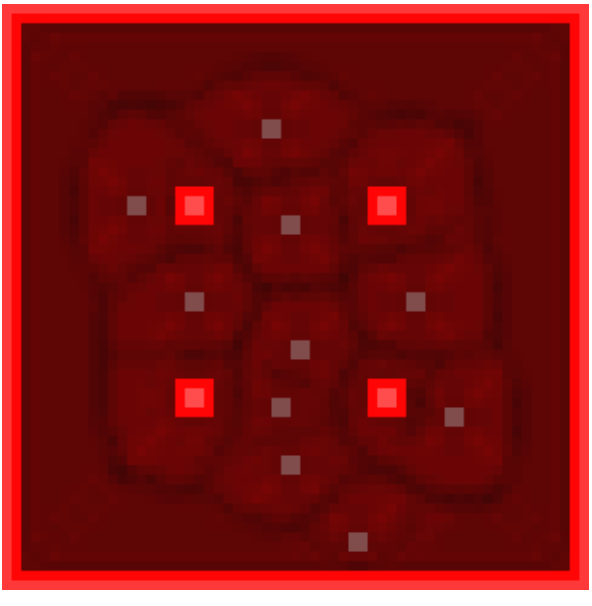


Fig. 9. Our final evolved, refined, red-minimized cellular morphogenesis interference pattern

goals — sustaining novelty in the case of [13] and reproducing existing filters in the case of [4] — we would ask what new insights or understanding about evolutionary computation are being achieved? While it may be that other researchers use refinement approaches similar in spirit to the one described above but do not report about it, we believe out case study provides a valuable insight into how one might approach the exploration problem in certain evolutionary

computation contexts. Namely, that for *complex* generative systems used in evolutionary art, as the genetic material evolves, the nature of the environment (e.g. here, substrate activation conditions and reduction of the number of other strain blocks) together with the fitness objectives (e.g. here, red minimization over a subset of the cells) may need to change in order to realize emergence or creativity. More broadly, we feel the lesson to be learned from our case study is that for sufficiently *complex* genome representations, evolution in stages with radical changes in fitness criteria may be a profitable evolutionary exploration strategy.

Given what we are proposing, a devil's advocate question worth considering is: Should our end-product image in Figure 9 now be further refined or perhaps even [human] genetically engineered in order to remove the lone gray block at the bottom that falls outside the central square where minimization took place? Unfortunately, we have no satisfactory answer to this question.

6 Conclusion

It is difficult to imagine how automated fitness functions could provide an evolutionary trajectory from images such as those found in Figure 1 to our final evolved image in Figure 9. Perhaps the approach that is most similar to ours is by Colton [3] who uses an inference engine, as opposed to what we have called refinement, to produce new rules for fitness functions. In the domain of evolutionary art, neither his scene generation application nor our pattern evolving application generalize well, but what we have shown in our case study is that by progressing in stages, analyzing the results, and using deductive reasoning we can achieve some measure of emergence and creativity through the process of evolutionary refinement. We expect that further advances towards automating evolutionary art will require just such an approach.

References

1. Burns, K.: Atoms of EVE: a Bayesian basis for aesthetics of style in sketching. Artificial Intelligence for Engineering, Design, Analysis and Manufacturing 20, 185–199 (2006)
2. Chavoya, A., Duthen, Y.: An artificial development model for cell pattern generation. In: Randall, M., Abbass, H.A., Wiles, J. (eds.) ACAL 2007. LNCS (LNAI), vol. 4828, pp. 61–71. Springer, Heidelberg (2007)
3. Colton, S.: Automatic invention of fitness functions with applications to scene generation. In: Giacobini, M., Brabazon, A., Cagnoni, S., Di Caro, G.A., Drechsler, R., Ekárt, A., Esparcia-Alcázar, A.I., Farooq, M., Fink, A., McCormack, J., O'Neill, M., Romero, J., Rothlauf, F., Squillero, G., Uyar, A.Ş., Yang, S. (eds.) EvoWorkshops 2008. LNCS, vol. 4974, pp. 381–391. Springer, Heidelberg (2008)
4. Colton, S., Torres, P.: Evolving aspproximate image filters. In: Giacobini, M., et al. (eds.) EvoWorkshops 2009. LNCS, vol. 5484, pp. 467–477. Springer, Heidelberg (2009)

5. Dorin, A., Korb, K.: A new definition of creativity. In: Korb, K., Randall, M., Hendtlass, T. (eds.) ACAL 2009. LNCS, vol. 5865, pp. 11–21. Springer, Heidelberg (2009)
6. Edmond, R., Sipper, M., Capcarrére, M.: Design, observation, surprise! A test of emergence. Artificial Life 5(3), 225–240 (1999)
7. Eggenberger, P.: Evolving morphologies of simulated 3d organisms based on differential gene expression. In: Proceedings of the Fourth European Conference on Artificial Life (ECAL 1997), pp. 205–213. Springer, Heidelberg (1997)
8. Galanter, P.: What is generative art? Complexity theory as a context for art theory. In: Soddu, C. (ed.) Proceedings of the Sixth International Conference and Exhibition on Generative Art (2003)
9. Greenfield, G.: Abstract art from a model for cellular morphogenesis. In: Sarhangi, R., Moody, R. (eds.) BRIDGES 2005 Conference Proceedings, pp. 137–142 (2005)
10. Greenfield, G.: Genetic learning for biologically inspired aesthetic processes. International Journal on Artificial Intelligence Tools 15(4), 577–598 (2006)
11. Greenfield, G.: The void series – generative art using regulatory genes. In: Soddu, C. (ed.) Proceedings of the Seventh International Conference and Exhibition on Generative Art, Alea Design, vol. 1, pp. 70–77 (2004)
12. Jacobsen, T.: Bridging the arts and sciences: a framework for the psychology of aesthetics. Leonardo 39(2), 155–162 (2006)
13. Machado, P., Romero, J., Manaris, B.: Experiments in computational aesthetics. In: Romero, J., Machado, P. (eds.) The Art of Artificial Evolution, pp. 381–415. Springer, Heidelberg (2008)
14. McCormack, J., Dorin, A.: Art, emergence and the computational sublime. In: Second Iteration (CDROM), Melbourne Centre for Electronic Media Arts. Monash University (2001)
15. Monro, G.: Emergence and generative art. Leonardo 42(5), 475–477 (2009)
16. Ross, B., Ralph, W., Zong, H.: Evolutionary image synthesis using a model of aesthetics. In: 2006 IEEE Congress on Evolutionary Computation, pp. 3832–3839. IEEE Press, Los Alamitos (2006)
17. Saunders, R., Gero, J.: Artificial creativity: a synthetic approach to the study of creative behavior. In: Gero, J. (ed.) Creative Design V, Key Centre of Design Computing and Cognition (2002)
18. Schmidhuber, J.: Simple algorithmic principles of discovery, subjective beauty, selective attention, curosity and creativity. In: Corruble, V., Takeda, M., Suzuki, E. (eds.) DS 2007. LNCS (LNAI), vol. 4755, pp. 26–38. Springer, Heidelberg (2007)
19. Young, D.: A local activator-inhibitor model of vertebrate skin patterns. In: Wolfram, S. (ed.) Theory & Application of Cellular Automata, pp. 320–327. World Scientific, Singapore (1986)

Aesthetic Learning in an Interactive Evolutionary Art System

Yang Li* and Chang-Jun Hu

School of Information Engineering, University of Science and Technology Beijing, Beijing, China
mailbox.liyang@gmail.com, huchangjun@ies.ustb.edu.cn

Abstract. Learning aesthetic judgements is essential for reducing the users' fatigue in evolutionary art system. Although judging beauty is a highly subjective task, we consider that certain features are important to please users. In this paper, the aesthetic preferences are explored by learning the features, which we extracted from the images in the interactive generations. In addition to color ingredients, image complexity and image order are considered highly relevant to aesthetic measurement. We interpret these two features from the information theory and fractal compression perspective. Our experimental results suggest that these features play important roles in aesthetic judgements. Our findings also show that our evolutionary art system is efficient at predicting user's preference.

Keywords: Evolutionary art, interactive evolutionary computation, image complexity, fractal compression.

1 Introduction

The concept of Evolutionary Art originated with Dawkins' Biomorphs [1]. Subsequently, the efforts of Sims [2] became the significant work for generating computer graphics. Genetic Programming (GP) was used in his work to evolve aesthetic images. Following the idea of Sims, symbolic expressions were used widely in research (see, for example, [3] [4]). Due to its unique characteristics as an intelligent and creative activity, most systems rely on Interactive Evolutionary Computation (IEC), which the user has the tedious task of making decisions for every generation. Therefore, one of the significant challenges is to reduce user's fatigue [5] [6].

The specific aspect of the problem we are interested in is to explore user's criteria for beauty in their aesthetic judgements. However, it is a highly subjective task. Firstly, there is no unanimous measurement for the standard of

* This work is supported by the Hi-Tech Research and Development Program (863) of China No. 2008AA01Z109 and the Centre of Excellence for Research in Computational Intelligence and Applications (CERCIA) at the University of Birmingham, UK.

C. Di Chio et al. (Eds.): EvoApplications 2010, Part II, LNCS 6025, pp. 301–310, 2010.

aesthetic value. Different people have different principles and appreciation of beauty. Secondly, sometimes it is hard to tell exactly what features are attractive to people. Therefore, we consider learning and simulating user's behavior in aesthetic judgments very important in the evolutionary process.

We show in this paper that there exist certain properties and features, which underlie the individual's judgement of beauty. Instead of making an ambiguous definition of aesthetics, we applied a statistical learning approach to analyze the features in these evaluated images generated by evolutionary process. The main research goals are to: (a) find features which are relevant to aesthetic measurement; (b) explore the relations in these features which finally quantitatively influence the aesthetic score.

To achieve these goals, we try to understand the aspects that appeal to people by using a computational approach. Based on Birkhoff's work [7], we consider order and complexity of the image both significant features for the aesthetic measures. In this paper, we aim to estimate (a) image complexity by using information theory and (b) the order of the image by using fractal compression. In addition, we build (c) a classifier to learn user's behavior in judging beauty.

The proposed measurement has been tested with sets of images generated from several independent experiments by our evolutionary art system. In our system, four mutation parameters are applied for users to manipulate. Fitness values are assigned in every step in evolutionary process, which are then applied for the training sets. Then we use the classifier to explore these features that we extract from the sets. Our experimental results show that our system is able to properly classify high valued and low valued images by learning from the interactive run.

The paper is organized as follows: In Section 2 we describe our method to estimate the image complexity and image order. Next, the experimental results and their analysis are included in Section 3. Finally, in Section 4 we draw conclusions and point directions for future research.

2 Features for Measuring Aesthetics

As we mentioned before, two important factors are addressed in measuring aesthetics. First, we estimate the image complexity from an information theory perspective. The reason why we consider informational aesthetics measurement is described by Rigau et al. [8]. In our work, we use different channels to calculate the information content. Second, we estimate the order of the image based on the low-complexity art theory by using fractal compression [9]. This theory is based on the minimum description length (MDL), which is related to "capturing the essence of the image".

2.1 Image Complexity Estimation

Our measurements of image complexity are based on the concepts of information theory. The entropy of each channel in HSL is calculated, which is considered

an estimate of the image complexity in the color space. To better characterize the complexity from different aspects, the same metrics are also applied to RGB and Y_{709}.

Assuming the image's information content represents the complexity, it is calculated by multiplying each pixel's Shannon entropy and the number of pixels. Shannon entropy of each pixel is computed from the intensity histogram for every channel. An intensity histogram for every channel is defined in discrete information bins where the input χ represents the bins of the histogram, with probability distribution $p_i = \frac{n_i}{N}(0 \leq i \leq \chi)$, in which n_i is the number of pixels in bin i. The value of χ is different according to the channel. In our case, χ for hue, saturation and lightness channel are 360, 100 and 100 respectively. Then the information content for hue channel is computed as follows:

$$Complexity_{hue} = N \times (-\sum_{x \in X} p_{hue}(x) log p_{hue}(x)), \tag{1}$$

The image complexity for saturation and lightness are also calculated in the same way.

Additionally, we also calculate the information channel based on RGB representation, and Y_{709} which is the luminance from linear red, green and blue. We consider the information of colorfulness in RGB channel and the luminance of Y_{709} very important factors. One problem is the range in RGB is 256^3, which is time consuming for our on-line system. Therefore, we defined a fast and robust method to compute the RGB color distribution. We divide the RGB color space into 512 cubes with eight equal partitions along each dimension in RGB. The intensity histogram is then reduced to 512 X_{RGB} bins. The χ for the channel Y_{709} is 256. Probability distributions of the variables X_{RGB} and $X_{Y_{709}}$ are used to calculate the entropy of RGB and Y_{709} channel.

2.2 Image Order Estimation

Order is a measure for the number of regularities found in the image [10]. We assume beauty is defined by a human because they can transform the visual scenes to their familiar knowledge. The human visual system can be described as a coding algorithm. This system could transform the data from art to their internal representation. In other words, an image which can be computed by an algorithm in the shortest description is considered the most beautiful in a set of images.

Schmidhuber derives a formula for measuring beauty: [9]

$$-logP(i \mid C) = -logP(C \mid i) + logP(C) - logP(i). \tag{2}$$

where C is the coding algorithm simulating human's perception system. i is the image selected from a set of images. $P(C)$ is a constant when C is given. $P(i)$ is the priori distribution of the images. $-logP(C \mid i)$ is the information required to compute C from i. Then, $P(i \mid C)$ is the conditional probability of selecting i from a set of images given the coding algorithm C. We aim to

maximize $P(i \mid C)$, which can be interpreted as finding the most beautiful image. Therefore, it corresponds to minimizing the information to describe i given the coding algorithm C.

Two important problems are addressed in this definition. The first problem is to devise a good coding algorithm. Fractal compression method is applied in our system, which has been found close to human visual system [11]. The second problem is estimating the information to describe the image using this algorithm.

We assume that the minimum information to calculate C from i is to estimate how fast the coding algorithm could process the image. Here, we could develop a formula to evaluate the information to compute C from i,

$$-logP(C \mid i) = \frac{CompressionRatio}{t_e - t_s}. \tag{3}$$

in which, t_e and t_s are the end-time and start-time of fractal encoding. In order to calculate this evaluation, we first average the RGB pixels into grey ones, then the image is divided into four equal regions. The fractal algorithm encodes each region based on its self-similarity. All the images are rendered at 64 by 64 resolution. Therefore the fractal encoding is not very time-consuming.

In our method, training images are assigned by one of two fitness value: high or low. The priori distribution of the images, $-logP(i)$, is assigned to 1 or 0 which represents high or low value. $logP(C)$ is disregarded. According to our definition, the order of the image is calculated as follows:

$$-logP(i \mid C) = \frac{CompressionRatio}{t_e - t_s} - logP(i). \tag{4}$$

3 Experimental Results

3.1 Experimental Setup

In this section we setup a system to learn user's behavior in IEC. Two kinds of experiments are designed for this purpose. Firstly, we focus on one single experiment from the first till the last iteration, paying particular attention to how these features influence the aesthetic judgement. Secondly, five independent experiments based on five different people also show that our model could predict properly the aesthetic and non-aesthetic images according to users' preferences. We performed 21 iterations in the first single experiment. The settings for other independent experiments are the same except *number of generation* and *mutation rates*, which are set manually according to different users.

Genetic Programming. Like most of the evolutionary art system, we use GP as our representation. The settings for GP are presented in Table 1. 87 images are shown in every generation, all of which are rendered at 64 by 64 resolution. The functions that we use in GP fall into three categories: unary, binary and ternary functions. The terminals are variable x, y or constants. The maximum initial tree depth is 6.

Table 1. Parameters of GP for representation

Parameter	Setting
Population size per generation	87
Number of generation	21
Mutation operator	coarse, color, pattern and fine mutation
Mutation Rate	manually set every generation, in the range 0-100
Unary function set	sin, cos, tan, abs, floor, ceil, sqr, square, cube, log, exp, negative, spiral, circle
Binary function set	+,-,*,/, max, min, pow, average
Ternary function set	if, lerp
Terminal set X,Y,	scalar and random constants
Initial maximum tree depth	6

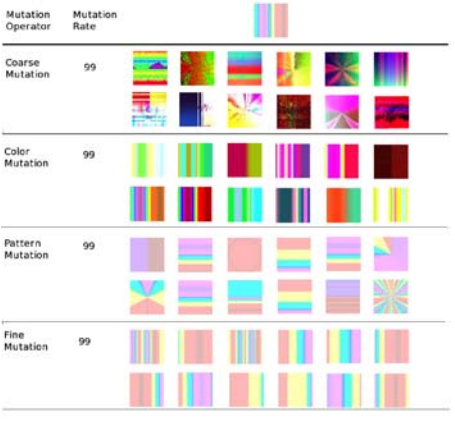


Fig. 1. Some examples of the first generation according to different mutation operators

Mutation Operators. In our system, only one parent will be selected in every iteration. Thus mutation operators are applied, which are the most important genetic operators for the diversity of images. Mutation is performed by changing values in a leaf node or functions in an internal node, deleting the subtree at any point or replacing the pruning part with a new random subtree. We resort to four kinds of mutation rate according to our genotype, which are coarse mutation rate, color mutation rate, pattern mutation rate and fine mutation rate. They represent the probability of changing the subtree, the node and the value. In Figure 1, the parent is the image on the top of the figure, examples of mutation populations generated by different mutation operators with corresponding mutation rates are shown below. These four specific operators are applied for better exploring the solutions in the design space.

Evolutionary and Learning Process. Here, we briefly describe how our system works:

1. A set of initial images are rendered by our system.
2. Current images are assigned by two levels of fitness values according to his/her preference. One of them is selected as the parent of the next generation. Mutation parameters are also set manually. Four mutation rates are set to manage the degree of the stylistic changing.
3. Features are extracted from the current populations and stored in a feature list.
4. The evolutionary process stops when a termination criteria is met. In our case, the number of generations is fixed in order to collect enough training samples.
5. The classifier is used to train the dataset collected from the previous generation. It is trained to distinguish between the higher valued and the low valued images.

Features are extracted from images of every iteration. The features that we used for our experiments are based on color ingredients, image complexity and image order. These characteristics are considered to be aesthetically relevant. The three categories are shown in Table 2. This process yields a total of 12 metrics for the whole image.

Table 2. Aesthetic Features Summary

Category	Description
Color Ingredients	Average value and standard deviation of Hue, Saturation and Lightness
Image Complexity	Information entropy of HSL, RGB and Y_{709}
Image Order	Low complexity based on fractal compressor

Finally, decision tree is used to train the samples collected from previous iterations, which is implemented by Weka J48 [12] . Default settings are used in this classifier. Decision tree is used because we could have a better understanding of the user's aesthetic judgement by analyzing it.

3.2 Aesthetic Learning

Accuracy of Prediction. Figure 2 shows the correctly classified accuracy of all the images, high valued and low valued images respectively. The results prove that our system could successfully classify aesthetic and non-aesthetic images since the eighth generation. It also shows that the low valued images have higher accuracy than the high valued ones, because it is easier to distinguish less aesthetic ones from a set of images as there are fewer satisfactory ones. In the first few generations, it is difficult to satisfy user because initial populations are random and abstract. The user's preference becomes clearer after several generations. Therefore, the accuracy of predicting aesthetic ones highly increases in the later generations.

Table 3 shows the numbers of images in training set and test set, the rate of correctly classified images, mean absolute error (MAE) and root mean squared

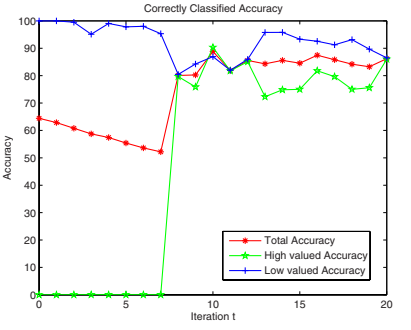


Fig. 2. Correctly classified accuracy of all the images, high valued and low valued images

error (RMSE) according to different generations. The last one shows the accuracy of the correctly classified images which we used a 10-fold cross validation. All the results show that our system is capable of learning user's behavior in the evolutionary process.

Decision Tree. The decision tree provides us with more important information on how aesthetics can be related to these features. As noted above, the result of using 10-fold cross validation in Table 3 was obtained from 1893 datasets with 1249 high valued images and 644 low valued ones. The rules for final aesthetic judgement can be demonstrated by the decision tree created by C4.5(J48). Partial view of the decision tree is introduced in Figure 3. The decision nodes are denoted by circles which split the trees during the process, while the leaf nodes are denoted by squares. The number followed by value 10 or value 0 is the number of correctly classified high valued and low valued images, sometimes the second number represents the number of instances incorrectly classified by the node.

The size of the tree is 25 with 13 leaf nodes. The tree provides us with important information on how these features are reflected to the aesthetic judgement. Let's discuss some interesting decision paths in this tree. The features denoted by image complexity from saturation (5_HpSaturation in Figure 3) and image order (Low Complexity) appear to be quite crucial roles in this decision process. This indicates that the user's judgement is mainly focused on the information gained from saturation channel and the order of the image.

Generation Results. In this single experiment, the subject manipulated 21 generations. 20 individuals that are chosen as the parents are shown in Figure 4. It shows that the user focused on exploring the color in the first few interactions, and then the pattern of the image is changed from the shape of an insect to the shape of an old man. It also explains why the accuracy increases remarkably after the 7th generation.

Table 3. Correctly classified images according to different generations

Iteration	Train Set	Test Set	Correctly Classified Rate	MAE	RMSE
0	86	1807	64.4162%	0.3592	0.5897
1	172	1721	62.8704%	0.3788	0.5921
2	258	1635	60.7951%	0.3946	0.6185
3	344	1549	58.7476%	0.4141	0.6301
4	430	1463	57.4153%	0.4218	0.635
5	516	1377	55.4103%	0.4454	0.6654
6	602	1291	53.6019%	0.4644	0.6712
7	688	1205	52.1992%	0.4769	0.6821
8	774	1119	80.0715%	0.2143	0.4295
9	860	1033	80.2517%	0.2086	0.4397
10	946	947	88.8068%	0.1347	0.3253
11	1032	861	81.8815%	0.1948	0.4103
12	1118	775	85.5484%	0.1681	0.3691
13	1204	689	84.3251%	0.1819	0.3915
14	1290	603	85.5721%	0.1623	0.3764
15	1376	517	84.5261%	0.1622	0.3829
16	1462	431	87.471%	0.1318	0.3543
17	1548	345	85.7971%	0.159	0.359
18	1634	259	84.1699%	0.1815	0.3769
19	1720	173	83.237%	0.1832	0.402
20	1806	87	86.2069%	0.1627	0.3665
Total	10-fold cross validation		87.3217%	0.1408	0.3426

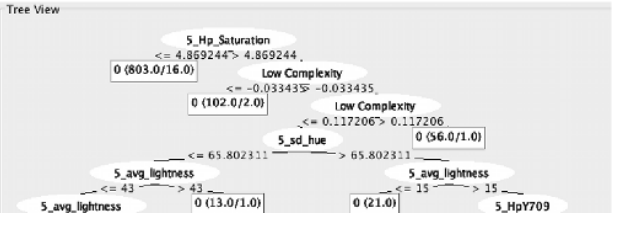


Fig. 3. Decision tree

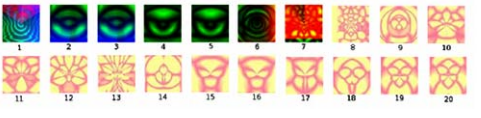


Fig. 4. 20 images which are selected by the subject as the parents

Individuals from the last generation are shown Figure 5. We find that only few images are quite different from the others, because a small proportion of coarse mutation is still allowed to prevent the stagnation in the evolutionary process.

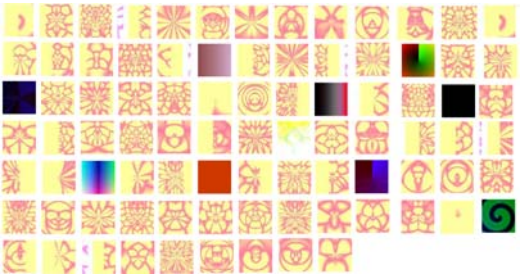


Fig. 5. 87 individuals in the 21st generation

3.3 Validation Study

We have conducted five experiments using our system to explore users' behavior. This test is applied to five different individuals. Our goal is to validate our system by showing the accuracy of correctly classified images along with generations.

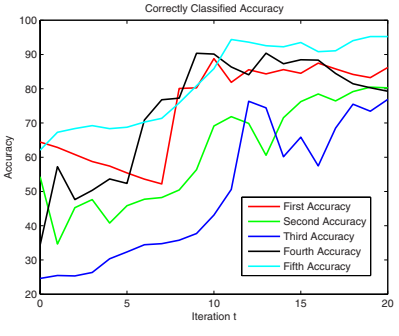


Fig. 6. Five experiments conducted by five different individuals

Figure 6 shows the results achieved by our system. Most of the experiments are able to classify properly between high valued and low valued images, attaining average success rate above 70%.

4 Conclusions and Future Work

This paper explores user's behavior of judging beauty in evolutionary art system. The two important features that we extracted from the generated images are image complexity and image order. We estimate them from the perspective of information theory and fractal compression. With several features, the classifier can distinguish low valued images from high valued ones. This helps us not only understand more about the factors that underlie the individual's judgement of beauty, but also explore more aesthetic criterias in computational aesthetic.

We have carried out a number of statistical and empirical studies to evaluate our computational aesthetic estimation. The results show that it is possible to study the trends of people's preferences that lead to higher or lower scores with these significant features. The future work of this research includes several tasks. First, more features that are relevant to aesthetics could be included to make a significant improvement. Second, exploring the relations among the mutation operators, the features and the fitness values may also help to analyze a specific style in generated images. Third, how to automate the aesthetic decision should be explored.

References

1. Dawkins, R.: The Blind Watchmaker. Harlow Longman (1986)
2. Sims, K.: Artificial Evolution For Computer Graphics. In: Proc. of the 18th Annual Conference on Computer Graphics and Interactive Techniques SIGGRAPH 1991, pp. 319–328. ACM Press, New York (1991)
3. Lutton, E.: Evolution of Fractal shapes for artists and designers. International Journal on Artificial Intelligence Tools 15(4), 651–672 (2006)
4. Machado, P., Cardoso, A.: All the Truth About NEvAr. In: Corne, D.P., Bently (eds.) Applied Intelligence, Special issue on Creative Systems, vol. 16(2), pp. 101–119. Kluwer Academic Publishers, Dordrecht (2002)
5. Wang, S.F., Wang, S., Takagi, H.: User Fatigue Reduction by an Absolute Rating Data-trained Predictor in IEC. In: Proc. IEEE Congress on Evolutionary Computation, pp. 2195–2200. IEEE Press, New York (2006)
6. Takagi, H.: Interactive Evolutionary Computation. In: Proc. of the 5th International Conference on Soft Computing and Information / Intelligent Systems, Iizuka, Japan, pp. 41–50 (1998)
7. Birkhoff, G.D.: Aesthetic Measure. Harvard University Press, Cambridge (1933)
8. Rigau, J., Feixas, M., Sbert, M.: Informational Aesthetics Measures. In: Proc. IEEE Computer society, pp. 24–34 (2008)
9. Schmidhuber, J.: Low-complexity art. Leonardo. Journal of the International Society for the Arts, Sciences, and Technology 30(2), 97–103 (1997)
10. Scha, R., Bod, R.: Informatie en Informatiebeleid 11(1), 54–63 (1993), English translation http://iaaa.nl/rs/compestE.html
11. Machado, P., Romero, J., Manaris, B.: Experiments in Computational Aesthetics: An Iterative Approach to Stylistic Change in Evolutionary Art. In: The Art of Artificial Evolution: A Handbook on Evolutionary Art and Music, pp. 381–415. Springer, Heidelberg (2008)
12. Witten, H.I., Frank, E.: Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations. Morgan Kaufmann Publishers, San Francisco (2000)

Comparing Aesthetic Measures for Evolutionary Art

E. den Heijer[1,2] and A.E. Eiben[2]

[1] Objectivation B.V., Amsterdam, The Netherlands
[2] Vrije Universiteit Amsterdam, The Netherlands
eelco@few.vu.nl, gusz@cs.vu.nl

Abstract. In this paper we investigate and compare four aesthetic measures within the context of evolutionary art. We evolve visual art with an unsupervised evolutionary art system using genetic programming and an aesthetic measure as the fitness function. We perform multiple experiments with different aesthetic measures and examine their influence on the evolved images. To this end we store the 5 fittest individuals of each run and hand-pick the best 9 images after finishing the whole series. This way we create a portfolio of evolved art for each aesthetic measure for visual inspection. Additionally, we perform a cross-evaluation by calculating the aesthetic value of images evolved by measure i according to measure j. This way we investigate the flexiblity of each aesthetic measure (i.e., whether the aesthetic measure appreciates different types of images). The results show that aesthetic measures have a rather clear "style" and that these styles can be very different. Furthermore we find that some aesthetic measures show very little flexibility and appreciate only a limited set of images.

1 Introduction

The goal of the research field of Computational Aesthetics is to investigate "computational methods that can make applicable aesthetic decisions in a similar fashion as humans can" [5]. Aesthetic measures are functions that compute the aesthetic value of an object. [2] was the first to publish on the subject of aesthetic measures, and his work has been influential in the field. Birkhoffs notion of aesthetics was based on the relation between Order and Complexity, expressed as $M = \frac{O}{C}$, where O stands for order and C for Complexity. Birkhoffs measure is now widely regarded a being mostly a measure of orderliness. Since Birkhoff, several researchers have investigated aesthetic measures from several points of view. [4] and [5] give good overviews of the field.

1.1 Research Question

In this paper we investigate and compare four aesthetic measures. Each aesthetic measure is used in an evolutionary art system as a fitness function (all evolutionary parameters are kept equal for all aesthetic measures). We evolve small

C. Di Chio et al. (Eds.): EvoApplications 2010, Part II, LNCS 6025, pp. 311–320, 2010.

Lisp like expressions that generate images, and compare the difference between the images created by the four aesthetic measures. Next, we investigate how the produced images using aesthetic measure M_N are judged by the other aesthetic measures. Hereby we obtain an indication of the neutrality of the measure.

The rest of the paper is structured as follows. First we discuss evolutionary art and the use of aesthetic measures within the context of evolutionary art (section 2). Section 3 discusses our software environment Arabitat. Next, we describe the experiments and their results in section 4.1. In section 4.2 we calculate the cross evaluation of the four aesthetic measures. Sections 5 and 6 contain conclusions and directions for future work.

2 Evolutionary Art

Evolutionary art is a research field where methods from Evolutionary Computation are used to create works of art (good overviews of the field are [12] and [1]). Some evolutionary art systems use supervised fitness assignment (e.g. [15], [13]), and in recent years there has been increased activity in investigating unsupervised fitness assignment (e.g. [14]). The field of Computational Aesthetics investigates how computational methods can be used to assign aesthetic judgement to objects (see [5] and [4]). Functions that assign an aesthetic value to an object are typically called aesthetic measures. In this paper we investigate four aesthetic measures, and compare their output.

2.1 Four Aesthetic Measures

The four aestetic measures that we investigate in this paper have different mechanisms and backgrounds, and we will describe them briefly. For a more detailed description we refer to the original papers. We will briefly describe the aesthetic measures by Machado & Cardoso, Ross & Ralph, the Fractal Dimension measure, and the Combined Weighted sum measure.

Machado & Cardoso. The aesthetic measure described in [8] builds on the relation between Image Complexity (IC) and Processing Complexity (PC). Images that are visually complex, but are processed easily have the highest aesthetic value. As an example, the authors refer to fractal images; they are visually complex, but can be described by a simple formula. The aesthetic measure M of an image I is defined as

$$M(I) = \frac{IC(I)}{PC(I)} \tag{1}$$

The Image Complexity can be regarded as the effort needed to compress an image, and is defined as

$$IC(I) = \frac{RMS(I)}{Compressionratio(I)} \tag{2}$$

where RMS refers to the difference between the original image and the compressed image, expressed as the root mean square. The compression ratio is the ratio between the original image size and the compressed image size. The authors suggest the use JPEG compression for image compression. We used a JPEG quality setting of 0.75 (medium quality). The Processing Complexity is calculated using fractal image compression; in our experiments we used images with a resolution of 300x300. The box-counting algorithm used a number of boxes between 6 and 75 and the threshold was set to 50.

Ross & Ralph (bell curve). A second aesthetic measure that we implemented is Ross and Ralph (Ralph's Bell Curve, [14]). This measure is based on the observation that many fine art painting exhibit functions over colour gradients that conform to a normal or bell curve distribution. The authors suggest that works of art should have a reasonable amount of changes in colour, but that the changes in colour should reflect a normal distribution (hence the name 'Bell Curve'). The computation takes several steps and we refer to [14] for details.

Fractal dimension. In [16] the authors investigate the aesthetic preference of people for several types of fractals (natural, artifical and man-made). The authors found a peak in the preference for fractal images with a fractal dimension around 1.35. Images with a higher fractal dimension were considered complex, and images with a lower dimension were considered uninteresting. We use this finding to construct an aesthetic measure. For a given image I with a fractal dimension d, we define our fractal dimension aesthetic measure M as

$$M(I) = max(0, 1 - |1.35 - d(I)|) \tag{3}$$

which means that only images with a fractal dimension between 1.1 and 1.6 have a positive aesthetic measure (where images with a fractal dimension of 1.35 have an aesthetic value of 1). We calculate the fractal dimension using a technique called "box-counting" (see [16]).

Combined Weighted Sum. We also wanted to investigate the usefullness of a combination of the aesthetic value by the aforementioned three aesthetic measures. We used a simple straightforward weighted sum measure were all weights were set to 1:

$$M(I) = \frac{\sum_{i=1}^{n} M_i(I)}{n} \tag{4}$$

3 Arabitat: The Art Habitat

Arabitat (Art Habitat) is our software environment in which we investigate evolutionary art. It uses genetic programming with Lisp expressions and supports both supervised and unsupervised evaluation. In this paper we only discuss unsupervised fitness evaluation using aesthetic measures. Currently we have implemented three aforementioned aesthetic fitness functions and a weighted sum

combination measure, and intend to implement more in the near future. In our system, a genotype consists of 1) a Lisp-style expression that returns a value of type double, and 2) a color lookup table. Lisp-like expressions are common within genetic programming (see [7]). Our genetic programming is type-safe and returns only results of type doubles.

The computation of a phenotype from the genotype is done as follows; for a target phenotype image with a resolution *(width, height)* we calculate the function value from the lisp expression (the genotype) for each *(x,y)* coordinate of the image. The resulting matrix of floating points is mapped onto an indexed colour table, and this results in a matrix of integers, where each integer refers to a colour index of the corresponding colour scheme. This way the colouring is independent of the double values (other approaches like [15] have functions that directly address colouring). The colour scheme is thus part of the genotype, and is also subject to mutation and crossover. A mutation in the colour scheme could result in an entirely different coloured image, even if the expression remain unaltered. The resulting image is passed to the fitness function (one of the aesthetic measures) for evaluation. See Figure 1 for a schematic overview (see http://www.few.vu.nl/~eelco/ for more examples in colour).

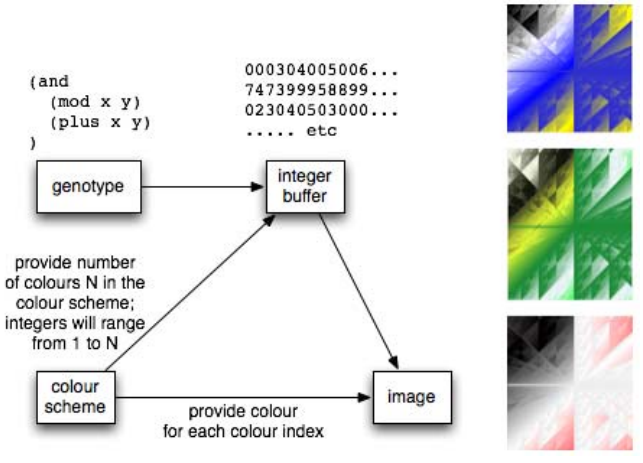


Fig. 1. A schematic overview of the expression of the genotype into the phenotype (image) for LISP expression ((and (mod x y) (plus x y))); the three images on the right are three renderings of the same expression, using three different colour schemes

Function set. Many functions used are similar to the ones used in [15], [13] and [14]. Table 1 summarizes the used functions (including their required number of arguments); The terminals x and y are variables that refer to the (x, y) coordinate of a pixel. 'Width' and 'height' are variables that refer to the width and height of the image. The use of width and height is useful because we usually perform evolutionary computation using images with low resolution

Table 1. Function and terminal set of our evolutionary art system

Terminals	x,y, ephem_double, ephem_int, width, height, golden_ratio, pi
Basic math	plus/2, minus/2, multiply/2, mod/2, div/2, average/2
Other math	sin/1 ,cos/1, tan/1, sinh/1, cosh/1, tanh/1, atan2/2, cuberoot/1, squareroot/1, hypot/2
Relational	minimum/1, maximum/1, if-then-else/3
Bitwise	and/2, or/2, not/1, xor/2
Noise	perlinnoise/2, smoothnoise/2, marble/2, turbulence/2, plasma/2
Fractal	mandelbrot/2, julia/2
Boolean	equals/2, lessthan/2, greaterthan/2

(say 300x300) and want to display the end result on a higher resolution. [15], [13] and [14] contain details on the functions used in our function set.

4 Experiments

In order to investigate and compare the four different aesthetic measure we conducted a number of experiments. We performed 10 runs for each aesthetic measure and collected the images of the 5 fittest individuals of each run. Next, we calculated the aesthetic measure of those 5 individuals by the other aesthetic measures. From the 50 images of each experiment (10 runs, 5 fittest individuals) we handpicked 9 images that were typical for that image set. Besides the aesthetic measure, all evolutionary parameters were the same for each run. We did many preliminary experiments and found that populations of around 200 usually tended to converge to one or two dominant individuals and their similar offspring. Since the goal of this paper is to compare the output of evolutionary art using different aesthetic measures, we decided to perform evolutionary search for 10 generations with a population of 200. For the genetic operators we used subtree mutation (rate 0.05), subtree crossover (rate 0.85), we initialized the population using the well-known ramped half-and-half initalization method (see [7]), and used tournament selection (tournament size 3) for both parent selection and survivor selection. For survivor selection we use elitist selection (best 1).

4.1 Results

We did 10 runs with our evolutionary art system using each aesthetic measure and collected the images of the 5 fittest individuals of each run. The average fitness of the population of 200 over 10 generations is given in Figures 3 and 5. Of the collected 50 images, we hand-picked 9 images. The reason for hand-picking from the image collection instead of selecting the images with the highest fitness is that some runs ran into premature convergence and had 5 very similar images at the end. Therefore we picked the images by hand, to give an impression of the variety of the images. Since we were mostly interested in comparing aesthetic

measures using the same EA parameters, we did not focus on optimizing the EA to reach an average fitness of 1.0. Our goal was exploration, not optimization. Therefore, many runs do not end in an average fitness of 1.0. In the next sections we shortly describe the characteristics of these selections.

Machado & Cardoso. The images produced using the Machado & Cardoso measure are presented in Figure 2. The images tend to be simple in structure, and they have a slight preference for primary colours (although not in all images). We suspect that the use of JPEG compression could possibly favour images with primary colours. Also apparant is that the images are diverse in structure, even if they are relatively simple. Most images produced using this aesthetic measure have a 'sixties'/ pop art look and feel. The images in [9] are slightly different; we suspect that is caused by using a different function set and a different colouring.

Ross & Ralph (bell curve). The images produced using the aesthetic measure of Ross & Ralph are presented in Figure 2. It is immediately apparant that these images are very different from the ones produced using the Machado & Cardoso aesthetic measure. Most images are very abstract and have a very distinct colour progression within the images. Many images resemble textures that are used in computer graphics, and that is similar to what the original authors found in their evolutionary art system (see [14]).

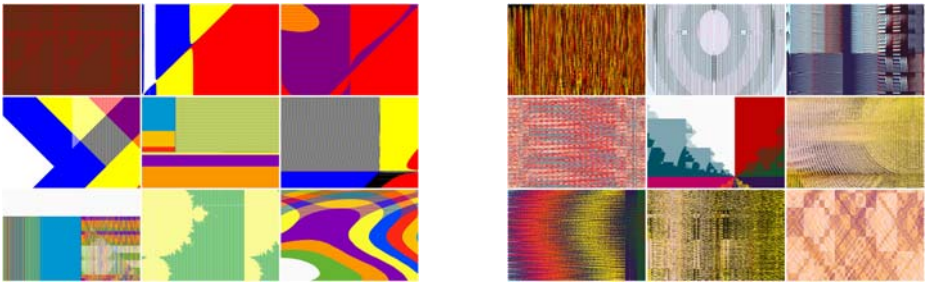


Fig. 2. Summary of images evolved using the aesthetic measure of Machado & Cardoso (left) and Ralph & Ross (right)

Fractal dimension. The image produced using our fractal dimension aesthetic measure are presented in Figure 4.

What is apparant from these images is that the style is again different from the previous two aesthetic measures. Next, we see that there is a tendency to use the fractal functions *mandelbrot* and *julia* (which generates Julia set figures) and the binary function *xor* and *and*. As far as we know we are the first to use the fractal dimension in an evolutionary art context, so we can not compare our images with other generated images.

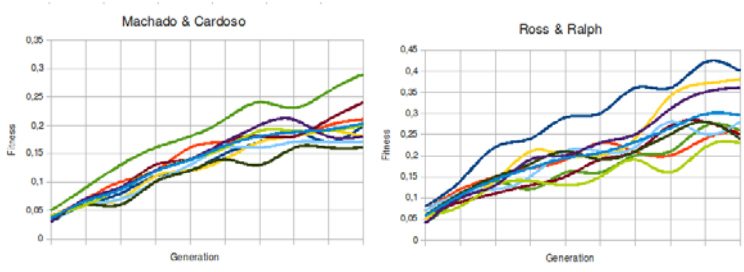


Fig. 3. Fitness progression of 10 different runs using the aesthetic measure by Machado & Cardoso (left) and Ralph & Ross (right); both ran 10 generations

Combined Weighted Sum. The images produced using the combined weighted sum aesthetic measure is given in Figure 4.

Ideally these images would be a combination of features of the previous images of the other aesthetic measures. We see that features of the aesthetic measure of Ralph & Ross seems to be dominant in the images, and that can be explained by the fact that the the average fitness of the runs using the Ross & Ralph measure

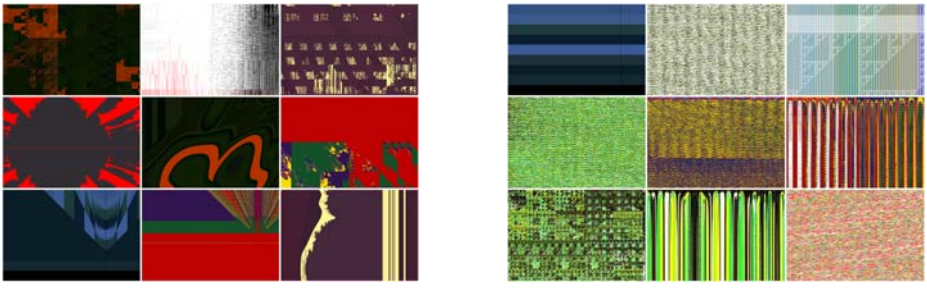


Fig. 4. Summary of images evolved using the Fractal Dimension aesthetic measure (left) and Combined weighted sum (right)

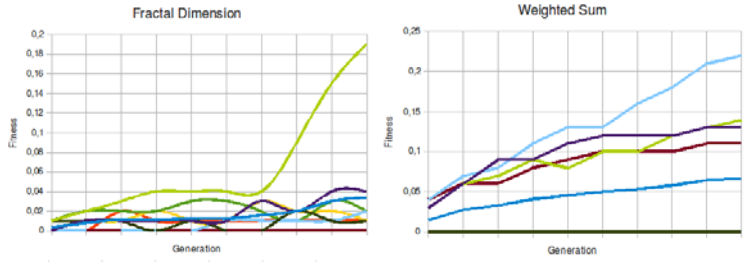


Fig. 5. Fitness progression of 10 different runs using the fractal dimension aesthetic measure (left) and the combined weighted sum measure (right); both ran 10 generations

is around 0.5, and the other two measures have an average fitness of around 0.2. Using weighted sum combination, the Ralph & Ross aesthetic measure thus has more 'weight' than the other two. In future implementations of combinations of aesthetic measures, we will use techniques from [3] to combine aesthetic measures in a more neutral fashion.

4.2 Cross Evaluation

After we had done the experiments with the four aesthetic measures, we wanted to know how the aesthetic measures would evaluate 'each others' work. The evaluation of the work of measure M_n of images produced using aesthetic measure M_m might give us an indication of the scope of the aesthetic measure. If an aesthetic measure only appreciates images that were generated using its own measure, then we could assume that its scope were fairly limited. On the other hand, if a measure also appreciates images that were created using another aesthetic measure, we could conclude that it is applicable to a broader scope of images. In the following table we have gathered the average fitness (and standard deviation) of the fifty fittest individuals that were collected for each experiment. The producing aesthetic measure is presented horizontally and the evaluation by all aesthetic measures is presented in the columns. From this table we can conclude a number of findings. First, all aesthetic measures like their own work best (except for the combined weighted sum measure). Next, we can clearly see that the fractal dimension aesthetic measure does not appreciate of images produced by other aesthetic measures; the average score is 0.0, which means that all images not produced using the fractal dimension aesthetic measure have a fractal dimension outside the range [1.1,..,1.6]. This basically means that the fractal dimension aesthetic measure is not widely applicable as a aesthetic measure; many people like fractal properties in images, but in reality, not many images actually have fractal properties (i.e. a fractal dimension within the range [1.1,..,1.6]). Next, we see that the Ralph & Ross aesthetic measure appreciates of its own work (which is not surprising) but also appreciates of the works produced using the Machado & Cardoso aesthetic measure.

Table 2. The cross evaluation of the aesthetic value of each others images. We present the average asethetic value and the standard deviation in parentheses.

		Evaluated by			
		Machado& Cardoso	Ross & Ralph	Fractal Dim.	Combined Weighted Sum
	Mach.& Card.	0.096 (0.054)	0.246 (0.363)	0 (0)	0.114 (0.124)
Produced	Ross & Ralph	0.035 (0.023)	0.562 (0.476)	0 (0)	0.199 (0.161)
By	Fract. Dim.	0.03 (0.009)	0.061 (0.194)	0.136 (0.305)	0.076 (0.115)
	Comb. Wei. Sum.	0.049 (0.031)	0.194 (0.337)	0 (0)	0.081 (0.115)

5 Conclusions

In this paper we have investigated and compared four aesthetic measures in an evolutionary art system. After our experiments we can conclude that the use of different aesthetic measures clearly results in different 'styles' of evolutionary art. Since all evolutionary parameters were kept equal in all experiments, we can conclude that all differences in artistic style are directly related to the aesthetic measures. Next, we can conclude that there are also differences in variety of the output of the four aesthetic measures. The measures of Machado & Cardoso and of Ross & Ralph have varied output. The fractal dimension aesthetic measure produces less varied output and seems less suitable as a universal aesthetic measure. Next we investigated how well the aesthetic measures like each others work. We found that the aesthetic measures by Machado & Cardoso and by Ross & Ralph appreciated work by others. The fractal dimension aesthetic measure however, did not appreciate the output by the other measures, and seems less suitable as a universal aesthetic measure. We think that the fractal dimension aesthetic measure can be useful in cooperation with other aesthetic measures in a multi-objective optimization setup. The output of the combination weighted sum measure resembles the output of the measure by Ralph & Ross, mainly because the average fitness of the Ralph & Ross measure was higher than the average fitness of the other two. In future implementations we could normalize the fitness values per aesthetic measure, in order to avoid unnecessary bias due to differences in maximum fitness. Finally, it is interesting to note that aesthetic measures used to have a passive role in computing the aesthetic value of an object, but seem to have a far more active role in creating art when applied in an evolutionary art system.

6 Future Work

In this paper we chose three aesthetic measures as input for experiments with evolutionary art. Machado & Cardoso continued to develop their aesthetic measure in later research; we intend to include these changes and improvements in our implementation. Furthermore, there exist more aesthetic measures in literature. We will implement the Pattern Measure of [6], and an aesthetic measure based on information theory described in [11]. Furthermore, we would like to further explore the combination of multiple aesthetic measures into a combined aesthetic measure using techniques from multi-objective optimization (see [3]). In our experiments we have hand-picked the output from the fittest individuals; in future research we would like to investigate the use of techniques from digital image processing to extract features from images. This way, it might be possible to investigate the output per aesthetic measure in a more systematic way.

References

1. Bentley, P.J., Corne, D.W. (eds.): Creative Evolutionary Systems. Morgan Kaufmann, San Mateo (2001)
2. Birkhoff, G.D.: Aesthetic Measure. Harvard University Press, Cambridge (1933)
3. Deb, K.: Multi-objective Optimization using Evolutionary Algorithms. Wiley, Chichester (2001)
4. Greenfield, G.: On the origins of the term "computational aesthetics". In: Neumann, et al. (eds.) [10], pp. 9–12
5. Hoenig, F.: Defining computational aesthetics. In: Neumann, et al. (eds.) [10], pp. 13–18
6. Klinger, A., Salingaros, N.A.: A pattern measure. Environment and Planning B: Planning and Design 27, 537–547 (2000)
7. Koza, J.R.: Genetic Programming: On the Programming of Computers by Means of Natural Selection. MIT Press, Cambridge (1992)
8. Machado, P., Cardoso, A.: Computing aesthetics. In: de Oliveira, F.M. (ed.) SBIA 1998. LNCS (LNAI), vol. 1515, pp. 219–228. Springer, Heidelberg (1998)
9. Machado, P., Cardoso, A.: All the truth about nevar. Applied Intelligence 16(2), 101–118 (2002)
10. Neumann, L., Sbert, M., Gooch, B., Purgathofer, W. (eds.): Computational Aesthetics 2005: Eurographics Workshop on Computational Aesthetics in Graphics, Visualization and Imaging 2005, Girona, Spain, May 18-20. Eurographics Association (2005)
11. Rigau, J., Feixas, M., Sbert, M.: Informational aesthetics measures. IEEE Computer Graphics and Applications 28(2), 24–34 (2008)
12. Romero, J., Machado, P. (eds.): The Art of Artificial Evolution: A Handbook on Evolutionary Art and Music. Natural Computing Series. Springer, Heidelberg (2007)
13. Rooke, S.: Eons of genetically evolved algorithmic images. In: Bentley and Corne [1], pp. 339–365
14. Ross, B., Ralph, W., Zong, H.: Evolutionary image synthesis using a model of aesthetics. In: IEEE Congress on Evolutionary Computation, CEC 2006, pp. 1087–1094 (2006)
15. Sims, K.: Artificial evolution for computer graphics. In: SIGGRAPH 1991: Proceedings of the 18th annual conference on Computer graphics and interactive techniques, vol. 25, pp. 319–328. ACM Press, New York (1991)
16. Spehar, B., Clifford, C.W.G., Newell, B.R., Taylor, R.P.: Universal aesthetic of fractals. Computers & Graphics 27(5), 813–820 (2003)

The Problem with Evolutionary Art Is …

Philip Galanter

Department of Visualization, Texas A&M University, College Station, Texas, USA
galanter@viz.tamu.edu

Abstract. Computational evolutionary art has been an active practice for at least 20 years. Given the remarkable advances in that time in other realms of computing, including other forms of evolutionary computing, for many a vague feeling of disappointment surrounds evolutionary art. Aesthetic improvement in evolutionary art has been slow, and typically achieved in ways that are not widely generalizable or extensible. So what is the problem with evolutionary art? And, frankly, why isn't it better? In this paper I respond to these questions from my point of view as a practicing artist applying both a technical and art theoretical understanding of evolutionary art. First the lack of robust fitness functions is considered with particular attention to the problem of computational aesthetic evaluation. Next the issue of genetic representation is discussed in the context of complexity and emergence. And finally, and perhaps most importantly, the need for art theory around evolutionary and generative art is discussed, and a theory that stands typical evolutionary art on its head is proposed.

1 Introduction

In this paper I will discuss several problems around evolutionary art (EA). Since EA has been an active practice for at least 20 years [1], some of this will be review. It is my hope that I can offer some new light on these matters as a practicing artist developing an art-theoretical understanding of evolutionary art.

First the lack of robust automatic aesthetic fitness functions for EA is reviewed, and possible new venues for research are presented. Next genetic representation in EA systems is reviewed. The focus is not so much how genes are used as a medium for evolution, but instead how current gene representation and expression limits emergence and innovation. And finally art theory for EA is considered, and a specific theory rooted in what I've called "truth to process" is offered.

2 The Problem of Fitness Functions for Evolutionary Art

Genetic algorithms and other evolutionary computing techniques are methods to search large multidimensional solution spaces for optimal results. Applications include automotive and aeronautic design, electronic circuit design, routing optimization, modeling markets for investment, and more.

What each of these applications has in common is that an a priori fitness function can be defined, and individuals can be automatically scored as to fitness. [2] In the case of an investment model the score might simply be an amount representing

C. Di Chio et al. (Eds.): EvoApplications 2010, Part II, LNCS 6025, pp. 321–330, 2010.

estimated profit. For circuit design a weighted formula might be used to combine scores for functionality, cost and number of components, power efficiency, and so on. Fully automated evolutionary systems can run with very large populations for hundreds or thousands of generations. Evolutionary art systems are at a significant disadvantage because it is not at all clear how aesthetic judgment can be automated for use as a fitness function. The alternatives for dealing with this problem follow.

2.1 Interactive Evolutionary Computing

One alternative to an automated fitness function for EA is to use aesthetic judgments made by people. From the first historical examples [1] to present interactive evolutionary computing (IEC) dominates the evolutionary art field. In a recent wide-ranging overview of evolutionary visual art Lewis has cataloged a large number of projects with nearly 200 citations. [3] The vast majority of the systems noted are interactive using some form of case-by-case human judgment.

The most obvious problem with having an artist or other judge "in the loop" is that it becomes the rate-limiting step of the iterative process. This is sometimes called "the fitness bottleneck." [4] EA systems can produce new populations orders of magnitude faster than a human can score them. The practical result is that IEC systems typically suffer from relatively small populations and few generations.

Human judgment is also limited by fatigue. Over time user choices will become less consistent, and skew towards novelty for its own sake rather than quality. [5, 6]

One strategy to finesse the fitness bottleneck and fatigue problems is to "crowdsource" the evaluation task. In Sims's Galapagos piece the length of time a visitor spends looking at a particular display is used as a fitness function. In Drave's Electric Sheep system the generated art is displayed on thousands of personal computers as a screen saver and the users of those systems can provide feedback as to their preferences. [7] But as satirically demonstrated by artists Komar and Melamid, making aesthetic choices by polling the public does not produce the unique kind of vision expected from artists. The resulting art trends towards a mediocre mean. [8]

2.2 Computational Aesthetic Evaluation

The Mechanical Turk was a device created in the 18th century that appeared to be a machine that could play chess. [9] The Mechanical Turk was, of course, more a feat of stage magic than computation. Despite the fact that it appeared otherwise, a human operator was hidden inside the cabinet. And it was this operator who made all the playing decisions and won or lost the game.

From a certain point of view using IEC to create art is a similar trick. Perhaps the most important and difficult component in traditional art creation is the exercise of aesthetic judgment while the artifact is being made.[1] And hidden in the IEC system is a human operator playing the game and making those critical decisions.

Autonomous EC systems capable of producing art would be much more satisfying. And using computational aesthetic evaluation (CAE) to provide automated fitness

[1] Art in the 20th century took a decidedly conceptual turn. Since then aesthetics as the pursuit of physical beauty is often not the first priority in art making.

scores would allow populations and generations akin to other applications, presumably resulting in better evolutionary art.

However, CAE is a distinctly non-trivial unsolved problem. Not that there haven't been partial attempts. For example, Sanders and Gero [10] and Greenfield [11] have researched agent-based evaluation systems. Jaskowski et al [12], Fornari et al [13], and Ciesielski et al [14] have reported on systems with automated scoring based on some form of error measurement with reference to an exemplar.

McDermott et al [15] used a combination of perceptual measures, spectral analysis, and low-level sample-by-sample comparison to develop synthesizer voices relative to exemplar sound targets. Khalifa and Foster [16] devised a two stage music composition system that analyzes note intervals and ratios for use in a fitness function.

Various numeric measures as aesthetic indicators have been explored such as Zipf's law (Manaris et al [17]), fractal dimension (Mori et al [18] and Taylor [19]), and various complexity measures (Birkhoff [20], Machado and Cardoso [21]).

Attempts to use connectionist models such as neural networks in computational aesthetic evaluation include Machado et al [22], Phon-Amnuaisuk et al [23], and Gedeon [24].

2.3 Hybrid Aesthetic Evaluation

Some researchers trying to deal with the fatigue problems associated with IEC have attempted to create hybrid aesthetic evaluation methods. In such systems a subset of the population is scored by human evaluation, and then those scores are somehow leveraged across the entire population. Takagi offers a broad overview of over 250 cited attempts to fuse EC and IEC systems in creating art systems as well as other application areas. [5] More recent reports on hybrid aesthetic evaluation include Yuan and Gong [6] , Machado et al [25], and Machwe and Parmee [26].

2.4 The Future of Aesthetic Evaluation for Evolutionary Art

The bad news is that one needn't get very far into the above literature to see that the fitness bottleneck restricting evolutionary art has not yet been conquered. IEC still well outperforms EC in terms of artistic results. And those EC and hybrid systems that have had limited success use methods that are generally idiosyncratic and quite specific to a given medium, style, and artistic goal.

This shouldn't be terribly surprising because we don't know much about how human aesthetic evaluation works either. And worse, CAE tends to ignore the culturally determined aspects that fluctuate in time both over the short and long term.

But there is some hope on the horizon. Journals such as Psychology of Aesthetics, Creativity, and the Arts, Empirical Studies of the Arts, and the Journal of Consciousness Studies should be of interest to those researching computational aesthetic evaluation. Experimental psychology is assembling, one detailed study at a time, a scientific picture of how human aesthetics works.

Joining these recent but more traditional efforts is the nascent field of neuroaesthetics. Neuroaesthetics involves the scientific study of the neurological bases for the creation, experience, and contemplation of works of art. A good place to start might be the chapter by Martindale where he has outlined a basic neural network model, and

then cites 25 empirical studies of specific and varied types of aesthetic experience compatible with, and suggestive of, this model. [27] It's important to note that Martindale has not implemented a computational neural network to exploit his model, but this kind of research may well give others an incentive for doing so.

And indeed this new science, and especially analysis at the level of neurology, is beginning to impact the way those in computer science think about connectionist computing. For example Hawkins has introduced a new design he calls "hierarchical temporal memory" (HTM) which is based on a theory of the neocortex. [28]

Finally, some in evolutionary psychology have speculated that our general aesthetic capabilities have been mostly driven by adaptations for mate selection. And animals with far simpler neurological systems than ours also seem to select mates based on a form of aesthetics. In some animals this capability is generalized. [29] With further advances in psychology, neuroaesthetics, and connectionist computing, CAE may not be as far away as it sometimes seems.

3 The Problem of Genetic Representation and Innovation

Many observers have noted that EA systems tend to produce works that have a certain cast or sameness about them. Some pieces will be better than others, and of course the evolutionary process can capture and reapply incremental improvements. But there seems to be an inevitable plateau beyond which the work does not improve, and most importantly, does not exhibit innovation.

In modern western culture the most highly valued artists are those who exhibit innovation. This rarely happens in a single giant leap of course. For example, if you study the paintings of Jackson Pollock in historical sequence you will first note middling cubist-inspired figurative work. Over time the figuration becomes more and more abstract. And finally the figures disappear entirely in the explosion of lines, drips, and splashes Pollock is so well known for. [30]

While one can somewhat imagine an EA system designed for figurative work "loosened up" to create abstract work, consider the counter example of Philip Guston. He famously, and very quickly, went from well-known abstract expressionist work to (deceptively) simple cartoon-like figurative work. [31] It's hard to imagine a current evolutionary system designed for abstraction suddenly producing the human form. The problems of sameness and lack of innovation are real and worth close study.

3.1 Complexification in Nature and Genetic Representation

Evolutionary computation may be inspired by natural evolution, but it is far less complex than then real thing. Perhaps artificial evolutionary systems lack the kind of innovation found in natural evolution due to this lack of complexity. The notion of complexity here is different than both complexity in information theory [32] and the notion of algorithmic complexity. [33-35] Those in complexity science tend to embrace notions similar to what Murray Gell-Mann has called "effective complexity." [37] In this view simple systems are either highly ordered or highly disordered, and complex systems exhibit a dynamic tension between order and disorder. This tension allows complex systems to exhibit emergence across multiple scales. (See figure 1).

To measure effective complexity, at least in principle, Gell-Mann proposes to split a given system into two algorithmic terms. The first algorithm captures structure and the second algorithm captures random deviation or noise. Effective complexity is proportional to the size of the optimally compressed program for the first algorithm that captures structure. Gell-Mann points out that this process is exactly what a complex adaptive system, such as an animal, does as it learns (models) its environment. Aspects that are random, or noise, are forgotten and aspects that exhibit structure are compressed (abstracted and generalized). Structural aspects that resist compression are experienced as being complex.

When a system moves from simplicity to complexity this is sometimes called complexification. [36] In nature gene related complexification happens in at least two ways. Complexification over a long time scale begins with simple single celled organisms, and evolves complex creatures such as humans. But complexification can also refer to developmental biology; the cascade of construction as DNA assembles proteins, proteins form organelles and then cells, cells organize into tissues and organs, and so on. Genes thus have two roles, both as machines that allow long-term evolution and as machines that initiate short-term construction exhibiting multiple levels of emergence and increasing scale.

The suggestion offered here is that EC with a single level of emergence from genotype to phenotype is not capable of the complexification that art requires.[2] Most evolutionary artists concentrate on the first function of genes (evolution) and do very little with the second (construction through multiple levels of emergence). Without sufficient complexification capacity EA systems cannot exhibit innovation in the sense seen in Pollock and Guston. They remain trapped in a phase space of overly similar aesthetics.

In the design of any evolutionary system the genetic representation has meta-significance in that it may constrain the space of not only all possible evolutionary paths, but also all possible developmental paths. Four types of genetic representation follow in complexification capacity order.

The simplest genetic representation is ***fixed parametric*** representation. Imagine a system for creating drawings of insects. There might be a gene for head size, another for body color, another for leg length, and so on. While such a system may draw a wide variety of insects it will never draw a spider because unless there is a "number of legs" gene all results will have six legs. The complexification capacity of this system is highly constrained.

Slightly more complicated is an ***extensible parametric*** representation. Such a system might have one gene per leg, and thus the ability to draw insects, spiders, and even centipedes and millipedes. But it will not be capable of drawing fish or birds because it lacks fin and wing genes. The complexification capacity of this system is still fairly constrained.

More complicated yet is a ***direct mechanical*** representation. In our example this genetic system doesn't describe the end result, but rather describes machines that can draw. Such a representation will, in theory, allow most anything to be drawn. In addition, during reproduction the genes themselves may mutate making the child different

[2] The developmental aspect of gene expression has been an object of previous discussion. See for example Bentley and Corne's discussion of embryogeny [37].

than the parent. For example, a machine that creates thin pencil lines may mutate into a machine that makes brushed ink marks. Such a system may seem to be of unlimited potential, i.e. unlimited complexification capacity. But such a system is only capable of a single layer of emergence. The machines immediately and directly draw the picture, and that is that.

The final genetic representation is a ***reproductive mechanical*** representation. Such a system is similar to the previous one, with the addition that within a single individual a machine may also create another machine, reproduce itself, or contribute to an emergent machine at a higher level of complexity and scale. This is, in fact, the kind of genetic representation found in nature. There is an upwardly layered increase of complexity as DNA creates proteins, proteins organize to create organelles, organelles organize to create cells, cells organize to create organs, and so on.

Reproductive mechanical genetic representation maximizes complexification capacity because it can initiate multiple layers of emergence across multiple scales. EA has focused on evolution while mostly ignoring this second aspect of genes. Reproductive mechanical genes may help solve the sameness and innovation problem.

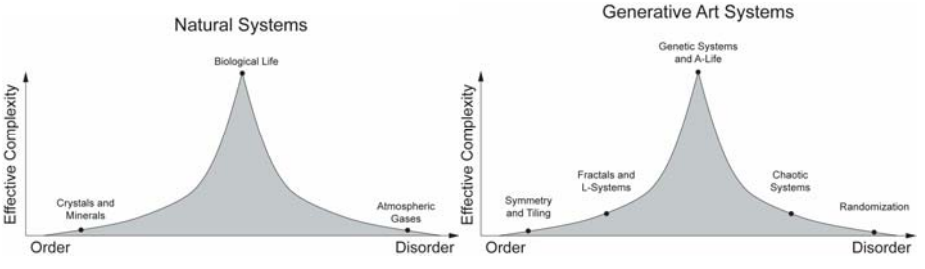


Fig. 1. On the left, effective Complexity in Natural Systems. On the right, effective Complexity in Generative Art Systems.

4 The Problem of Art Theory for Evolutionary Art

Art is more than a series of sensory experiences[3]. It is also a stream of ideas that bind art production, art criticism, and art meaning to the larger culture. A natural place to start is with the question "what is art?" Most agree that to define art is to propose a theory of art. [38] And as a corollary, art without theory loses its definition, its identity, and its meaning. As a practical matter evolutionary art theory will be required before EA will be able to gain entry to the wider art world and inclusion in the general cannon of art.

As part of a broad future-oriented overview McCormack recently offered a number of grand challenges with regard to evolutionary art. [39] The last and least discussed of his grand challenges is the development of art theory for evolutionary and generative art. McCormack *does* refer to a new sense of aesthetics related to artificial life and evolutionary art as reflected in theorist Mitchell Whitelaw's notion of

[3] Please note that at this point that we are shifting gears from primarily scientific or technical considerations to a discussion of art theory. And art theory has its own traditions, rules of evidence, and notions of acceptable rhetoric.

metacreation. [40] Metacreation refers to the role of the artist shifting from the creation of artifacts to the creation of processes that in turn create artifacts. But, in fact, what Whitelaw has called metacreation is not new, nor is it intrinsically digital.

In previous writings I've noted that the common element of all generative art is the ceding of control by the artist to an autonomous system. [41] With the inclusion of systems such as symmetry, pattern, and tiling one can view generative art (or Whitelaw's metacreation) as being as old as art itself. This view of generative art also includes 20th century chance procedures as used by Cage, Burroughs, Ellsworth, Duchamp, and others. What is new today isn't generative art per se, but rather the use of complex systems such as artificial life or evolution rather than the previously used simple systems.

Given that this generative art theory turns on the use of systems by the artist it should not be surprising that Gell-Mann's notion of effective complexity in systems can be used to classify various kinds of generative art. When generative art systems are viewed in this way it suggests a robust theory of generative art. (See figure 1).

This view of generative art casts a very wide net that is independent of any particular past or future technology. It correctly identifies the use of systems, rather than computers, as being the defining aspect of generative art. And by including artists and generative work already well accepted in the art world other forms of generative art are pulled into to the standard art canon. Generative subgenres such as evolutionary art should no longer be left isolated as awkward art world orphans.

While further discussion is beyond the scope of this paper, it's worth noting that in other writing I've used this framework to provide a bridge to a more general critique of the modern/postmodern dialectic I've called "complexism." [42, 43] From that view evolutionary art not only rehabilitates formalism as being significant and meaningful, it also reintroduces dynamism and the aesthetics of process and motion.

4.1 Evolutionary Art Theory and Truth to Process

At various points in art history the notion of "truth to materials" has ascended. In modern architecture this meant that concrete was presented as concrete, and steel beams were presented as steel beams. Clement Greenberg took a similar tack in his critique of abstract expressionism. [44] This involved the rejection of illusory space and representation where the canvas acts as a simulated window. Instead the canvas was simply considered as a flat surface supporting non-representational paint. The underlying idea is that aesthetic power comes from the honest presentation of the essential nature of the medium being used in its purest form.

Given the relative lack of art theory for evolutionary art this essentialist approach may be a good first approximation. So what is the one thing that all evolutionary art shares, and without which it would cease to be evolutionary art? In this case what is essential is not a material property at all, but rather the evolutionary process itself. And so by extension evolutionary art aesthetics should focus on the process rather than the material results. And so what can be said about this process?

It's already been noted that evolution in nature depends on multiple layers of emergence at ever increasing scales. *Evolution is by nature a bottom up process.*

And although the factual details are sketchy, the levels of emergence that created DNA, then proteins, then organelles and cells, happened long before a trace of more

complex multicellular organisms appeared. DNA did not form so that someday man could appear. Rather DNA likely emerged from existing autocatalytic complexes and then sustained itself through reproduction. *Evolution is not teleological.* [45]

In addition evolution does not produce complex creatures by a direct mapping of genotype to phenotype. *Evolution depends on multiple levels of complex emergence.*

Compare this to how most evolutionary art is created. Most EA systems can't innovate via multiple levels of emergence. They generate results with a troubling sameness. To compensate EA systems end up being designed from the top down. The gene pool may start in a random state, but the dice are already loaded because the genetic representation has been designed to lead to the general kind of result desired.

In typical evolutionary art a single level of emergence limits complexity. The bottom up nature of evolution is turned top down. And the teleology that doesn't exist in natural evolution is introduced in the art that it supposedly inspired.

From an essentialist art theory point of view typical fitness function driven evolutionary art is incoherent due to self-contradiction.

Evolutionary art in the context of fine art and rigorous art theory cannot assert itself while it contradicts itself. The process is what makes evolutionary art unique, powerful, and meaningful. And truth to process in evolutionary art is what will turn the field away from self-contradiction and incoherence[4].

Truth to process in evolutionary art demands a bottom up approach. Gene expression should not directly produce a final work but merely trigger the first of many levels of emergence. The artist's primary aesthetic concern should be about putting on display the process, not the product, of complexification.

Generative art theory in general, including that of evolutionary art in particular, moves from noun-dominated art to new practices where verbs become the content. The artifact may be less important than the process. The artifact may even be entirely irrelevant. Truth to process is where the beauty will be found.

References

1. Todd, S., Latham, W.: Evolutionary art and computers, 224 p., 32 p. of plates, Academic Press, London (1992)
2. Fogel, L.J.: Intelligence through simulated evolution: forty years of evolutionary programming. Wiley series on intelligent systems, vol. Xii, p. 162. Wiley, New York (1999)
3. Lewis, M.: Evolutionary Visual Art and Design, in The art of artificial evolution: a handbook on evolutionary art and music. In: Romero, J., Machado, P. (eds.), pp. 3–37. Springer, Berlin (2008)
4. Werner, G.M., Todd, P.M.: Frankensteinian Methods for Evolutionary Music Composition. In: Griffith, N., Todd, P.M. (eds.) Musical networks: Parallel distributed perception and performance. MIT Press/Bradford Books, Cambridge (1998)
5. Takagi, H.: Interactive evolutionary computation: Fusion of the capabilities of EC optimization and human evaluation. Proceedings of the IEEE 89(9), 1275–1296 (2001)

[4] Systems that eschew a priori fitness function objective optimization, and instead rely on co-evolutionary and artificial life-based methods, point towards a truer process-oriented form of evolutionary art. But these systems still typically lack the gene-initiated mechanics for creating multiple levels of emergence and complexification.

6. Yuan, J.: Large population size IGAs with individuals' fitness not assigned by user. In: Huang, D.-S., Wunsch II, D.C., Levine, D.S., Jo, K.-H. (eds.) ICIC 2008. LNCS (LNAI), vol. 5227, pp. 267–274. Springer, Heidelberg (2008)
7. Draves, S.: The electric sheep screen-saver: A case study in aesthetic evolution. In: Rothlauf, F., Branke, J., Cagnoni, S., Corne, D.W., Drechsler, R., Jin, Y., Machado, P., Marchiori, E., Romero, J., Smith, G.D., Squillero, G. (eds.) EvoWorkshops 2005. LNCS, vol. 3449, pp. 458–467. Springer, Heidelberg (2005)
8. Ross, A.: Poll stars. Artforum International 33(5) (1995)
9. Aldiss, B.: The mechanical Turk - The true story of the chess-playing machine that changed the world. In: Tls-the Times Literary Supplement, vol. (5170), p. 33 (2002)
10. Saunders, R., Gero, J.S.: Curious agents and situated design evaluations. Ai Edam-Artificial Intelligence for Engineering Design Analysis and Manufacturing 18(2), 153–161 (2004)
11. Greenfield, G.: Robot paintings evolved using simulated robots. In: Rothlauf, F., Branke, J., Cagnoni, S., Costa, E., Cotta, C., Drechsler, R., Lutton, E., Machado, P., Moore, J.H., Romero, J., Smith, G.D., Squillero, G., Takagi, H. (eds.) EvoWorkshops 2006. LNCS, vol. 3907, pp. 611–621. Springer, Heidelberg (2006)
12. Jaskowski, W.: Learning and recognition of hand-drawn shapes using generative genetic programming. In: Giacobini, M. (ed.) EvoWorkshops 2007. LNCS (LNAI, LNBI), vol. 4448, pp. 281–290. Springer, Heidelberg (2007)
13. Fornari, J.: Creating soundscapes using evolutionary spatial control. In: Giacobini, M. (ed.) EvoWorkshops 2007. LNCS (LNAI, LNBI), vol. 4448, pp. 517–526. Springer, Heidelberg (2007)
14. Ciesielski, V.: Evolution of animated photomosaics. In: Giacobini, M. (ed.) EvoWorkshops 2007. LNCS (LNAI, LNBI), vol. 4448, pp. 498–507. Springer, Heidelberg (2007)
15. McDermott, J., Griffith, N.J.L., O'Neill, M.: Toward user-directed evolution of sound synthesis parameters. In: Rothlauf, F., Branke, J., Cagnoni, S., Corne, D.W., Drechsler, R., Jin, Y., Machado, P., Marchiori, E., Romero, J., Smith, G.D., Squillero, G. (eds.) EvoWorkshops 2005. LNCS, vol. 3449, pp. 517–526. Springer, Heidelberg (2005)
16. Khalifa, Y., Foster, R.: A two-stage autonomous evolutionary music composer. In: Rothlauf, F., Branke, J., Cagnoni, S., Costa, E., Cotta, C., Drechsler, R., Lutton, E., Machado, P., Moore, J.H., Romero, J., Smith, G.D., Squillero, G., Takagi, H. (eds.) EvoWorkshops 2006. LNCS, vol. 3907, pp. 717–721. Springer, Heidelberg (2006)
17. Manaris, B., et al.: Developing fitness functions for pleasant music: Zipf's law and interactive evolution systems. In: Rothlauf, F., Branke, J., Cagnoni, S., Corne, D.W., Drechsler, R., Jin, Y., Machado, P., Marchiori, E., Romero, J., Smith, G.D., Squillero, G. (eds.) EvoWorkshops 2005. LNCS, vol. 3449, pp. 498–507. Springer, Heidelberg (2005)
18. Mori, T., Endou, Y., Nakayama, A.: Fractal analysis and aesthetic evaluation of geometrically overlapping patterns. Textile Research Journal 66(9), 581–586 (1996)
19. Taylor, R.P., Chaos, F.: Nature: a new look at Jackson Pollock. Fractals Research, Eugene (2006)
20. Birkhoff, G.D.: Aesthetic measure, vol. xii, pp. 2 l., 3-225. Harvard University Press, Cambridge (1933)
21. Machado, P.: Computing aesthetics. In: de Oliveira, F.M. (ed.) SBIA 1998. LNCS (LNAI), vol. 1515, pp. 219–228. Springer, Heidelberg (1998)
22. Machado, P., Romero, J., Manaris, B.: Experiments in Computational Aesthetics. In: Romero, J., Machado, P. (eds.) The art of artificial evolution: a handbook on evolutionary art and music. Springer, Berlin (2008)
23. Phon-Amnuaisuk, S.: Evolving music generation with SOM-fitness genetic programming. In: Giacobini, M. (ed.) EvoWorkshops 2007. LNCS (LNAI, LNBI), vol. 4448, pp. 557–566. Springer, Heidelberg (2007)

24. Gedeon, T.D.: Neural network for modeling esthetic selection. In: Ishikawa, M., Doya, K., Miyamoto, H., Yamakawa, T. (eds.) ICONIP 2007, Part II. LNCS (LNAI, LNBI), vol. 4985, pp. 666–674. Springer, Heidelberg (2008)
25. Machado, P., et al.: Partially Interactive Evolutionary Artists. New Generation Computing 23(2), 143–155 (2005)
26. Machwe, A.T.: Towards an interactive, generative design system: Integrating a 'build and evolve' approach with machine learning for complex freeform design. In: Giacobini, M. (ed.) EvoWorkshops 2007. LNCS (LNAI, LNBI), vol. 4448, pp. 449–458. Springer, Heidelberg (2007)
27. Martindale, C.: A Neural-Network Theory of Beauty. In: Martindale, C., Locher, P., Petrov, V.M. (eds.) Evolutionary and neurocognitive approaches to aesthetics, creativity, and the arts, pp. 181–194. Baywood Pub., Amityville (2007)
28. Hawkins, J., Blakeslee, S.: On intelligence, 1st edn., 261 p. Times Books, New York (2004)
29. Watanabe, S.: Pigeons can discriminate "good" and "bad" paintings by children. Animal Cognition (2009)
30. Varnedoe, K., et al.: Jackson Pollock. In: Abrams, H.N. (Distributed by), 336 p. Museum of Modern Art, New York (1998)
31. Read, H.E., Stangos, N.: The Thames and Hudson dictionary of art and artists. In: Rev., expanded and updated ed. World of art, 384 p. Thames and Hudson, New York (1994)
32. Shannon, C.E.: A mathematical theory of communication. The Bell System Technical Journal 27(3), 379–423 (1948)
33. Kolmogorov, A.N.: Three approaches to the quantitative definition of information. Problems in Information Transmission 1, 1–7 (1965)
34. Solomonoff, R.J.: A formal theory of inductive inference, Part I and Part II. Information and Control 7, 1–22, 224-254(1964)
35. Chaitin, G.J.: On the length of programs for computing finite binary sequences. Journal of the ACM (13), 547–569 (1966)
36. Casti, J.L.: Complexification: explaining a paradoxical world through the science of surprise, 1st edn., vol. xiv, 320 p. HarperCollins, New York (1994)
37. Bentley, P., Corne, D.: Creative evolutionary systems, vol. xxxi, 576 p., 8 p. of plates. Morgan Kaufmann/Academic Press (2002)
38. Carroll, N.: Theories of art today, vol. vi, 368 p. University of Wisconsin Press, Madison (2000)
39. McCormack, J.: Facing the Future: Evolutionary Possibilities for Human-Machine Creativity. In: Romero, J., Machado, P. (eds.) The art of artificial evolution: a handbook on evolutionary art and music, pp. 417–451. Springer, Berlin (2008)
40. Whitelaw, M.: Metacreation: art and artificial life, vol. x, 281 p. MIT Press, Cambridge (2004)
41. Galanter, P.: What is Generative Art? Complexity theory as a context for art theory. In: International Conference on Generative Art. Generative Design Lab. Milan Polytechnic, Milan (2003)
42. Galanter, P.: What is Complexism? Generative Art and the Cultures of Science and the Humanities. In: International Conference on Generative Art. Generative Design Lab, Milan Polytechnic, Milan (2008)
43. Galanter, P.: Complexism and the role of evolutionary art. In: Romero, J., Machado, P. (eds.) The art of artificial evolution: a handbook on evolutionary art and music, pp. 311–332. Springer, Berlin (2008)
44. Greenberg, C., O'Brian, J.: The collected essays and criticism. University of Chicago Press, Chicago (1986)
45. Kauffman, S.A.: At home in the universe: the search for laws of self-organization and complexity, vol. viii, p. 321. Oxford University Press, New York (1995)

Learning to Dance through Interactive Evolution

Greg A. Dubbin and Kenneth O. Stanley

School of Electrical Engineering and Computer Science
University of Central Florida
Orlando, FL 32816, USA

Abstract. A relatively rare application of artificial intelligence at the nexus of art and music is dance. The impulse shared by all humans to express ourselves through dance represents a unique opportunity to artificially capture human creative expression. In particular, the spontaneity and relative ease of *moving to the music* without any overall plan suggests a natural connection between temporal patterns and motor control. To explore this potential, this paper presents a model called *Dance Evolution*, which allows the user to train virtual humans to dance to MIDI songs *or* raw audio, that is, the dancers can dance to any song heard on the radio, including the latest pop music. The dancers are controlled by artificial neural networks (ANNs) that "hear" MIDI sequences or raw audio processed through a discrete Fourier transform-based technique. ANNs learn to dance in new ways through an interactive evolutionary process driven by the user. The main result is that when motion is expressed as a function of sound the effect is a plausible approximation of the natural human tendency to move to music.

1 Introduction

The ubiquity of dance throughout the cultures of the world [1] hints at its deep connection to human creativity and self-expression. The power of music and dance as a tool for self expression is further demonstrated by the popularity of such music and rhythm-oriented games as *Guitar Hero*[1], *Rock Band*[2], and *Dance Dance Revolution*[3]. Yet although in recent years researchers in artificial intelligence (AI) have begun to focus on creativity in music and art [2,3,4,5], with few exceptions [6], dance is less explored. Nevertheless, dance can potentially provide insight into how the auditory and motor modalities are connected in creative self-expression. Thus its study is relevant to the enterprise of AI.

Unlike Yu [6], who focused on choreographed dance sequences, the model in this paper investigates the more spontaneous self-expression that results from

[1] Guitar Hero (R) is a trademark of Activision Publishing, Inc.

[2] Rock Band, Rock Band 2 and all related titles are trademarks of Harmonix Music Systems, Inc., an MTV Networks company.

[3] (C) 2008 Konami Digital Entertainment, Inc. "Dance Dance Revolution" is a registered trademark of Konami Digital Entertainment Co., Ltd. KONAMI is a registered trademark of KONAMI CORPORATION.

C. Di Chio et al. (Eds.): EvoApplications 2010, Part II, LNCS 6025, pp. 331–340, 2010.

simply listening to entertaining music, such as in a club setting. In a step toward generating spontaneous dance to arbitrary music, this paper presents a model called *Dance Evolution* in which virtual dancers learn to dance to music encoded as either MIDI or raw audio. In effect, dancers can learn to dance to any song that you might hear in a dance club or on the radio. The model in Dance Evolution can take MIDI sequence data or process raw audio through discrete Fourier transforms to extract an approximation of such data. The resulting temporal progression is input into an artificial neural network (ANN), which outputs a sequence of motor commands that control the body of the virtual dancer. Thus the motion of the dancer becomes a *function* of the beats of the song.

Of course, an important question is how the ANN can learn to make the right moves. However, it turns out that it is possible to quickly discover mappings between audio and motor output that produce movements that appear natural, suggesting that one reason dance is so appealing is that its search space is forgiving. Thus the aim in Dance Evolution is not to learn an *optimal* dance but rather to enable the user to effectively explore the space of possible dances. For this purpose, the user drives an interactive evolutionary algorithm built on the NeuroEvolution of Augmenting Topologies (NEAT) approach to evolving ANNs. In effect, the user *breeds* new dancers from ones that were appealing in the past. In fact, because each evolved ANN embodies the *personality* of a dancer, the same dancer can be transferred from one song to another.

The main insight, that dance can be considerd a function of changing audio over time, suggests a direct coupling between motor control and audio processing. By implementing a model based on this principle, the practical result is an interactive application in which an unbounded space of dance behaviors can be explored and assigned to any song.

2 Background

This section reviews foundational technologies to the Dance Evolution approach.

Interactive Evolutionary Computation. IEC is a growing field within machine learning that takes advantage of humans' ability to make sophisticated subjective judgments [7]. Early IEC implementations include Richard Dawkins's [8] Biomorphs, which evolved visual patterns, and the pioneering work of Sims [5,9], who evolved both art and virtual creatures. In IEC the user is presented with a set of (usually visual) alternatives and evaluates their fitness. The evolutionary algorithm generates a new generation of candidates based on this feedback. The process repeats until the user is satisfied. While the risk is that the user may become fatigued before finding a satisfactory candidate [7], the hope in Dance Evolution is that watching dancers is sufficiently motivating in part to mitigate the effect of fatigue.

NeuroEvolution of Augmenting Topologies. Dance Evolution encodes dance policies as ANNs that "hear" input from a music file and output requests for limb

movement. The ANNs in Dance Evolution are trained through the NeuroEvolution of Augmenting Topologies (NEAT) method, which has proven successful in a variety of control and decision making tasks [10,11,12]. Thus NEAT makes a good platform for evolving controllers for dancers.

NEAT begins evolution with a population of small, simple networks and grows the network topology over generations, leading to increasingly sophisticated behavior. For evolving dance, this process means that dances can become more elaborate and intricate over generations. Stanley and Miikkulainen [12] provide complete introductions to NEAT.

Music Processing. If dancers can only dance to processed MIDI files, the user can only enjoy a limited library of songs. To broaden the possibilities, this paper takes the significant step of extending the functionality to the popular MPEG-1 Audio Layer 3 (MP3) music format. This capability allows the user to enjoy a normal library of songs. Thus the problem is to parse raw sound in such a way that the ANN can interpret it. The approach in this paper is inspired by an algorithm described by Scheirer [13] that can determine the beat from raw musical input.

3 Approach

Dance Evolution was implemented in the Panda3D[4] simulator, which was chosen for its rapid prototyping capabilities and ability to simulate articulated bodies. This section details the main approach, starting with inputs.

3.1 ANN Inputs

To allow the ANN to "hear" the music, data from the song is input over time as a vector representing the current pulses as the song unfolds. For MIDI (i.e. *musical instrument digital interface*) files, this vector contains four parameters. The first three are periodic functions of the beat, measure, and length of the song. The last signal in the vector represents the volume of the *low drum* track, which adds song-specific variation. In this way, the ANN is aware of the major temporal building blocks of music and generates dance as a function of that information. Because MIDI files represent this data explicitly, it is easily extracted and input into the ANN. However, the MIDI format only allows a proof of concept of the technique because most popular songs are available only in raw audio. Thus a key focus of this research is to extend the capability to dancing to raw audio.

3.2 Audio Processing

As Scheirer [13] explains, a beat is a repeating pulse with a regular period throughout a piece of music. To discover the beat, these pulses must be identified;

[4] Panda 3D Software Copyright (c) 2000-2005, Disney Enterprises, Inc. All rights reserved.

however they can appear distinctly within different *subbands* of the song. Therefore, to recognize pulses, the song must first be decomposed into its subbands. This task is accomplished efficiently with a Fast Fourier Transform (FFT).

The FFT in Dance Evolution decomposes the amplitude vector formed from 1,024 samples of the song, which combined represent approximately $\frac{1}{40}$ of a second, into multiple subbands that represent the average energy of the song in several frequency-ranges. In particular, Dance Evolution decomposes the song into 15 subbands whose ranges are $\left[\frac{(\frac{N}{15}*i+4)*f_e}{N}, \frac{(\frac{N}{15}*(i+1)+4)*f_e}{N}\right]$ Hz with $i \in [0, 14]$, $N = 1,024$, and f_e set to the sampling rate of the song, which is usually about 44,100 Hz (the constant 4 is the remainder of dividing 1,024 by 15). As noted by Scheirer [13], differing range divisions do not have a significant effect on results. The FFT also calculates the direct current (DC) term, i.e. the non-periodic component of the vector that represents the average energy of the song.

A beat occurs when the energy in a subband increases dramatically with respect to the average energy of the subband. The FFT of the 1,024 samples represents the near-instantaneous energy of each channel. To search for a beat, this energy is averaged with the previous 43 samples, which represents about one second of time for the common sampling rate of 44,100 Hz. This interval was selected to allow real time processing while maintaining sufficient fidelity.

The ratio of the instantaneous energy to the average energy amplifies differences between them such that a high ratio means that the pulse may be a beat. This pulse information can be input directly into an ANN, causing a spike in the input to correspond to a pulse in the music. This procedure both reduces time spent processing the music and preserves non-beat pulses, which may still be important to the dance. The non-beat pulses have a similar affect to the drum track information from the MIDI files. 16 values representing the pulses from each of the 15 subbands as well as the DC component are input into the ANN. The input vector is calculated and fed into the ANN in real time in each frame.[5]

To make pulse data suitable for ANN input, it must be (1) normalized and (2) lengthened. Because the magnitude of a raw audio pulse is unbounded, the sigmoid function $f(x) = \frac{1}{1+e^{-\alpha(x-\beta)}}$ is applied to each input to both limit the pulse values to $[0, 1]$ and further separate the strong pulses from the weak ones, in effect filtering out the weak signals. Based on an analysis of several songs, α and β are set to 1 and 6 respectively to keep the point of inflection above the bulk of the weaker pulses.

The second problem is that the pulses occur over short periods of time, which does not give the ANN enough time to properly react before the input pulse terminates. Dances produced with such short pulses thus appear jerky. To address this problem, the ANN is provided with a "memory" of the last few inputs by adding a fraction of the previous input into the current one such that $I_t = f(x) + \gamma \cdot I_{t-1}$. This final step provides the ANN with smooth decay curves following each pulse. The result is smooth, natural-appearing motion.

[5] The engine updates the visual display each such frame.

Fig. 1. Dance Evolution User Interface. The user can click on dancers to breed them and control the degree to which they change with the mutation slider in the lower-left.

3.3 ANN Outputs

The ANN controls three-dimensional models within an interactive environment. The output nodes of the ANN request the actuation of the joints of the model. The models' joints are chosen from the head, spine, arms, hips, and legs such that each joint has enough mobility to noticeably affect the dances. Each output affects the angle of an axis (heading, pitch, or roll) of one joint, for a total of 34 outputs. Every frame, the outputs of each ANN are queried and the angles of the models are updated accordingly. Activation travels through the the ANN at a rate of one link per frame. It is important to note that because ANNs evolved by NEAT can contain recurrent connections, in principle it is possible to react to more than just the current beat.

To display the dance, the outputs of the ANN must be converted into actuation commands to move the models. Each frame, the next input vector is calculated from the most recently heard music and loaded into the ANN. Each output of the ANN is in the range $[0, 1]$ and scaled linearly into $[-.2, .2]$. This activation is scaled according to the frame rate so that dancers are consistant regardless of frame rate. Finally, the joint is moved along a sine curve between the physiological limits at a rate determined by the activation, creating a rhythmic motion that avoids being stuck at the limits. That is, the limbs continually oscillate between their limits with a frequency determined by the activation, such that higher activation causes faster movement.

In this way, because each input represents the most recent pulses in the song as heard by the user, the ANN is able to react in real time to the music.

3.4 ANN Training

ANNs in Dance Evolution are trained through IEC, which allows the user to direct their evolution intuitively. Left-clicking on one of the five model dancers produces a new generation of mutants of that model's ANN. The model that is clicked keeps its ANN, which ensures the favorite dancer is not lost. Right-clicking creates a set of hybrids by crossing over the clicked model's ANN with each of the other models' ANNs following the NEAT method, and replacing

the old ANNs of each unclicked model with the new offspring. The user can control the mutation power through a slider provided at the bottom of the screen, which gives the user significant control over evolution. The interface is shown in figure 1. The population can be initialized randomly or from ANNs saved from previous sessions. In addition to saving trained dancer ANNs, the user can load any previously saved ANN.

Users can choose to play any song in their library. Interestingly, ANNs can react to any such song, including those for which they were not trained. In this way, the same dance *personality*, which is embodied in an ANN, can be transferred from one song to another.

4 Experiments and Results

This section is divided into two parts: a brief analysis of learning to dance to MIDI and the more ambitious goal of dancing to raw audio. Please note that the text in this section is accompanied with video demonstrations at: `http://eplex.cs.ucf.edu/dance-evolution-videos.html`.

4.1 Dancing to MIDI

Learning to dance to MIDI is significantly easier than learning with raw audio because the beat and measure are provided explicitly by the MIDI format. Showing what is possible under such conditions provides a context for the results of the greater challenge of learning to dance to raw audio.

MIDI dancers were evolved to MIDI dance songs composed by Bjorn Lynne (Shockwave-Sound.com). The main result is that a wide variety of dances evolved that tightly follow the rhythm of the song, demonstrating that the idea that dance can be generated as a function of temporal information in music can work in principle (see "MIDI Sequence" video). While some of the specific bodily motions sometimes appear unrealistic, it is possible to evolve toward more realistic motions. The quality of results with MIDI was sufficient to win the Best Student Video Award at the AAAI video competition [14]. The dancers perform the most dramatic movements and changes to the beat of the song. The next section investigates the performance of the approach when confronted with the greater challenge of raw audio.

4.2 Dancing to Raw Audio

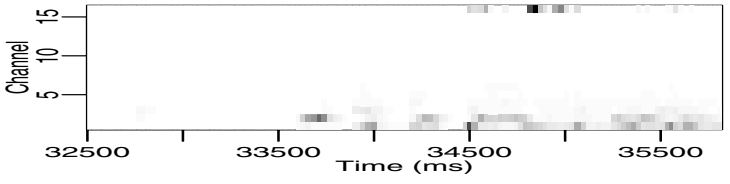
This section analyzes raw audio performance in detail through a variety of specific experiments. The primary aim is to provide significant insight into *how* the results follow from the raw audio processing procedure, from which it will be possible to expand and improve further in the future.

A brief sequence from the song *Tubthumping*[6] by Chumbawamba is chosen to illustrate in detail the responsiveness of the algorithm to pertinent components of raw music. Specifically, the inputs resulting from processing and the

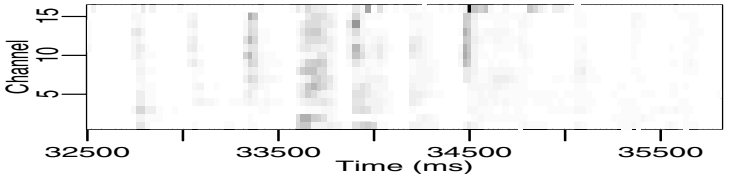
[6] *Tubthumping* is from the album "Tubthumper," released by EMI Group Ltd.

corresponding ANN outputs are graphically analyzed to test whether processing captures important signals from the music and whether the ANN reacts accordingly. *Tubthumping* has a very distinct chorus in which the lyrics state "I get knocked down, but I get up again." These lyrics provide a reference point in the analysis of the network outputs.

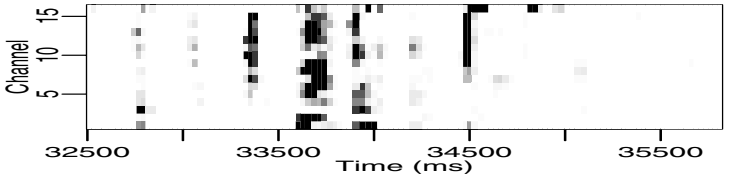
Figure 2 depicts the initial three steps of the raw audio processing of the segment with the lyrics, "I get knocked down, but I get up again," which is completed before inputting data into the ANNs (see "Tubthumper Clip" video for actual footage). Figure 3 demonstrates the movement during this segment as well as the inputs and outputs of the ANN that generated this dance.



(a) First Step: Fast Fourier Transform



(b) Second Step: Pulse Isolation

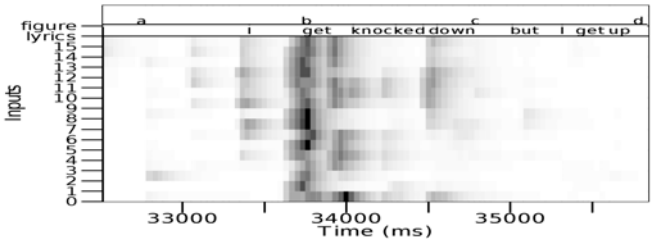


(c) Third Step: Sigmoid Transformation

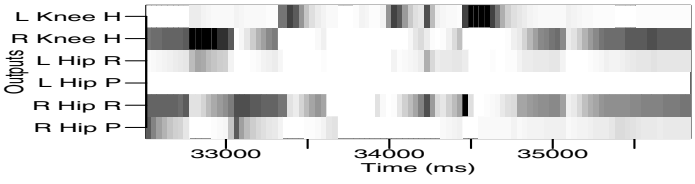
Fig. 2. Processing Raw Audio. The first three steps of processing the song *Tubthumping* are shown. The FFT results are shown for each channel (a). The ratio of the instantaneous to the average FFT isolates pulses in the music (b). The sigmoid function then filters out weak pulses and normalizes the input (c).

The ANN that controls the dancer in figure 3 evolved the humorous behavior of bobbing down when the chorus sings "I get knocked down." Spikes in the processed input can be observed at each of these lyrics. Note that these spikes represent instrumental beats in various frequencies that happen to correspond to the timing of the lyrics, which is how Dance Evolution learns to respond. Other spikes representing a variety of instrumental sounds are also apparent, indicating that the inputs of the ANN indeed receive spikes in the music to which a human

(a) Time = 32727 ms (b) Time = 33726 ms (c) Time = 34764 ms (d) Time = 35726 ms



(e) Input Strength for each Channel Over Time



(f) Upper and Lower Leg Outputs Over Time

Fig. 3. Dancing to Music. An ANN trained with Chumbawamba's *Tubthumping* dances to the chorus of "I get knocked down, but I get up again". The sequence (a–d) are screen shots taken at regular intervals during the dance. Over this time, the ANN sees the inputs processed from this part of the song (e). The outputs for the left and right legs (f) cause the dancer's legs to change direction quickly at the words "I" and "down," corresponding to pulses in the input (e).

would react. Correlated with this input, the output in figure 3f shows that the ANN requests the right and left knees to change direction (i.e. the sign switches) when the lyrics, "I get knocked down" are sung (approximately 33,300 ms into the song). The act of both legs changing direction simultaneously causes the dancer to dip. Thus figure 3 confirms that, with user input, NEAT could find a policy to express the desired behavior as a function of change in subbands.

A single ANN can transfer some of its characteristic behavior between songs, but also exhibits new behavior resulting from different active frequency subbands in different songs. Demonstrating this capability to transfer, a single ANN was trained on George Clinton and the Parliament's *Give Up the Funk (Tear the Roof off the Sucker)*[7] and transferred to to Billy Joel's *Piano Man*[8]. Characteristic

[7] *Give Up the Funk (Tear the Roof off the Sucker)* is from the album "Mothership," released by Casablanca Records.

[8] *Piano Man* is from the album "Piano Man," released by Columbia Records.

behaviors such as spreading the arms remain throughout, although the dance slows to match the tempo of *Piano Man* (see "transition clip").

There are generally several interesting dances each generation; therefore, further evolution is most naturally directed toward curiosity-driven exploration rather than a defined goal. Thus an entertaining strategy for Dance Evolution is undirected search, i.e. selecting the most interesting or fun behavior.

Accordingly, five runs, evolved over two songs each, were executed with such a strategy. Every initial ANN, with no prior training, contains 51 nodes and 578 connections. The chance of adding a node is 15% and the chance of adding a connection is 25%. After 10, 20, 30, and 40 generations, the average number of nodes and connections was 54.48 nodes and 579.76 connections, 54.68 nodes and 583.56 connections, 55.28 nodes and 585.88 connections, and 56.08 nodes and 588.36 connections, respectively. Because the songs were on average 4.1 minutes, the average number of generations per minute was 6.1, which means approximately one generation every 9.8 seconds. This short duration suggests that subjective impressions of dance can form quickly. The dances generated (see "variety clip" featuring *Walk Like an Egyptian*[9]) included splits, intricate arm and leg movements, head bobbing, toe tapping, and more, usually in concert.

In general, as evolution progresses, behaviors appear more correlated and smooth; which is both a function of selection and the added network structure.

5 Discussion

Music is composed of a series of periodic spikes in sound energy of different frequencies. Humans dance by reacting to these events by creatively mapping them to motor outputs. Similarly, Dance Evolution evolves ANNs that react to audible spikes of different frequencies through motor outputs.

Dance Evolution applies AI to a unique domain. Both the domain and the interactive interface minimize the risk of user fatigue, allowing Dance Evolution to exploit human intuition to solve an otherwise poorly defined problem.

A promising future extension to Dance Evolution is to embed knowledge of the inherent symmetry of the body into the genetic encoding. Such *indirect encoding* [15,16] would bias the networks toward producing dance sequences that are partially symmetric, which may increase their natural appeal.

One significant practical application of this achievement is within the video game industry. Games based on music such as *Guitar Hero* and *Rock Band* are becoming increasingly popular. For game designers, the capability to produce a large variety of dancers to raw audio recordings can potentially enhance such games significantly and speed up their development.

Perhaps most interesting is that the functional perspective has also been shown plausible in music (as a function of time [2]), and art (as a function of space [4]). By demonstrating that this model can work, Dance Evolution suggests that human creative expression is indeed possible to model in AI.

[9] *Walk Like and Egyptian* is from the album "Different Light," released by Columbia Records.

6 Conclusion

This paper presented Dance Evolution, a program that takes advantage of IEC and ANNs to approach the subjective problem of learning to dance. Additionally, the paper described the algorithm implemented in Dance Evolution to process raw audio files into a form suitable for ANN controllers. Dance Evolution proved capable of training both specific and varied dances on a range of songs. Because these ANNs simply express functions of the music, the dances that they produce are naturally coupled to the music.

References

1. Kurath, G.P.: Panorama of dance ethnology. Curr. Anthropol. 1(3), 233–254 (1960)
2. Hoover, A.K., Rosario, M.P., Stanley, K.O.: Scaffolding for interactively evolving novel drum tracks for existing songs. In: Giacobini, M., et al. (eds.) EvoWorkshops 2008. LNCS, vol. 4974, pp. 412–422. Springer, Heidelberg (2008)
3. Romero, J., Machado, P. (eds.): The Art of Artificial Evolution: A Handbook on Evolutionary Art and Music. Natural Computing Series. Springer, Heidelberg (2007)
4. Secretan, J., Beato, N., D'Ambrosio, D.B., Rodriguez, A., Campbell, A., Stanley, K.O.: Picbreeder: Evolving pictures collaboratively online. In: CHI 2008: Proc. of the 26th annual SIGCHI, pp. 1759–1768. ACM, New York (2008)
5. Sims, K.: Artificial evolution for computer graphics. In: Proc. of SIGGRAPH, pp. 319–328. ACM Press, New York (1991)
6. Yu, T., Johnson, P.: Tour jeté, pirouette: Dance choreographing by computers. In: Cantú-Paz, E., Foster, J.A., Deb, K., Davis, L., Roy, R., O'Reilly, U.-M., Beyer, H.-G., Kendall, G., Wilson, S.W., Harman, M., Wegener, J., Dasgupta, D., Potter, M.A., Schultz, A., Dowsland, K.A., Jonoska, N., Miller, J., Standish, R.K. (eds.) GECCO 2003. LNCS, vol. 2724, pp. 156–157. Springer, Heidelberg (2003)
7. Takagi, H.: Interactive evolutionary computation: Fusion of the capacities of ec optimization and human evaluation. In: Proc. of the IEEE, pp. 1275–1296 (2001)
8. Dawkins, R.: The Blind Watchmaker: Why the Evidence of Evolution Reveals a Universe Without Design. W. W. Norton (September 1986)
9. Sims, K.: Evolving virtual creatures. In: SIGGRAPH 1994: Comp. Graph, pp. 15–22. ACM, New York (1994)
10. Taylor, M.E., Whiteson, S., Stone, P.: Temporal difference and policy search methods for reinforcement learning: An empirical comparison. In: Proc. of the Twenty-Second Conference on AI, July 2007, pp. 1675–1678 (2007); Nectar Track
11. Stanley, K.O., Bryant, B.D., Miikkulainen, R.: Evolving neural network agents in the nero video game. In: Proc. of the IEEE 2005, CIG 2005 (2005)
12. Stanley, K.O., Miikkulainen, R.: Evolving neural networks through augmenting topologies. Evolutionary Computation 10, 99–127 (2002)
13. Scheirer, E.D.: Tempo and beat analysis of acoustic musical signals. Journal of Acoustical Society of America 103(1), 588–601 (1998)
14. Balogh, J., Dubbin, G., Do, M., Stanley, K.O.: Dance evolution. In: Proceedings of the Twenty Second AAAI AI Video Competition. AAAI Press, Menlo Park (2007)
15. Stanley, K.O., D'Ambrosio, D.B., Gauci, J.: A hypercube-based indirect encoding for evolving large-scale neural networks. Artificial Life 15 (2009)
16. Stanley, K.O., Miikkulainen, R.: A taxonomy for artificial embryogeny. Artif. Life 9(2), 93–130 (2003)

Jive: A Generative, Interactive, Virtual, Evolutionary Music System

Jianhua Shao[1], James McDermott[2], Michael O'Neill[2], and Anthony Brabazon[2]

[1] University of Nottingham
dustin.shaojianhua@gmail.com
[2] University College Dublin
jamesmichaelmcdermott@gmail.com, m.oneill@ucd.ie, anthony.brabazon@ucd.ie

Abstract. A novel paradigm and system for interactive generative music are described. Families of musical pieces are represented as functions of a time variable and several variables under user control. Composition/performance proceeds in the following two stages. Interactive grammatical evolution is used to represent, explore, and optimise the possible functions. The computer mouse or a Wii-controller can be used for real-time interaction with the generative process. We present rationale for design decisions and several pieces of example music.

Keywords: Generative music, evolutionary computation, grammatical evolution, interaction.

1 Introduction

Generative music is music which is not specified by a score, but by an algorithm, a set of rules, a set of processes, a mapping from randomness, or some other such method. Collins [4] provides a good introduction, quoting a definition of generative art as art that is "generated, at least in part, by some process that is not under the artist's direct control" [3]. Of course this requires a definition of "direct". Collins also quotes Sol LeWitt: "the idea becomes a machine that makes the art". This brings to mind a famous remark made in the context of meta-programming: "I'd rather write programs that write programs than write programs." (Richard Sites). Generative art is "meta", in the same sense: the artist creates not a single instance of the work, but instructions with the potential to create a family of instances. Meta-programming is also an example of the observation that constructive laziness is a characteristic of good programmers. In this spirit, we like the implicit description given by Brian Eno, who has been responsible for popularising both the term "generative art" and the musical genre: "[...] I've always been lazy, I guess. So I've always wanted to set things in motion that would produce far more than I had predicted." [6].

Collins [4] also mentions an important distinction, that between interactive and non-interactive generative art. In the latter, the artist cannot intervene after the generative process has begun. Interactive generative music allows intervention and "performance": at one extreme, a musical piece such as Queen's

C. Di Chio et al. (Eds.): EvoApplications 2010, Part II, LNCS 6025, pp. 341–350, 2010.

Brighton Rock might be seen as interactive generative music, where the interaction has been increased to a full instrumental performance and the generative aspect reduced to an extreme echo effect.

Our focus in this paper is a form of interactive generative music where material is created through more typical generative processes—musical processes embodied as algorithms and equations—but interaction by direct manipulation of some of the equations' parameters is possible, via either the mouse or the popular, intuitive Nintendo Wii Remote. Since our system is intended to be usable by anyone with no computer/mathematical background or training, we adopt a point of view characterised by the term "hidden variables". Parameters are not intended to be explicitly understood by the performer. Rather, the performer gradually and implicitly learns their effects in different contexts.

The search for algorithms and equations leading to interesting generative music with viable interaction possibilities is a difficult task even for users with computer experience. The space of possible equations and algorithms is large. We therefore require a tool for navigating it, and we choose interactive grammatical evolution (GE), a form of interactive evolutionary computation (IEC) which has been successful in previous applications [12,2,5,10,7]. Our system, called "Jive" (for "generative, interactive, virtual, evolutionary") thus allows two levels of composition/performance. First, the creation of the generative piece itself is a compositional process done through IEC. It fixes many aspects of the "family of instances". Secondly, performance of a particular instance is done by live control of hidden variables. During the IEC process, many short experimental instantiations will be created as the user comes to grips with the possibilities presented by the evolving population. This two-phase process, using IEC to create interactive generative music, is the central novel contribution of this paper.

The remainder of this paper is laid out as follows. Previous work is reviewed in Sect. 2. The Jive system is described, with motivation and examples for its design decisions, in Sect. 3. Results obtained using this system, and refinements based on their success and failure, are given in Sect. 4. The final sections contain discussion, conclusions and future work.

2 Previous Work

Evolutionary approaches to music generation are well-known [12,2]. Generative processes such as L-systems have been explored both within and without the evolutionary context [11]. Magnus' "Evolutionary Musique Concrète" [9] and many others have used evolutionary dynamics as the primary means of driving the development of music over time. Others have used non-interactive EC with computational fitness functions [5]. These differ from the approach adopted here, where we see interactive EC as a tool, and the generative aspect of the music could exist independently of EC.

The *Genophone* system [10] has some of the same aims as that explored in the present work, in that both *performance mappings* and material are created. There are two major differences. *Genophone* operates at the level of sound

synthesis parameters, whereas our focus is on score-level generation. Also, our representation, using context-free grammars, is entirely different.

However the most direct source of inspiration for this work is the NEAT-based "compositional pattern-producing network" approach [15,7]. Complex networks of functional relationships map input variables to plausible, realistic output music (or graphical art, etc.). The functional networks are created using IEC. Although the input variables are derived from pre-written input music (the aim is to automatically produce rhythm tracks to accompany existing music), the approach has more general potential to map any input parameters to output music. This is the point we take up.

The commercial system *Noatikl*, which is descended from *Koan*, used to create Eno's seminal *Generative Music 1*, allows user interaction with generative pieces according to a non-evolutionary paradigm.

Generative grammars, like the context-free grammar used in our grammatical evolution approach, have been extensively used for both the analysis and generation of music, for example by Lerdahl and Jackendoff [8]. However, we wish to draw an important distinction here. In our work, we believe for the first time, the generative grammar is used to create code—specifically a set of arithmetic and boolean functions, which drive the generative process. It is not used to create musical material directly, and so our work has no direct bearing on grammatical theories of music.

3 The Basic Jive System

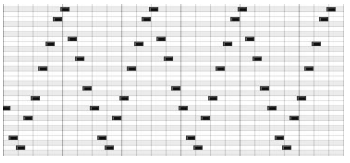
The Jive system consists of four components: generative, interactive, virtual, and evolutionary. They are described in the following four sections.

3.1 Generative

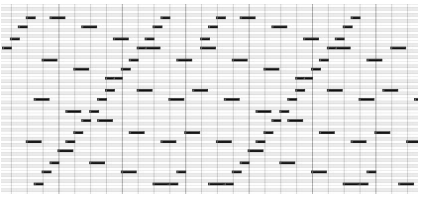
Fundamentally, Jive is a generative music system in which music is a function of time, and time is seen as a discrete variable. In the simplest possible case, a function such as $f(t) = 40 + 7\sin(2\pi t)$ or $g(t) = t \bmod 60$ will create a piece of music, albeit a very boring one. Here, the output of functions f and g is a number, which we round (if necessary) to an integer and interpret as a MIDI note number. We make one immediate improvement, however: since we wish to minimise the random feel associated with much generated music, we will map the integer to a diatonic scale.

There are several routes towards creating more interesting music. First, we require the ability to play *any* succession of notes. It is well-known that a combination of sinusoidal functions is sufficient to represent any periodic function of one variable (and since a piece of music will be finite in length, the periodicity requirement is unnecessary). This universality is the property we require. Another possibility, which leads more immediately to musical results, is to use combinations and variations of a linear function like $h(t) = a + q(t \bmod p)$. By varying the values of a (a *pitch offset*), q (a *scaling* factor), and p (a *periodicity*

parameter), we can produce a function of one variable, time, which produces ascending and descending scales, and in general piecewise-linear sequences, as depicted in Fig. 1(a)[1].



(a) Simple piecewise-linear results obtainable using a single h function



(b) A single voice can become arbitrarily complex by summing multiple h functions.

Fig. 1. Examples demonstrating output of a single voice

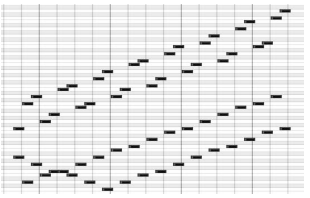
The important role played by the mod operator is to provide periodicity. A similar role might have been played by a sin function as noted above. A single instance of the h function chunks time into periods, giving our generative piece repetitive pattern. We can also sum multiple instances of the function, with varying values for a, q, and p, to obtain a function $h(t) = \sum_i h_i$, again of one variable, time, which can produce any desired sequence of single notes[2]. In order to keep things rhythmically coherent, we will constrain p to take on values of the form $p = p_1$ or $p = p_1 p_2$, where p_1 and p_2 are small integers (2, 3, 4, or 6), and their values are fixed for a given piece of music. They function as the primary and secondary *rhythmic characteristics* of the piece. The summed h function will have periodicity equal to the lowest common multiple (LCM) of its component periodicities: the LCM is constrained by this scheme to be a relatively low value. The types of results obtainable using this scheme are depicted in Fig. 1(b).

The next step is to add a secondary pattern at a longer time scale. We achieve this using the *quotient* function. We alter our summed function to allow expressions of the form $h_i(t) = (t \text{ quot } a) + q(t \bmod p)$. The quot function performs integer division. During the time-steps 4-7, the expression t quot 4 has a constant value, 1, which is used as an offset. This causes any simple pattern created by q, t and mod to be repeated at different offsets, giving a harmonic feel, as in Fig. 2(a).

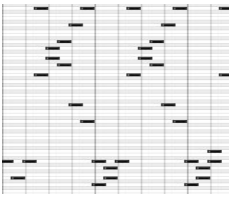
Multiple voices are not difficult to achieve: we can simply create new functions $h' = \sum h'_i, h'' = \sum h''_i$, and so on. For now we stick to three such functions. Each calculates pitches using independent parameter values, so a higher degree

[1] These clips demonstrating successive levels of development, together with software, example grammars, and four demo pieces, are available at `http://sites.google.com/site/odcsssjian2009/`.

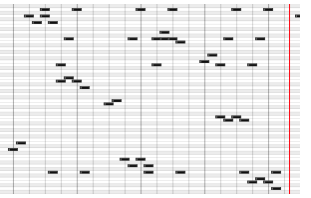
[2] This is like genetic programming-style symbolic regression, for music.



(a) The *quotient* function gives a feeling of harmonic movement.



(b) Multiple independent voices.



(c) Omitting some notes gives a first approximation of phrases.

Fig. 2. Simple examples of harmony and phrasing

of complexity in pitch-movement can now arise. There is one exception: periodicity is still constrained by the primary and secondary rhythmic characteristic parameters. Some examples are given in Fig. 2(b).

We add the possibility of rests, rather than a constant succession of notes. We add an independent boolean function to each voice, which calculates a true or false value at each note indicating whether to play the pitch calculated for that voice by its numerical function, or to remain silent. The boolean function is created, for now, using `<`, `<=`, `==`, and other comparisons of t against arithmetic expressions similar to those used for pitch calculation. This allows much more interesting phrase-structures to emerge. The music being generated is still very simple but is suddenly beginning to sound like music. Some examples are depicted in Fig. 2(c).

Finally, for now, we can make the form of our equations entirely open-ended. Instead of using the fixed function h with varying numerical parameters, as described earlier, we can write a context-free grammar which creates arbitrary functional expressions in our input variables, using other periodic functions such as sin() and cos() combined with multiplication, addition, subtraction and (protected) division. It quickly becomes impossible to predict the style of an individual from inspection of its code, but this approach has the advantage of being more open-ended. A good generative system will sometimes surprise its creator, and this is more likely to occur using an open-ended representation. It is certainly capable of producing compositions which the authors of this paper could not have written by hand.

3.2 Interactive

Since the system as described in the previous section creates material as a function of time rather than from an explicit score, it may be regarded as generative. However it is capable only of producing simple periodic pieces. A key aim in this research is to allow the user/performer to interact with the generative music as it plays. This will allow the material to change and develop over time. Hence, we add to our system some continuous-valued variables representing user input.

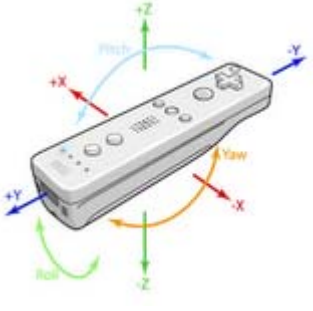
These are then available to be incorporated into the numerical expressions for pitch and boolean expression for note presence/absence.

We provide multiple variables, which will be controlled by the user as described in the next section, and we allow some flexibility in the way they are used. In general, they can be used as offsets or scaling parameters to existing parameters in our equations. Each input variable may be used more than once in our various equations, but there is no requirement that every variable be used at all. This indirect, optional, and multiple usage of input variables we refer to as "hidden variables", as discussed in more detail in Sect. 5.

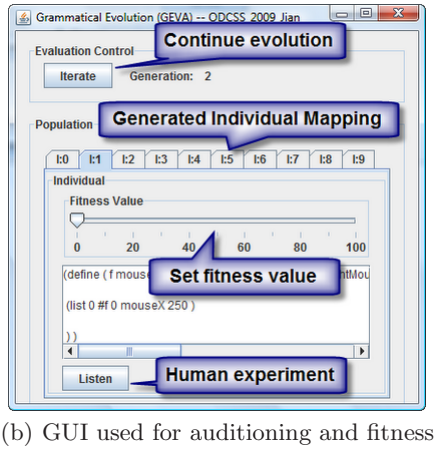
3.3 Virtual

The continuous user-input variables referred to in the previous section are the user/performer's main means of interaction with a playing piece (it is also possible to change tempo manually, but this is not of interest here). Although the boundary between sequencer, performance system, and instrument is blurred by the system we have described, we refer to Jive as a *virtual instrument*. This term is typically used to mean playable musical instruments which are "disembodied" or implemented purely in software or electronics.

The user/performer can perform with the Jive system in two ways. The mouse is a simple method: it provides X and Y values. Although the system has been programmed to read multiple mouse buttons also, these are not used in any current configurations. The Nintendo Wii remote control, also known as the "wiimote", is a more sophisticated option. It can provide either absolute position or accelerometer data in three dimensions, as well as multiple buttons. Again, we currently use only a subset: the absolute X, Y and Z values. The wiimote, shown in Fig. 3(a), does not require a GUI. It is interfaced to our software using the WiiRemoteJ library (`http://www.world-of-cha0s.hostrocket.com/WiiRemoteJ/`).



(a) The Nintendo Wii Remote



(b) GUI used for auditioning and fitness evaluation

Fig. 3.

Both controllers are sufficient for their intended purpose here. Small movements tend to lead to small changes in output; larger movements can lead to entirely different behaviours. Well-timed movements are required, but great dexterity is not. More interesting interaction possibilities can be created by sending not only current 2D or 3D position data, but also a buffer of recent positions, as input variables to the equations. In the case of the wiimote, for example, a buffer of size 3 now gives us 9 continuous input variables in addition to time, and the pitch equations and boolean note presence/absence functions can use any or all of them. One advantage is that smoother changes occur—however there is a trade-off with a loss of fine control, since the sound being played at each instant is now dependent on previous controller movements. Sometimes the periodic patterns that occur, using the buffer, seem less rigid and more natural. It allows for potential "emergent gesture recognition", since in principle relative movement as well as absolute position are now available to the system.

3.4 Evolutionary

The system as described so far is complete, in that it can be used to create and to perform pieces of music. It is not user-friendly: most musicians, if they are human, are not capable of performing symbolic regression in their heads while composing. Instead, we now introduce the last component of the system, interactive evolutionary computation (IEC). This algorithm works in the same way as typical EC, but allows the user to specify individuals' fitness values, or to perform selection directly. Because it generates multiple individuals in an iterative process, with the user required only to assess their quality, it avoids the need for the user to write equations directly when creating a piece.

The IEC GUI used for generation, auditioning and selection of pieces, and iteration of the algorithm, is shown in Fig. 3(b). It remains to describe the representation we have chosen for our individuals.

Grammatical evolution (GE) is a proven technique for representing code in diverse languages with arbitrary constraints on its form [14]. The language syntax and the allowable or desirable syntactic forms (such as our equations) are specified in a context-free grammar in Backus-Naur Form. An individual's genome is an integer array, which specifies the grammar productions to be chosen during the derivation process.

In our work, GE is implemented using the freely-available GEVA software [13], written in Java. GEVA was configured to use a population size of 10, an unlimited number of generations, a one-point crossover rate of 0.7, an int-flip mutation rate of 0.02, and generational replacement. Our equations and boolean functions, together with some boilerplate code for dealing with input variables and time, were specified as BNF grammars producing programs in the JScheme language [1]. The grammars we have used are available for download: see Sect. 6. The jMusic API is used to take the values returned by execution of JScheme individuals and translate them into MIDI.

Although any given version of our equations might be represented as a linear, real-valued genome—one gene per parameter—we have found that the GE

approach, using a BNF grammar, gives much greater flexibility. The process of altering the *form* of an equation is much easier with a grammar, compared to re-writing code. Grammars can also be entirely *open-ended*, allowing (for example) a sum of multiple expressions h'_i, or an XOR-combination of boolean note presence/absence—and the number of such expressions may be unknown in advance and left for evolution to determine. This is much more difficult or impossible using a GA-style representation.

4 Results and Refinements

The basic system as described so far is already capable of producing some music sufficiently interesting to be worth describing.

The system is clearly lacking in the area of rhythm. Phrases are essentially created by knocking notes out, i.e. switching on a "rest" flag for one or more notes, leaving gaps between sub-sequences which the ear then interprets as isolated phrases. In the demo pieces RD_0 and RD_2, we attempt to work against this shortcoming. An open-ended grammar makes a good deal of complexity available during the performance phase. Multiple basic behaviours were available in the mouse's (x, y) plane, for example an "ascending" behaviour, partly controlled by relative position; a "stop" or steady-state behaviour; and a dense periodic pattern. Composition in this case consisted not only of choosing which behaviour to switch to, but when. It was possible to take advantage of this to create higher-level structure in the pieces. It was also possible to use a voice creating sparse material, whose pitch was under direct control, to act as a melody. This put the continuous material in the background, to some extent overcoming the "continuous stream" feel.

The demo piece ML_0 uses the "memory" facility and is very complex, chaotic at times. The piece was evolved through approximately 22 generations.

In the final demo piece, MD_0, we have added two new features. A voice can now choose to play a chord, chosen from a small selection of major and minor triads and sevenths. A drum grammar was also created and used in Jive to produce a looped rhythm track. Note that all demo pieces have been rendered in an external program using manually-chosen synthesizers and effects.

5 Discussion

Generative music at its best brings out the abstract patterns which underlie (but do not solely *constitute*) many (but not *all*) types of music. The "hidden variables" approach adopted here has some satisfying results in this context. A change in a single hidden variable may have multiple effects. This imposes a type of large-scale coherence on the output—multiple voices may react simultaneously to a change in a user parameter, for example. This can lead to a re-interpretation of non-generative music. When several instruments in an orchestral piece crescendo and then switch to a new section, one could interpret the multiple voices as manifesting the effects of shared, hidden variables.

An interesting aspect of generative music is the blurry boundaries it creates between composer, performer, and listener. This is augmented in our work by the explicitly interactive element. The user of the Jive system plays roles including programmer (editing the grammar to fix a different rhythm), critic and composer (auditioning and selecting generated pieces during interactive evolution), and finally composer, conductor and performer (interacting with a fixed piece). We have occasionally used terms like "user/conductor" and "user/performer" to emphasise this blurriness.

Sometimes the control available to the user/performer seems very crude. While performing, it is not (in general) possible to insert or delete arbitrary notes. This apparent drawback has two, perhaps unexpected advantages. Firstly, since the system generates material continuously, with precise timing and no "wrong" notes, the user/conductor is freed from low-level details. The mouse and to a lesser extent the wiimote are, after all, inadequate controllers for the type of dextrous performance required by typical musical performance.

Secondly, the user/conductor gains a higher-level type of control despite the lack of low-level detail. As discussed in Sect. 4, the user/composer has control not only of which behaviour to switch to, but also when. He or she is not only performing the gestures which correspond to desired sonic results, but is reacting to the current musical context created by the ongoing algorithm and by his/her previous actions. This ability allows the censoring of undesired sections and the creation of complex musical syntax, such as call-and-response patterns among the several behaviours. The user's control of timing is sufficiently fine to allow the behaviours to work together in interesting ways.

6 Conclusions and Future Work

In this paper, we have presented the Jive system, a novel system and paradigm for interactive generative music. It uses IEC with aesthetic selection to create a generative process, and is a virtual instrument for real-time expression and performance. We have explained the design decisions and shown examples of the possible outputs. We consider that the styles of music possible with Jive, though limited, are interesting and characteristic.

Any virtues the system has arise from two sources. Firstly, our representation includes set of primitives which are not in themselves musical but function well as building-blocks for music. Secondly, we use a two-phase composition/performance process. The user/performer has some ability to overcome failings present in the composed piece, as discussed in the context of the demo pieces. The choosing of constraints and the exploration of what can be done despite them is characteristic of many art-forms. There is also the possibility of a third phase, in which the user edits the grammar in order to constrain the possible outputs. Of course this is only available to technically-oriented users.

The Jive system is not finished, and there are several possibilities for improvement. In our future work, our first priority is to address the system's shortcomings in the area of rhythm. We intend to add the possibility of more complex rhythms with the addition of triplets, quintuplets, etc. More complexity in the

phrasing and grouping of notes is also required. The interactive evaluation of multiple individuals per generation is a time-consuming task. There is some scope for automatic evaluation, for example by filtering individuals which give over-dense or inadequately varied material.

We believe that the generative, hidden-variable approach used here is a good way of making musical games, similar in spirit to popular "guitar karaoke" games but far more open-ended and expressive.

Acknowledgements

Jianhua Shao's work was funded by Science Foundation Ireland under the OD-CSSS scheme. James McDermott is funded by the Irish Research Council for Science, Engineering and Technology under the Empower scheme.

References

1. Anderson, K., Hickey, T., Norvig, P.: JScheme and JScheme documentation, http://jscheme.sourceforge.net/jscheme/main.html (last accessed October 30, 2009)
2. Bentley, P.J., Corne, D.W. (eds.): Creative Evolutionary Systems. Morgan Kaufmann, San Francisco (2002)
3. Boden, M.: What is generative art?, cOGS seminar (October 2007)
4. Collins, N.: The analysis of generative music programs. Organised Sound 13(3), 237–248 (2008)
5. Dahlstedt, P.: Autonomous evolution of complete piano pieces and performances. In: Proceedings of Music AL Workshop (2007)
6. Eno, B.: Generative music, lecture, San Francisco, USA (June 1996), http://www.inmotionmagazine.com/eno1.html
7. Hoover, A., Rosario, M., Stanley, K.: Scaffolding for interactively evolving novel drum tracks for existing songs. In: Giacobini, M., Brabazon, A., Cagnoni, S., Di Caro, G.A., Drechsler, R., Ekárt, A., Esparcia-Alcázar, A.I., Farooq, M., Fink, A., McCormack, J., O'Neill, M., Romero, J., Rothlauf, F., Squillero, G., Uyar, A.Ş., Yang, S. (eds.) EvoWorkshops 2008. LNCS, vol. 4974, pp. 412–422. Springer, Heidelberg (2008)
8. Lerdahl, F., Jackendoff, R.: A Generative Theory of Tonal Music. MIT Press, Cambridge (1983)
9. Magnus, C.: Evolutionary musique concrète. In: Rothlauf, F., Branke, J., Cagnoni, S., Costa, E., Cotta, C., Drechsler, R., Lutton, E., Machado, P., Moore, J.H., Romero, J., Smith, G.D., Squillero, G., Takagi, H. (eds.) EvoWorkshops 2006. LNCS, vol. 3907, pp. 688–695. Springer, Heidelberg (2006)
10. Mandelis, J., Husbands, P.: Genophone: Evolving sounds and integral performance parameter mappings. International Journal on Artificial Intelligence Tools 15(4), 599–622 (2006)
11. McCormack, J.: Evolutionary L-systems. In: Hingston, P.F., Barone, L.C., Michalewicz, Z., Fogel, D.B. (eds.) Design by Evolution: Advances in Evolutionary Design, pp. 169–196. Springer, Heidelberg (2008)
12. Miranda, E.R., Biles, J.A. (eds.): Evolutionary Computer Music. Springer, Heidelberg (2007)
13. O'Neill, M., Hemberg, E., Bartley, E., McDermott, J., Brabazon, A.: GEVA: Grammatical evolution in java. SIGEVOlution 3(2), 17–22 (2008)
14. O'Neill, M., Ryan, C.: Grammatical Evolution: Evolutionary Automatic Programming in an Arbitrary Language. Kluwer Academic Publishers, Dordrecht (2003)
15. Stanley, K.O.: Compositional pattern producing networks: A novel abstraction of development. Genetic Programming and Evolvable Machines 8(2), 131–162 (2007)

A Neural Network for Bass Functional Harmonization

Roberto De Prisco, Antonio Eletto, Antonio Torre, and Rocco Zaccagnino

Dipartimento di Informatica ed Applicazioni
University of Salerno, Italy
{robdep,zaccagnino}@dia.unisa.it,
{a.eletto3,a.torre14}@studenti.unisa.it

Abstract. This paper presents the design, implementation and testing of a neural network for the functional harmonization of a bass line. The overall network consists of three base networks that are used in parallel under the control of an additional network that, at each step, chooses the best output from the three base networks.

All the neural networks have been trained using J.S. Bach's chorales. In order to evaluate the networks, a metric measuring the distance of the output from the original J.S. Bach's harmonization is defined.

1 Introduction

Writing computer programs that solve musical problems has been object of study since the invention of the computer. The literature has plenty of work that use various computer science techniques (like formal grammars, genetic algorithms, pattern matching, cellular automata, neural networks) to tackle musical related problems. In particular we cite the work of D. Cope (e.g. [2]) and E.R. Miranda (e.g., [9]) as examples of research work in computer music.

The musical problem considered in this paper deals with music compositions made up of 4 voices (soprano, alto, tenor and bass). Famous examples of this type of compositions are J.S. Bach's chorales[1].

Related work. Automatic composers for 4-voice music have been provided, for example, by Ebcioglu [4] and Schottstedt [12] using rules and expert-systems. Another system capable of composing 4-voice chorales (and not only) is the EMI system by Cope [2]; the EMI system uses a combination of various techniques (formal grammars, rules, music analysis, pattern matching). McIntyre [8], Wiggings, Papadopoulos, Phon-Amnuaisuk and Tuson [13] and De Prisco and Zaccagnino [3] use genetic algorithms. Phon-Amnuaisuk [10] investigated the use of heterogeneous cellular automata.

The works cited above do not use neural networks. The papers by Lehmann [6] and [7] use neural networks to find an harmonization of a given melody line. The

[1] We remark that Bach would start composing from a soprano line and the problem considered in this paper starts with the bass line as input.

C. Di Chio et al. (Eds.): EvoApplications 2010, Part II, LNCS 6025, pp. 351–360, 2010.

neural network used by Lehmann is "sequential" in the sense that the information used for harmonizing at time t is whatever happened up to time t and thus uses no information about the continuation of the melody line.

Hild, Feulner and Menzel [5] describe a neural network called Harmonet for harmonizing melodic lines in the style of J.S.Bach. For harmonizing at time t Harmonet uses information about the next melody note and only the harmonic information up to time t. Harmonet does not use specific harmonic information about the next (possible) chord(s) and the current tonality and degree.

This paper. The specific problem considered in this paper is the following: take as input a bass line and find, for each note of the bass, an appropriate chord (only the name, that is the *function*, of the chord and not the notes' positions). For simplicity, and without loss of generality, we consider rhythms based on quarter notes. Notes with a smaller length are considered only if they start on a quarter beat (otherwise they are just passing notes and they are ignored for the harmonization). The problem can formally be defined as follows: given a sequence of quarter notes $b_1, b_2, \ldots, b_n$ for the bass line find an appropriate chord C_i for each bass note b_i, for $i = 1, 2, \ldots, n$. Figure 1 shows an input (left side) and the same input with the corresponding output (right side).

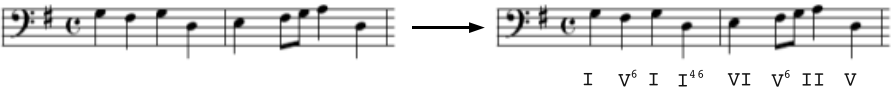


Fig. 1. An input bass line (left) and a functional harmonization of the input (right)

The chords we consider are all the possible chords with 3 notes (triads) or 4 notes (quadriads). We provide a neural network that solves the above problem.

Among related papers, the closest work are Harmonet [5] and [6,7]; note that the exact problem considered in this paper is slightly different because the input is a bass line and not a melody line. A more important difference is that our network does use information about the next bass note or even the next (possible) chords and their context (tonality and degree). As a matter of fact, the tests presented in this paper have shown that using information only on the next bass note is not good enough and we obtain much better results by looking at the next (possible) chords. This technical aspect is newer with respect to the above cited previous works.

The proposed network has been evaluated defining and using a metric that measures (or at least, attempts to measure) the distance between the output of the network and the harmonization made by J.S. Bach. A formal comparison with the output of previous work has not been possible because the cited previous works do not contain neither complete examples of output nor any other formal evaluation of the proposed neural networks.

Background. Due to space constraints, we assume that the reader is familiar with both basic music notions and with neural networks.

In particular, the reader should be familiar with the tempered music system, the notes (A, A#, B, C, C#, D, D#,E, F, F#, G, G#), the concepts of scale and tonality, the degrees of a scale (I, II, III, IV, V, VI, VII), the notion of chords, chord inversions (e.g. II^6, V^{46}, I^{357}) and with harmony rules. We refer the reader to any good book on harmony, like [11], for a discussion about all these notions.

On the neural networks side the reader should be familiar with the functioning of an artificial neural network. The network presented in this paper is a fully connected three-layer feed-forward neural network, using a sigmoid activation function The learning process has been obtained with the back-propagation algorithm.

We have implemented in Java a general training environment for neural networks. This allowed us to explore several models and even use them in parallel as we will explain in the next section.

2 The Neural Networks

Recall that our objective is that of constructing a neural network capable of assigning an appropriate chord to each bass note in a given input bass line. There are particular sequences of chords that are used more often; for example the sequence II-V-I, is very common. The length of such very common sequences usually ranges from 2 to 4 chords. So, to analyze a piece harmonically, one can focus the attention on the harmonic relationship between pairs, triples and quadruples of consecutive chords.

This paper uses 3 models, called *base models*. These base models differ in the number of previous chords used for the prediction. Model 1 uses only one chord, model 2 uses two chords and model 3 uses three chords:

Base Model 1: $C_i{=}f(C_{i-1},b_i)$
Base Model 2: $C_i{=}f(C_{i-2},C_{i-1},b_i)$
Base Model 3: $C_i{=}f(C_{i-3},C_{i-2},C_{i-1},b_i)$

where b_i and C_i are, respectively, the bass note and the chord on beat i.

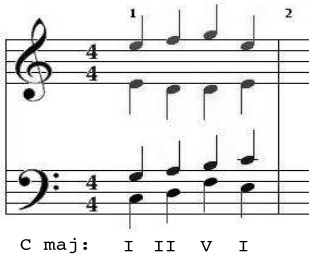
Obviously, each base model learns different harmonic information, because each one of them uses different input knowledge. The base models were applied individually to obtain different functional harmonization of the same bass line. In order to use all three base models in parallel, an additional neural network, that will be described later, has been implemented.

A crucial step in the design of a neural network is the choice of the representation of the data. The relevant data unit is the chord that is assigned to a bass note together with information about the tonality; such information is stored in a *Chord* data structure as follows:

- *Bass*: the bass of the chord, is coded as a binary number between 0 and 11, corresponding to the binary encoding of the MIDI value of the note, modulo 12: $C \equiv 0000, C\# \equiv 0001, D \equiv 0010, \ldots\ldots, A\# \equiv 1010, B \equiv 1011$;

– *Tonality*: the current tonality, encoded the same way as the bass note;
– *Quality of tonality*: major or minor, specified with a bit (1=major, 0=minor).
– *Degree*: the chord's degree encoded as a binary number between 1 and 7: $I \equiv 001, II \equiv 010, III \equiv 011, \ldots\ldots, VII \equiv 111$.

Figure 2 provides an example.



	Bass	Tonality	Maj/Min	Deg.
Chord 1	0000	0000	1	001
Chord 2	0010	0000	1	010
Chord 3	0110	0000	1	101
Chord 4	0101	0000	1	001

Fig. 2. Binary representation of the Chord data structure

Base model 1 bases its prediction for the current bass note on the previous chord. In order to start-up the network a first chord is needed. Model 1 uses chord I in the starting tonality (this is what happens in many compositions). In order to start-up the network using base model 2, the first and the second chord are needed. Model 2, uses chord I as the the first chord and obtains the second chord using the base model 1. Similarly, base model 3, uses chord I as first chord and obtains the second and the third chord exploiting base models 1 and 2.

The three base models have been used to obtain functional harmonizations of input bass lines. One would expect base model 3 to be the best one since it uses more information to predict the next chords. However the results of the tests have shown that this is not the case. Indeed there were portions of the output that were better in the other models. Since no one of the three models fared better than the other, we used them in parallel in the following way. In order to predict the next chord C_i, each base model is used to get one of three possibilities, C_i^1, C_i^2, C_i^3, for the chord at position i. To choose one of these chords a new neural network, called *control* network, is used. We designed two control networks, the first one that exploits only the subsequent bass note b_{i+1} and the second one that exploits information about chords C_i and C_{i+1}. Next we will describe both networks.

Bass Control Network (BCN). In order to choose the chord C_i for the bass note b_i, the BCN uses the models 1,2 and 3 to get three possible chords C_i^1, C_i^2, C_i^3 for b_i.

In order to choose one of these three chords the BCN has been trained to solve the following problem: given a pair of consecutive chords C_{i-1}, C_i provide a guess for the next bass note b'_{i+1}. We write b'_{i+1} because this note can be

different from b_{i+1}. Then we compute such a guess for the following pairs: b^1_{i+1} = BCN (C_{i-1}, C^1_i), b^2_{i+1} = BCN (C_{i-1}, C^2_i), b^3_{i+1} = BCN (C_{i-1}, C^3_i).

The BCN chooses the $\bar{j}$, $\bar{j} = 1, 2$ or 3, that minimizes the Hamming distance between $b^{\bar{j}}_{i+1}$ and b_{i+1}; recall that the bass notes are represented with a binary number. Finally the chord selected for b_i is $C^{\bar{j}}_i$.

The results obtained with the Bass Control Network were not satisfactory. We believe that this is due to the fact that the next bass note is too little information to make an educated guess about the chord to select. Thus we designed a more complicated control network which exploits information about the subsequents chords.

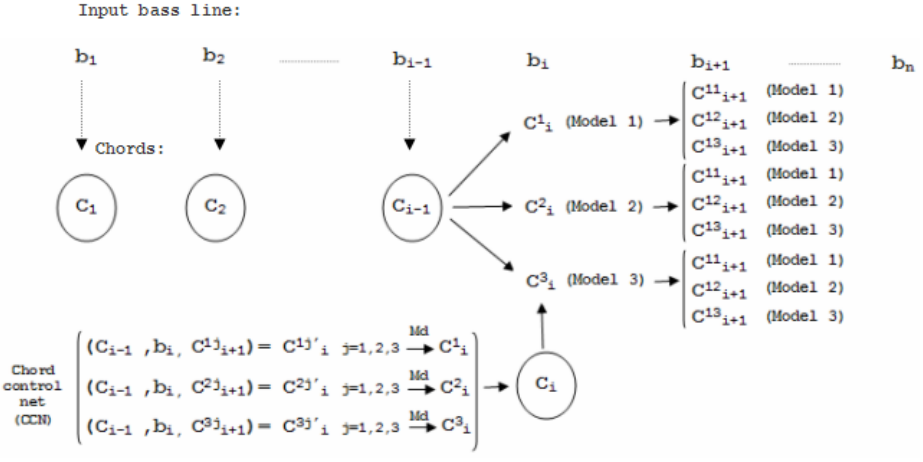


Fig. 3. Control net based on the previous and on the next chord. M_d is the musical distance.

Chord Control Network (CCN). This control network uses as control information not only the bass note at position i but the entire (possible) chord at position i and also the (possible) chord at position $i+1$. Figure 3 illustrates the behavior of the Chord Control Network. In the figure, we are in the process of selecting a chord for bass note b_i, that is the CCN has already chosen chords for $b_1, b_2, \ldots$ up to b_{i-1}. Using the three base models to predict the chord for b_i we get the three possibilities C^1_i, C^2_i, C^3_i. Now we use again the three base models to obtain predictions for the chord at position $i+1$. This results into 9 possible chord sequences for b_i and b_{i+1}, namely:

1. C^1_i C^{11}_{i+1} (model 1, model 1)
2. C^1_i C^{12}_{i+1} (model 1, model 2)
3. C^1_i C^{13}_{i+1} (model 1, model 3)
4. C^2_i C^{21}_{i+1} (model 2, model 1)
5. C^2_i C^{22}_{i+1} (model 2, model 2)
6. C^2_i C^{23}_{i+1} (model 2, model 3)
7. C^3_i C^{31}_{i+1} (model 3, model 1)
8. C^3_i C^{32}_{i+1} (model 3, model 2)
9. C^3_i C^{33}_{i+1} (model 3, model 3)

In order to choose one of these 9 possibilities (which will provide the choice for the chord at position i) the Chord Control Network has been trained to solve the following problem: given a triple C_{i-1}, b_i, C_{i+1} provide a guess for C'_i, the chord to give to b_i. Again we use C'_i and not C_i because this is only a guess. We compute such a guess from the following triples:

1. $C_i^{11'} = \text{CCN}(C_{i-1}, b_i, C_{i+1}^{11})$
2. $C_i^{12'} = \text{CCN}(C_{i-1}, b_i, C_{i+1}^{12})$
3. $C_i^{13'} = \text{CCN}(C_{i-1}, b_i, C_{i+1}^{13})$
4. $C_i^{21'} = \text{CCN}(C_{i-1}, b_i, C_{i+1}^{21})$
5. $C_i^{22'} = \text{CCN}(C_{i-1}, b_i, C_{i+1}^{22})$
6. $C_i^{23'} = \text{CCN}(C_{i-1}, b_i, C_{i+1}^{23})$
7. $C_i^{31'} = \text{CCN}(C_{i-1}, b_i, C_{i+1}^{31})$
8. $C_i^{32'} = \text{CCN}(C_{i-1}, b_i, C_{i+1}^{32})$
9. $C_i^{33'} = \text{CCN}(C_{i-1}, b_i, C_{i+1}^{33})$

Define the *musical distance* from two chords C_j^k, C_j^z with the same bass b_j as the sum of the distance between the tonality T_j^k of C_j^k and the tonality T_j^z of C_j^z, and the distance between the degree G_j^k of C_j^k and the degree G_j^z of C_j^z.

The distance between two tonalities T_j^k and T_j^z is the distance in the circle of fifths of the tonalities, which is the difference in alterations in their scales. For example, the tonality *C major* (no sharps and flats) and the tonality *G major* (1 sharp) have distance 1, while the tonality *G major* and the tonality *Bb major* (2 flats) have distance 3.

The distance between two degrees G_j^k and G_j^z is calculated as the difference, in absolute value, between the inversion number of b_j in G_j^k and the inversion number of b_j in G_j^z. We define the *inversion number* of a bass b in a degree G as 0 if G is in root position, as 1 if G is in first inversion, as 2 if G is in second inversion, and as 3 if G is in third inversion.

For example consider the bass E, the chord I in C major and the chord III in C major, then the inversion number of E is 1 respect to I and 0 respect III.

So, let Inv_j^k the inversion number of b_j in G_j^k and Inv_j^z the inversion number of b_j in $G^z{}_j$, then the distance between the degrees G_j^k and G_j^z is $|Inv_j^k$ - $Inv_j^z|$.

Since, because of approximation errors, it is possible that the neural network occasionally outputs invalid chords, we also define the distance between two chords C_j^k, C_j^z when one of them is not a valid musical chord; such a distance is defined to be 10. Table 1 summarizes the distance weights used by the CCN.

Finally, the CCN selects $\bar{j}$ and $\bar{k}$ in such way that the musical distance of $C_i^{jk'}$ and C_i^j is minimized. That is $(\bar{j}, \bar{k}) = \texttt{argmin}\{\text{musical-distance}(C_i^{jk'}, C_i^j)\}$. Chord $C_i^{\bar{j}}$ is chosen as the chord for b_i.

Table 1. Chord distances summary; d_T is the tonalities distance d_G the degrees distance

Comparison between 2 chords	Distance
Same chord	0
Same tonality, different degree	d_G
Modulation	d_T
Invalid chord	10

3 Training, Validation and Test Results

The data-set used to train the networks has been taken from J.S. Bach chorales. Hence we expect our network to imitate J.S. Bach's harmonization. The training patterns (single chords, pair of consecutive chords and triples of consecutive chords) were taken at random from several Bach's chorales.

Table 2. Training, validation and test data summary. Note that the error rate in the BCN and CCN lines of the table is only for the control part of the overall networks.

	training patterns	validation patterns	test patterns	test error rate
Base model 1	1070	350	205	21.9%
Base model 2	1043	313	201	12.4%
Base model 3	1000	295	197	10.1%
BCN	1000	340	203	70%
CCN	1050	330	204	9.5%

Table 2 summarizes the sizes of the training, validation and test data sets that we have used and the resulting testing error rate that is the percentage of test patterns that the individual models have not learned.

As we can see from Table 2, the error rate of the BCN is vey high. This makes the network unusable.

Figure 4 shows the evolution of the MSE (Mean Square Error) function during the training and validation for the base model 1 and for the base model 2, while Figure 5 shows the evolution of the MSE function during the training and validation for the base model 3 and for the CCN. All networks were trained in about 200 epochs because, as we can see from the graphs, the MSE functions stabilize (converge) after about 150-200 epochs.

We evaluate the application of the models obtained in the training phase by using them to harmonize the bass line of the following chorales[2]: *BWV* 2.6, *BWV* 6.6, *BWV* 16.6, *BWV* 20.11, *BWV* 32.6, *BWV* 36.4, *BWV* 20.7, *BWV* 43.11, *BWV* 39.7, *BWV* 153.9.

For each one of the bass line, we have run the base networks and the combined network obtained using the three base networks in parallel with the CCN.

In every experiment we have computed the musical distance of the output from the original harmonization by Bach as follows. Let $b_1 \ldots\ldots b_n$ the input bass line. Let $B_1 \ldots\ldots B_n$ be the functional harmonization made by Bach. Let $C_1 \ldots\ldots C_n$ be the functional harmonization made from one our neural networks. For each $i = 1, \ldots\ldots, n$ we calculate the musical distance (as defined in the previous section) between C_i e B_i. The sum of the musical distances between C_i and B_i for each $i = 1, \ldots\ldots, n$ is the total distance.

Figure 6 shows the comparison between J.S. Bach's original harmonization and the CCN functional harmonization for Chorale 32.6. For each note in the bass line the chord by Bach is equal to the chord by CNN, except for the chords

[2] The BWV numeration has been taken from [1].

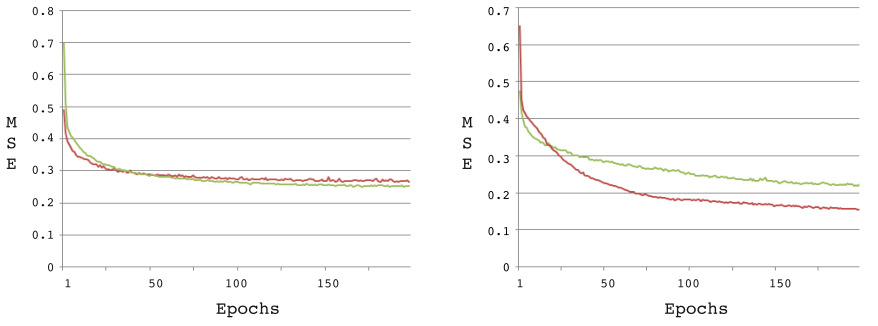


Fig. 4. Training (lighter line) and validation (darker line) errors for base model 1 (left) and base model 2 (right)

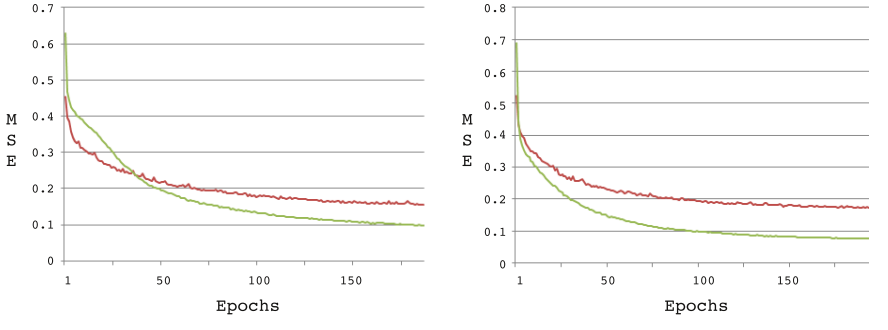


Fig. 5. Training (lighter line) and validation (darker line) errors for base model 3 (left) and the Chord Control Network (right)

that are circled in the figure. The first difference is for the bass note D on beat 4 (in measure 1) where the chord by Bach is I in G major, while the chord by the CCN is V in G major. The tonality distance is 0. The inversion number of D respect to I in G major is 2, the inversion number of D respect to V in G major is 0, so, the grade distance for this pair of chords is $|2-0|$ that is 2. Then the musical distance between this pair of chords is $0+2$ that is 2. Computing in a similar way the other musical distances we have that the total musical distance between J.S. Bach's original harmonization and the CCN functional harmonization for chorale BWV 32.6 is $2+2+1+1+4$, that is 10.

Table 3 provides a summary of the results of the tests. No one of the base models is always better the others. However the combined network obtained using the CCN always produces a much better output.

We observe that for each chorale, $85\% - 90\%$ of the chords provided by the CNN coincide with those of original J.S. Bach's harmonization. As for the chords that are different from those chosen by Bach, the musical distance is always very

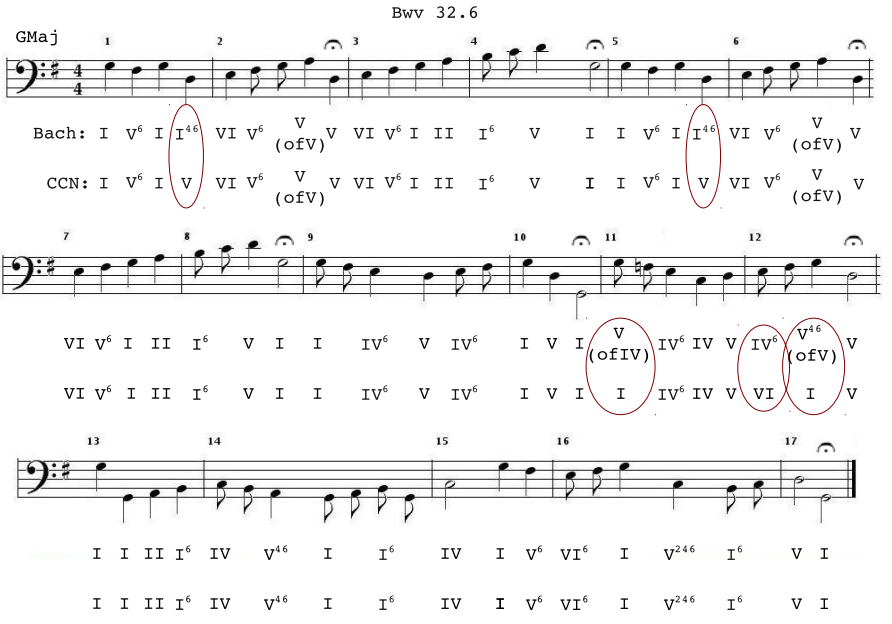


Fig. 6. Differences between functional harmonizations by Bach and by the BCN

Table 3. Distance from J.S. Bach's original harmonization

	BWV 2.6	BWV 6.6	BWV 16.6	BWV 20.11	BWV 32.6	BWV 36.4	BWV 20.7	BWV 43.11	BWV 39.7	BWV 153.9
Base model 1	310	85	85	90	74	37	88	71	143	55
Base model 2	225	65	120	75	36	18	112	78	147	159
Base model 3	290	140	175	72	62	145	102	112	66	146
CCN	75	15	75	30	10	13	23	21	13	14

small. This means that the functional harmonizations obtained by the CNN are always very similar to the original J.S. Bach's harmonization.

Instead, each single base model deviates a lot from Bach's harmonization.

4 Conclusions

We conclude with a couple of observations and directions for future work. The first observation is the following: we used a binary representation because such a representation is compact and results in a very fast the training phase. It would be interesting to investigate the use of other representations basing them on similarities and differences between notes and chords and the resulting effect on the networks. It would be also interesting to investigate other configurations for the neural network, like for example the use of a tangential activation function.

We have used the network the learn the style of Bach. It would be interesting to use it to learn the style of other composers.

References

1. Website on Bach's chorales. A list with BWV numbers can be found here, http://www.jsbchorales.net, http://www.jsbchorales.net/bwv.shtml
2. Cope, D.: Experiments in Musical Intelligence, A-R edn. (1996)
3. De Prisco, R., Zaccagnino, R.: An Evolutionary Music Composer Algorithm for Bass Harmonization. In: Giacobini, M., et al. (eds.) EvoWorkshops 2008. LNCS, vol. 5484, pp. 567–572. Springer, Heidelberg (2009)
4. Ebcioglu, K.: An expert system for harmonizing four-part chorales. In: Machine models of music, pp. 385–401. MIT Press, Cambridge (1992)
5. Hild, H., Feulner, J., Menzel, W.: Harmonet, a neural net for harmonizing chorales in the style of J. S. Bach. In: Advances in Neural Information Processing Systems, vol. 4, pp. 267–274. Morgan Kaufmann, San Mateo (1992)
6. Lehmann, D.: Harmonizing melodies in real time: the connectionist approach. In: Proceedings of the International Computer Music Conference, Thessaloniki (1997)
7. Lehmann, D., Gang, D., Wagner, N.: Tuning neural network for harmonizing melodies in real-time. In: International Computer Music Conference, Ann-Arbor, Michigan (1998)
8. McIntyre, R.A.: Bach in a box: The evolution of four-part baroque harmony using a genetic algorithm. In: First IEEE Conference on Evolutionary Computation, pp. 852–857 (1994)
9. Miranda, E.R.: Composing Music with Computers. Focal Press (2001)
10. Phon-Amnuaisuk, S.: Composing Using Heterogeneous Cellular Automata. In: Giacobini, M., et al. (eds.) EvoWorkshops 2008. LNCS, vol. 5484, pp. 547–556. Springer, Heidelberg (2009)
11. Piston, W., DeVoto, M.: Harmony. W.W. Norton, New York (1987)
12. Schottstaedt, B.: Automatic species counterpoint. Tech Rep. STAN-M-19, Stanford University CCRMA. A short report appeared in Current Directions in Computer Music Research, Mathews, Pierce (eds.). MIT Press, Cambridge (1989)
13. Wiggins, G., Papadopoulos, G., Phon-Amnuaisuk, S., Tuson, A.: Evolutionary methods for musical composition. International Journal of Computing Anticipatory Systems (1999)

Combining Musical Constraints with Markov Transition Probabilities to Improve the Generation of Creative Musical Structures

Stephen Davismoon and John Eccles

Ian Tomlin Academy of Music, Edinburgh Napier University
s.davismoon@napier.ac.uk

Abstract. This paper explores a fundamental dilemma regularly faced by any composer engaged in computer-aided composition: the generation of musically 'meaningful' or 'sensible' musical structures beyond those of a very localized nature. Initially an outline of the uses of Markov processes in music is given, highlighting their strengths and weaknesses. There then follows a brief discussion describing some of the many constraints (cultural and otherwise) that have been and continue to be placed upon us all as we engage with music that emanates from relatively localized regions of the sonic continuum. Finally, through the combination of additional musical constraints with Markov transition probabilities within a stochastic optimization process, specific improvements in creating sensible musical structures are described.

1 Markov Chains and Music

"…the strength of evolutionary algorithms in art is their use as explorational and navigational tools in vast spaces of possible artistic material." (P. Dahlstedt [1])

The use of Markovian processes in music dates from the pioneering work of a few seminal figures during the 1950s. Much is owed by subsequent developments to the work of Lejaren Hiller, Leonard Isaacson - the fourth movement of their jointly composed *Illiac Suite* (1957) stands as the first composition in the repertoire to employ Markovian processes - their 1959 book "Experimental Music: Composition with an Electronic Computer" remains a classic in the field. While its pages in many ways discusses the Markovian relationships that exist between notes within what would normally be described as a tonal network, Iannis Xenakis' 1963 book *Formalised Music* [2] considerably augmented the creative potential for their use in post-tonal music. Since this time Markov Chains have become something of a mainstay technique in computer music; both for compositional creative work as well as for the extensive body of work that has employed Markovian principles in re-creative musicological research; where compositional models from the past have been 're-made' through their application to musical data. Charles Ames [3] and David Cope [4] are amongst those that have made significant contributions to the literature on this subject in recent decades.

C. Di Chio et al. (Eds.): EvoApplications 2010, Part II, LNCS 6025, pp. 361–370, 2010.

In summary the benefits of Markov Chains in Music are:

- They are very good at securing long-term structural coherence for the composer, by way of their ability to provide something akin to a generative grammar, this becomes especially the case when Markovian processes are used hierarchically: the 'local' motivic level informed by the textural level and *vice-versa*.
- They can aid the composer to secure long-term harmonic and textural coherence even if a highly complex language is adopted.
- Their utilization often allows for much more 'organic' resultant structures than those attained by other quasi-serial; textural/sonorist or open score approaches.
- They have also proven to be a powerful aid to musicologists in the reconstruction of historical compositional models.

However, quite apart from these strong deep-level structural benefits, there is the further positive outcome in the way that they regularly surprise the composer/listener by their output, thereby creating a developmental feedback loop; musically testing if you like the ears of the individuals that are listening to the results. This 'by-product' of their use should not be too readily overlooked; they are very powerful at allowing one to uncover unusual though related musical territories - the health of any art has always been dependent upon the discovery of new modes of expression.

However, it is very difficult to generate convincing original creative long-term musical structures with Markov Processes beyond that of a motive [5] - and it is often very difficult even to do this. And higher order Markov Processes become too specific and lose their creative potential. Furthermore they can significantly increase the amount of compositional re-working of the algorithm's output. From the composer's viewpoint it is impossible to generate structures that might be as developed as a musical phrase [6], let alone that of a theme group and/or its variants.

Of course, put rather simplistically, the root of these problems largely stems from the fact that the computer is not a musician and does not have cultural/aesthetic knowledge or memory. It is unaware as to how musically (un)fit or otherwise the output is from any given Markov matrix. Furthermore it is felt that this problem needs to be addressed as a necessary next step before a truly rich, meaningful, interactive/generative music is able to unfold in real-time. What is proposed is to constrain the unfolding of the Markov matrix in line with specific music meta-data. Even with these constraints we will still have an enormously rich and diverse search space available to the composer, but one that will guarantee improved levels of musical sense and will go to some lengths in addressing the 'fitness bottleneck' problem so often spoken of in the Evolutionary Music Literature…

"The past decade and more has shown that an EA has no difficulty in replicating a composer in 'hard-work'… But without the most stringently defined search space an unmanageably large amount of potential material, mostly unusable, is apt to be produced. Biles (1994) has described this situation as the fitness bottleneck." [7]

The need for the use of practice-based music constraints in computer-aided composition has also previously been discussed elsewhere by Davismoon in some detail [8].

2 Musical Constraints

"the perceptible variety of outputs from a given system is usually quite constrained compared with the mathematical space" (N. Collins [9])

Karlheinz Stockhausen in his seminal paper *How Time Passes* (1957) [10] draws attention to the many aspects of our natural human constraints with respect to how we discern differences in pitch, duration and tempo. These findings resulted from several years of intense sound research at the NWDR studios part-sponsored by the physicist Homer Dudley of Bell Telecommunications and would also certainly have been influenced by Stockhausen's exposure to the earliest researches of Information Theory as applied to sound by Werner Meyer-Eppler (whose ideas, arguably, also had a profound effect upon the development of Integral Serialism). This was followed by further in-depth research on a number of fronts relating to the fields of acoustics and psychoacoustics and as a result we now know much more about the natural constraints that act upon our hearing apparatus when we are exposed to sound. For example, Albert Bregman in his exhaustive work *Auditory Scene Analysis* [11] points out – amongst many things - the complex and discerning ability of human hearing to put together parts of a sound in order to make a intelligible composite whole - the comprehension of a conversation while in a crowded and noisy environment for example.

We also of course place physical constraints upon ourselves with respect to the language that we speak in order for it to be understood. The frequency constants in vowel production for example have been broadly researched. These constraints are also related to those found with respect to pitch in vocal music. In Western Classical music this tradition is fundamentally linked to the development of its system of clefs. In many respects the origins of orchestration grow from facets of choral tradition. It is only comparatively recently that pitches extensively beyond the human vocal range have been used in orchestral writing. As the Baroque era progresses, we witness an ever greater increase of 'instrumentally' idiosyncratic composition; exploring the innate qualities of the various instruments of the orchestra, in many respects this is evidenced by the development of the concerto form, thereby discovering a whole new set of creative constraints. The developments of compositional constraints in the context of the Western Classical tradition, in the sense of the laws governing harmony and counterpoint, developed at least as far back as Medieval times and have experienced several important epochs of experimentation, exploration and consolidation during the centuries that followed, often, though not always, emanating from the constraints found in the models of vocal music (i.e. voice leading, preparation and resolution of dissonances etc).

Irrespective of physical and instrumental constraints we also apply a whole set of what might be called Cultural Constraints when we engage with music. In many ways this is core to the philosophical question of what music is expressive of and by what means this is communicated. While this has been something of a central question to music aesthetics at least since Medieval times, a somewhat different analytic perspective was thrown on the subject with work that commenced during the 1980s, through the new disciplines of Music Semiotics and Topic Theory Analysis with the work of Jean-Jacques Nattiez [12], Francois-Bernard Mache [13], Kofi Agawu [14] and Raymond Monelle [15] allowing a glimpse into how music of various genres and

cultures can be explained by comparative cultural and/or structurally understood constraints. Analytic examples in the literature abound from works of musical High Modernism through to TV theme tunes, folk tunes from around the world, even to the songs of the animal kingdom, that would seem to operate their own palette of musical constraints. As a body of work it invites us to look beyond the purely musical aspect of the construction of a musical work in order to be able to see how it cuts across the cultural matrix.

Given the widespread evidence of multi-layered constraints acting upon our engagement with music, it would seem sensible to apply certain constraints that will emphasize musical sense in the output of a Markov algorithm.

3 Combining Musical Constraints and Transition Probabilities

Although Markov transition matrices encapsulate useful stylistic information, when used to generate musical lines directly, they often give musically unsatisfactory results. For example, Fig. 1. shows a typical musical phrase generated by Markov processes and illustrates a number of musical problems arising from their direct application.

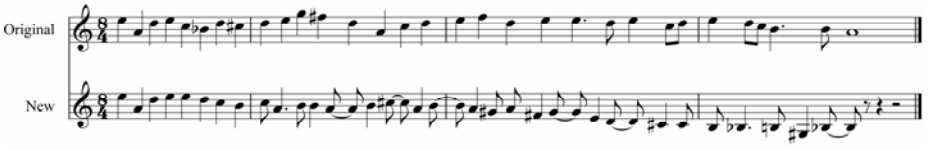


Fig. 1. A line of music generated by two Markov processes, one controlling the succession of melodic intervals the other note durations. The original line from which the transition probabilities were derived is shown in the top line (soprano line from *Ballo Quinto* in *Libro primo de balli, gagliarde, et correnti a quattro voci*, Kapsberger, 1615).

1. *Pitch Drift*: the range of notes not only drifts down but also goes out of range if the intended singer is a soprano (as in the original). Pitch drift can be either up or down and can often be more extreme than that illustrated. Even if the final notes end up in the desired range, mid-line excursions can still stray from required boundaries.
2. *Time Drift*: this manifests musically as excessive syncopation. This may or may not be a desirable result for the composer but it is clearly out of style with the original.
3. *Tonal Drift*: the tonal centre of the line has moved away from the opening A.
4. *Final Pitch*: the final note is effectively a random event and does not necessarily make musical sense.
5. *Final Note Duration*: as for the final pitch, the final note duration is again random and can often be at odds with the composer's intentions.
6. *End Time*: although the number of musical events is the same as in the original line (31) a random end time results. The phrase can often be considerably longer or shorter than the original and is particularly problematic for multi-part writing.

For convenience the first three problems are referred to collectively as *drift* and the last three as *end point* problems.

The overall problem is that the statistics do not encapsulate other musical information that is "obvious" to the composer. For example, the frequency of occurrence of crotchets and quavers tells us nothing about their location with respect to the beat; the likelihood of one interval following another does not directly give information about preferred tonal centres. It is possible to use other statistics (for example, the frequency of occurrence of quavers on the beat or absolute pitch distributions) but this leads to a more complex generation procedure.

It is the view of the authors that, in practice, the composer needs to control musical constraints such as phrase length, pitch range and degree of syncopation of individual lines in order to mesh with higher level musical goals of direction, style and form and meet necessary technical constraints as dictated by the given instrument and performer. The stylistic information encapsulated in the transition matrices is valuable only if it can be integrated with that relating to the musical meta-structure. Control of these parameters is thus as important as the desired statistics and various trade-offs are necessary if a greater musical sense is to be obtained.

3.1 Tackling Drift and the End Point Problems with Stochastic Optimization

One possible method to tackle the drift is to modify the transition matrices as the line is generated, taking into account notes already in the line. Measures of syncopation and pitch can be used to bias the generating matrices to return the line to the desired ranges. Musically, this can be pictured as an elastic tension that pulls out of range notes into bounds. The greater the tension (for example, the greater the degree of syncopation), the greater the likelihood of the next note being returned to the beat or desired pitch range. Mathematically, the probability vectors that make up the transition matrix can be viewed as points on the faces of a simplex which are moved towards more favourable edges or vertices by a simple linear transformation. Thus, if k ($0 \leq k \leq 1$) is some normalized measure of strain, $\mathbf{p}_i$ is a probability vector representing the i^{th} row of a transition matrix and $\mathbf{v}$ is a probability vector containing only favourable transitions (i.e., the non-zero components all correspond to outcomes which will bring the next note back into – or at least toward – the desired range), then on each step of the chain the actual $\{\mathbf{p}_i\}$ used are given by

$$\mathbf{p}_\mathbf{i}^* = \mathbf{p}_\mathbf{i} + k(\mathbf{v} - \mathbf{p}_\mathbf{i}) \tag{1}$$

This method has the advantage of being simple to compute and, because it modifies the underlying matrix, tends to preserve the desired stylistic musical properties. Although addressing the problems of drift, it is not so suited to tackling the end point problems.

To tackle the end point problem, the generating algorithm needs to be able to work with fixed (or at least probabilistically defined) final notes as well as fixed starting notes. Musically, it is also much more convenient to specify the duration of a phrase in beats and bars rather than the number of transitions in the Markov chain. The requirements to satisfy multiple, possibly conflicting, constraints led the authors to consider stochastic optimization methods rather than a Markov process to generate musical lines. The transitions matrices are then used as optimization objectives in

order to preserve the stylistic information but can be combined with additional functions representing other musical constraints.

3.2 Simulated Thermal Annealing

In order to evaluate the idea, a simple thermal annealing method based on the original work of Kirkpatrick[16], [17] was used. This was selected because of its relative simplicity and straightforward mapping of the musical problem into the basic annealing framework. The problem also seems well suited to more advanced methods of multi-objective optimization [18], however, for initial work, many useful insights have been gained from the simpler annealing implementation with a scalar objective function. In summary:

- The state of the system, *S,* is represented fully by a line of music. This comprises a set of notes each defined by a pitch and duration. The line is assumed to be monophonic (only one note sounds at a time) and continuous (no rests). The start time of each note is implicitly given by the succession of note durations.
- A scalar function, $E(S)$, can be defined which quantifies the optimization goals and the trade-offs between different objectives. This is described below.
- Reconfiguration rules can be defined which change the state of the system and repeated application of the rules allow all possible states of the system to be reached. In the case of a musical line, segments of the line are chosen at random and transposed or time shifted. Note durations are modified by splitting or merging randomly selected notes.
- An annealing schedule to control the successive application of the reconfiguration rules and selection of newly selected states can be defined. For the purposes of this work a simple exponential temperature decay was used.

3.2.1 Objective Function

For a standard annealing process, the objective function, $E(S)$, must give a single scalar value of how well a musical line matches the desired properties. The value of E must always be zero or positive with smaller values indicating a better match. Since multiple musical properties of a line are to be considered, a weighted sum, $\Sigma w_i E_i$, of the various properties is used. In the work described here, four components are considered; note interval transition probabilities, note duration transition probabilities, syncopation and pitch range.

For comparison of the transition matrices, a simple Euclidian distance of the composite probability vectors is used

$$E_{\mathbf{P}} = E(\mathbf{P}, \tilde{\mathbf{P}}) = \frac{1}{N} \sum_{i=1}^{N} \sum_{j=1}^{N} (p_{ij} - \tilde{p}_{ij})^2 \tag{2}$$

where $\mathbf{P}$ is the NxN transition matrix of a given property for a line and $\tilde{\mathbf{P}}$ is the desired matrix (i.e., the matrix chosen to generate a line in a simple generative system). A single normalizing factor of $1/N$ is used because the matrix rows are probability vectors and thus already normalized so that the components sum to one. Significantly different matrices are thus expected to give values of $E_{\mathbf{P}}$ around unity with close matches approaching zero.

The syncopation measure is derived by considering whether or not the start of each note is aligned with an appropriate subdivision of the bar. For example, crotchets are expected to start in the beat and quavers either on the beat or one quaver off the beat. Notes which start at the expected time make no contribution to the objective function, a single syncopated note contributes $1/M$ (where M is the number of notes in the line) and longer sequences of m syncopated notes contribute $(2^m-1)/M$. The intention of this function is to allow occasional syncopated notes or short sequences but longer sections of syncopation rapidly become very costly.

The pitch range is handled as penalty function since out of range notes are not acceptable in the final output. Any note outwith the specified range (for the *Ballo Quinto* C4-C6) is assigned a value of one, all other notes zero. The contribution for a line is not normalized by line length. This is to ensure that even a single note out of range will not in general be accepted.

The relative weights of the four components of the objective function are set by the composer. The components have been defined such that a weight of unity is generally a good starting point for experiment but will of course vary according to the material and composer's intentions. In the work described here weight values of {1, 0.2, 1, 10} for the interval, duration, syncopation and range respectively were used.

3.2.2 Reconfigurations Rules

Four reconfiguration rules are defined, two modify pitch and two modify duration. On each iteration, one rule is selected at random with equal likelihood and applied to the current line.

The pith rules select a random segment of the line and either transpose all of the notes in the segment by a randomly selected interval or move the segment to another randomly selected position in the line. Line length and note durations are preserved. Segment lengths are chosen with equal likelihood from the range [1, $N/2$] where N is the number of notes in the line. The transposition interval (in semitones) is selected uniformly from [-12, 12], line segments may thus be transposed up or down by up to an octave.

The duration reconfiguration rules select a note at random and either split it into two notes of smaller duration with the same pitch or merge it with the following note to create a single longer note with the same pitch. Simple note values (crotchet, minim, semibreve) are halved; dotted note values are split in a 2:1 ratio.

In all cases, the first and last notes in the line are excluded from the process in order to honour the end point constraints. The duration merge rule excludes the last two notes from its selection process so as not to affect the final note.

3.2.3 Annealing Schedule

The basic annealing method of Kirkpatrick is used. After each application of the reconfiguration rule the newly created line is accepted with probability

$$P = \begin{cases} 1 & \Delta E_i < 0 \\ \exp(-\Delta E_i / kT) & \Delta E_i \geq 0 \end{cases} \tag{3}$$

where $\Delta E_i = E_i - E_{i\text{-}1}$ is the change of energy on the i^{th} iteration. Thus if the newly configured line has a lower energy than the previous one it is always accepted; if it has a higher energy it is accepted with decreasing likelihood as ΔE_i increases.

As the annealing proceeds, the system is slowly "cooled" by reducing the effective temperature kT and lines with positive ΔE are accepted less and less frequently. A simple exponential decay is used for kT. After each annealing epoch kT is reduced by a constant ratio, $kT := \alpha kT$. All work described here used an epoch of 10,000 iterations and values of $\alpha = 0.98$. Because of the normalization of the energy function values, it was possible to use an initial value of kT (kT_0) of around unity.

The initial line is generated by filling the line between the fixed (composer specified) end points with crotchets of random pitch. Pitches are selected uniformly from the range used in the objective function.

3.3 Annealing Results

The annealing process was implemented within an existing experimental software framework that supported the import and export of MIDI files and the generation of Markov transition matrices and instances of corresponding chains. For the initial investigation, only pitch and rhythm drift were considered, not tonal drift. All three end point problems are addressed by enforcing the fixed duration of the line within the reconfiguration rules and by not allowing the first and last notes to be changed within the relaxation process. Fig. 2. illustrates a typical output of the process using the same transition probabilities for note interval and duration within the objective function as used in the original example of Fig. 1.

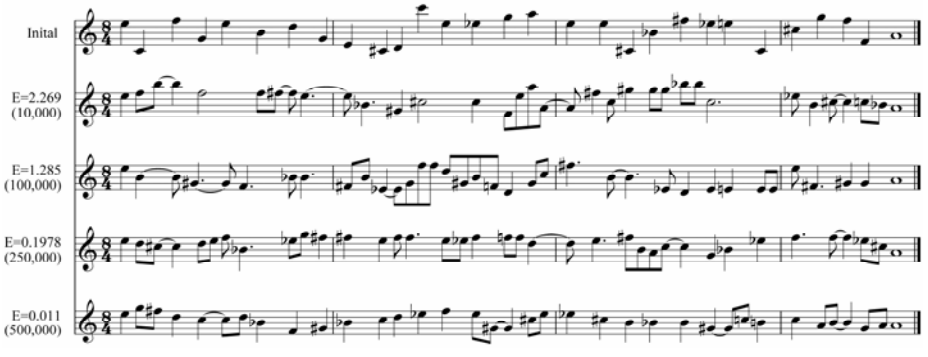


Fig. 2. A typical output from the annealing process. The initial random line, 3 intermediate lines and the final result after 500,000 iterations are shown. The energy, E, and number of iterations are given at the start of each line. An initial value of kT_0=1.0 was used and γ=0.98. The first and last notes and interval and duration statistics are from the original *Ballo Quinto* shown in Fig. 1. The pitch range was set to C4 – C6.

Musically, the results are considered to be much more satisfactory than the simple Markov process illustrated earlier, primarily because the duration and final note can be fixed by the composer. The musical material within the line is, as expected, of a similar style to the unmodified Markov process. Some tonal drift can be seen within the line (for example, the Bb-C-D-Eb in bar 2) but this should not be surprising given that no explicit pitch information (other than extreme range) was incorporated in the objective

function. It is possible that the fixed first and last notes exert a degree of tonal influence in the vicinity of the end points but this is an area for further investigation.

4 Conclusions

Assessing the musical fitness of a line by how well it meets multiple, simple objectives and constraints is considered to be the key to the process. It neatly sidesteps the fitness bottleneck problem by removing the need to directly quantify the overall musicality of the line and the evaluation of the lower level objectives and constraints (for example, the interval distributions and measures of syncopation) is a task well suited to computer. Because the drift and end-point problems are also addressed, the outputs of the annealing process are much more useable than those generated directly by Markov chains. Furthermore, the stochastic nature of the optimization allows for occasional notes and intervals to be used which would never have been generated directly by a Markov chain because of zero probabilities in the transition matrix. This is seen very much as a creative, rather than regenerative, process.

For any given set of objectives and constraints, there are clearly many near-optimal solutions. In the simple annealing process described here, the composer is required to navigate the "weight space" of the single objective function by running the process a number of times. When the quality of the optimization outputs is high, this can be a creative task, the composer exploring the near-optimal solution space by experimenting with the different weights. With this perspective, however, the problem appears well suited to the application of multiple-objective optimization techniques and, with suitable formulated constraints, the most musical solutions are anticipated to lie on the Pareto-optimal front. Future work will consider both an evaluation of multiple-objective optimization methods and extending the types of musical objectives used, including those encountered through instrumental and vocal performance technique; to original complex polyphonic textures, none of which are currently suited through direct generation by Markov chains.

The results of this methodology in some senses add an interesting echo to the serial vision of Schoenberg. The composer can now pinpoint pitch and rhythmic compositional End Points (*alla* Deterministic Serialism), however the space in-between each step is open to probabilistic relativistic determination. Thus allowing a whole new universe of expressivity to remain open for exploration.

References

1. Dahlstedt, P.: Thoughts on Creative Evolution: A Meta-generative Approach to Composition. Contemporary Music Review 28(1), 48 (2009)
2. Xenakis, I.: Formalized Music: Thought and Mathematics in Music, Pendragon revised edn. (1992)
3. Ames, C.: The Markov Process as a Compositional Model: A Survey and Tutorial. Leonardo 22(2), 175–187 (1989)
4. Cope, D.: The Algorithmic Composer. A-R Editions, Inc. (2000)
5. Hiller, L.A., Isaacson, L.M.: Experimental Music: Composition with an Electronic Computer, p. 15. McGraw-Hill, New York (1959); Reprinted Greenwood Press (1979)

6. Schoenberg, A.: Fundamentals of Musical Composition. In: Strang, F., Stein, L. (eds.), pp. 3–9. Faber and Faber (1967)
7. Husbands, P., et al.: Evolutionary Computer Music. In: Miranda, E.R., Biles, J.A. (eds.), vol. 15. Springer, Heidelberg (2007)
8. Davismoon, S.: For the Future. Contemporary Music Review 28(2), 237–243
9. Collins, N.: Musical Form and Algorithmic Composition. Contemporary Music Review 28(1), 106 (2009)
10. Stockhausen, K.: How Time Passes. Die Reihe, vol. 3 (English Translation Cardew, C) (1957/1959)
11. Bregman, A.: Auditory Scene Analysis: The Perceptual Organization of Sound, New edn. MIT Press, Cambridge (1994)
12. Nattiez, J.J.: Music and Discourse: Toward a Semiology of Music. Princeton University Press, Princeton (1990)
13. Mache, F.-B.: Music Myth and Nature. Harwood Academic Publishers (1993)
14. Agawu, K.: Playing with Signs: Semiotic Interpretation of Classical Music. Princeton University Press, Princeton (1992)
15. Monelle, R.: Linguistics and Semiotics in Music. Routledge, London (1992)
16. Kirkpatrick, S., Gellat, C.D., Vecchi, M.P.: Optimization by Simulated Annealing. Science 220(4598), 671–680 (1983)
17. Kirkpatrick, S.: Optimization by Simulated Annealing: Quantitative Studies. Journal of Statistical Physics 34(5-6), 975–986 (1984)
18. Deb, K.: Multi-Objective Optimization using Evolutionary Algorithms. John Wiley, Chichester (2001)

Dynamic Musical Orchestration Using Genetic Algorithms and a Spectro-Temporal Description of Musical Instruments

Philippe Esling, Grégoire Carpentier, and Carlos Agon

Institut de Recherche et Coordination Acoustique / Musique
1 place Igor Stravinsky, 75004 Paris, France
{philippe.esling,gregoire.carpentier,carlos.agon}@ircam.fr

Abstract. In this paper a computational approach of musical orchestration is presented. We consider orchestration as the search of relevant sound combinations within large instruments sample databases. The working environment is *Orchidée* an evolutionary orchestration algorithm that allows a constrained multiobjective search towards a target timbre, in which several perceptual dimensions are jointly optimized. Up until now, *Orchidée* was bounded to "time-blind" features, by the use of averaged descriptors over the whole spectrum. We introduce a new instrumental model based on Gaussian Mixture Models (GMM) which allows to represent the complete spectro-temporal structure. We then present the results of the integration of our model and improvement that it brings to the existing system.

Keywords: Orchestration, Genetic Algorithms, Gaussian Mixture Models, Instruments Temporal Evolution, Instrumental Models.

1 Introduction

Automatic orchestration is a relatively new topic in computer-aided composition, for which only a few number of systems exist today (for a complete review see [3]). An orchestration problem consists in finding a mixture of instrument samples that jointly minimizes different perceptual distances to a given *target*. *Orchidée* – the most recent of them, developed at IRCAM[1] – offers significant innovative features. First, the system architecture has been designed to fully separate instrumental knowledge from the search algorithms, allowing *Orchidée* to work with virtually any kind of timbre modeling. Secondly, a client/server architecture is put forward, providing the users with the opportunity of implementing their own control interfaces while keeping the computational issues in the core program. Last, *Orchidée* explicitly addresses the timbre multidimensionality and the combinatorial complexity of instrument mixtures by the joint use of a multicriterion optimization and evolutionary algorithms [2]. *Orchidée* has already

[1] Institut de recherche et coordination acoustique/musique – http://www.ircam.fr

C. Di Chio et al. (Eds.): EvoApplications 2010, Part II, LNCS 6025, pp. 371–380, 2010.

been used by many notable composers, among them Jonathan Harvey for his latest piece *Speakings* [8].

However, like all previous systems, using "time-blind" features only, *Orchidée* was therefore unable to cope with time-evolving sounds and could only address static timbres. In this paper, we extend this approach with an instrumental model based on Gaussian mixtures to represent both temporal and spectral properties. This model allows to capture the relevant properties of instrumental timbres with great data reduction while allowing for efficient resynthesis of them. Moreover, the strength of this model is to infer almost automatically the temporal structure of a sound with modified duration while keeping the attack and release.

The reminder of this paper is organized as follows. In Section 2 we present the system architecture and the genetic exploration algorithm. In Section 3 we introduce the instrumental model that accounts for the temporal structure, we then present the results of our model in terms of accuracy. We subsequently present the experimental results from the integration of our work in the orchestration system *Orchidée,* noting the improvement over the "static" orchestration results.

2 System Architecture

The core element of *Orchidée* is a background-running server that communicates with client interfaces. The server consists of a "long-term" part which is a database-like organized description of a collection of instrument sound samples, and a "short-term" part whose elements relate to the current orchestration task. The problem is therefore fully defined by three components: An orchestra (i.e. a set of instruments), a set of musical-related constraints and a description of the target timbre (usually provided as either a concrete or synthesized sound). All the above elements may be easily defined by the user through the client interface. Afterwards, the server instantiate a problem-specific object, in which data structures are optimized for efficient numerical computation. The problem is then properly stated to be solved by a multiobjective genetic algorithm, whose core elements are implemented in a dedicated toolbox. Until recently, the optimization process could be run on up to six different timbre objectives : the spectral centroid, the spectral spread, the log-attack time, the main resolved partials, the mel spectrum and the amplitude modulation. All these features are time-averaged values preliminary extracted from a collection of instrument sound samples.

Once the orchestration problem is fully stated, the optimization process runs as depicted in Fig. 1. First, the orchestra is mapped onto a discrete, m-dimensional space E (where m is the size of the orchestra), in such a way that all genetic operators can be expressed as closed operations on E. From there, an initial population is randomly drawn in E. Additionally, the target and constraints are converted into objective and penalty functions respectively, which are both used for fitness computation. Then, the mating pool is filled with a binary tournament procedure and classical uniform crossover and 1-point random mutation apply. Offspring systematically enter the current population from which individuals in

denser regions are removed during a diversity preservation procedure. The algorithm iterates until a maximum number of iterations is reached, and the Pareto archive is returned to the user as the set of solutions. From there, the algorithm may be redrawn from a specific Pareto solution for search intensification in the neighborhood. Now, half of the initial population are clones of the selected solution and the objective functions are modified to focus the search in the desired region of the criteria space.

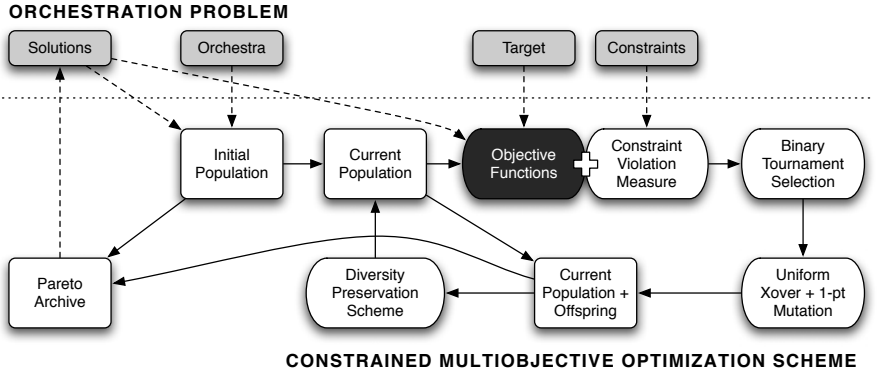


Fig. 1. Genetic Orchestration algorithm flowchart

We introduce in this paper, a new feature to describe temporal aspects of sounds. We derive an objective function that reflects the distance (on a temporal basis) between a target and an orchestration proposal. This function is inserted as a new objective function for the evolutionary algorithm (black box in Fig. 1).

3 Temporal Descriptor

All previous works on this subject have focused on *vertical* orchestration, analyzing only the sustained part of instruments and thus completely disca-rding the temporal evolution of sounds. However, the territory of timbre is not confined to a static structure of proportions. It rather comprises "variation laws" that regulate the interaction between frequency and amplitude functions in an evolving context over time. It is therefore essential to move to a higher level of modeling, by understanding the *micro-temporal* qualities of timbre in order to capture the sound as a spectro-temporal structure. Advantages of this approach are twofold. First, the generated orchestration may be considered more realistic as it accounts for the whole spectro-temporal structure. Second, it allows the use of evolving playing modes like crescendo, glissando, multiphonics and so on.

3.1 Existing Models

When looking at existing instrumental models, we can quickly realize the almost complete absence of time modeling. Instrumental models are usually designed for

discrimination or classification systems [9]. For such purposes, instruments are usually represented in a simplistic manner with a set of spectral descriptors. The instrumental model of Tardieu [11] developed for orchestration, completely omits the temporal information. Sounds are thus considered purely stationary. In order to model this information, we can imagine the simplistic ADSR model applied to each partial. However, despite an advanced parameterization [5], performances in terms of accuracy are very limited. An improvement of this technique is to use Legendre polynomials which can increase the overall accuracy. This technique will serve as a benchmark for our model. Another method for time modeling is to compute the modulation spectrum [10]. Unfortunately it causes a significant increase of data amount which is inconsistent with the purpose of our model. Other approaches may be envisaged, such as a source-filter model [13] which would preserve the temporal information, while reducing the dimensionality of the data. However, this limits the modeling ability to a few production types. It would thus be beneficial to use representation techniques such as the Harmonic Temporal Clustering model [6], which can represent the instrumental information both harmonically and temporally using a mixture of gaussians. We use this representation as a baseline for our model.

3.2 Our Model

When establishing a model, there is always a tradeoff between accuracy and data reduction. Yet, facing this uncertainty problem, we can use perceptual studies to guide our discrimination of unnecessary features. A particularly relevant study in the context of instrumental models was conducted by McAdams and Beauchamp [7] which deals with the discrimination of instrumental sounds with simplified spectro-temporal parameters. This study gives insight on the perceptual importance of various spectro-temporal characteristics, and which can thus be simplified. Firstly, two simplifications seem extremely discriminating. The amplitude envelope coherence and the smoothing of spectral envelope indicate that the independent evolution of the partials amplitude is a fundamental feature. These results show that computing average magnitudes for each partial is largely insufficient. It is therefore crucial to represent the envelope independently for each partial. In contrast, the microvariations simplification in both amplitude and frequency appear to be much less discriminating perceptually. It is therefore possible to smooth (up to a certain level) the various envelopes while retaining the appropriate information.

In order to smooth the frequency microvariations, we use a constant-Q transform [1], which has a logarithmic frequency scale. This transform can be seen as a series of logarithmically spaced filters. By choosing wisely the center frequencies of the filters, the constant-Q transform shows the advantage of frequency bins directly corresponding to musical notes. Moreover, the dimensionality of constant-Q transformed data is much lower, which will allow faster computation of the next steps. We apply this transform with a resolution of 96 frequency bins per octave which yields bins corresponding to one-sixteenth of tone.

To represent each frequency bin, we use Gaussian Mixtures Model (GMM) [12]. A frequency bin is therefore defined as a weighted sum of Gaussian distributions

$$B_{x;\theta} = \sum_{c=1}^{C} \alpha_c \mathcal{N}\left(x; \mu_{\mathbf{c}}, \sigma_c\right) \tag{1}$$

with C the number of components, μ_c the mean, σ_c the variance and α_c the weight of component c, $0 < \alpha_c < 1$ for all components and $\sum_{c=1}^{C} \alpha_c = 1$. The parameters list

$$\theta = \{\alpha_1, \mu_1, \sigma_1, \ldots, \alpha_C, \mu_C, \sigma_C\} \tag{2}$$

defines a particular Gaussian mixture model. The problem is thus to find the set of parameters $\hat{\theta}$ which maximizes the likelihood $\mathcal{L}(X;\theta)$ to observe the data

$$\hat{\theta} = \underset{\theta}{argmax} \ \left(ln \ \mathcal{L}(X;\theta)\right) \tag{3}$$

In order to estimate the model parameters, we use the Expectation Maximization algorithm (EM). The application of this algorithm for Gaussian mixtures [4] runs as follows. The known data X are interpreted as incomplete data. The missing part Y is to know which component produce each sample x_n. For each x_n exists a binary vector $y_n = \{y_{n,1}, \ldots, y_{n,C}\}$, with $y_{n,c} = 1$ if the sample is produced by component c, or zero otherwise. The expectation step is to calculate the conditional expectation of the log-likelihood of all data. The probability can be calculated using Bayes' law

$$w_{n,c} = \frac{\alpha_c^i p\left(x_n \mid c; \theta^i\right)}{\sum_{j=1}^{C} \alpha_j^i p\left(x_n \mid j; \theta^i\right)} \tag{4}$$

where α_c^i is the prior probability (of the estimate θ^i) and $w_{n,c}$ is the posterior probability that x_n is produced by the component c. The particularity of our model is that the means of Gaussian functions are spaced by a distance proportional to the diffusion parameter σ_k which is also the common standard deviation of all distributions. We obtain the following mixture

$$B_{x;\theta} = \sum_{c=1}^{C} \alpha_c \mathcal{N}\left(x; \mu_{\mathbf{k}} + c.\sigma_k, \sigma_k\right) \tag{5}$$

It is necessary to maintain only the mean of the first Gaussian μ_k and the common standard deviation σ_k. By applying the maximization step for estimating the parameters distribution for a mixture of C components, we obtain the update formulas

$$\alpha_c^i = \frac{1}{N} \sum_{n=1}^{N} w_{n,c} \tag{6}$$

$$\mu_k^i = \frac{\sum_{c=1}^{C} \sum_{n=1}^{N} \left(x_n - c.\sigma_k^{i-1}\right) w_{n,c}}{\sum_{c=1}^{C} \sum_{n=1}^{N} w_{n,c}} \tag{7}$$

$$\sigma_k^i = \frac{1}{2}\left(\left(\left(a_k^i\right)^2 + 4b_k^i\right)^{1/2} - a_k^i\right) \tag{8}$$

$$\begin{cases} a_k^i = & \sum_{c=1}^{C}\sum_{n=1}^{N} c\left(x_n - \mu_k^i\right) w_{n,c} \\ b_k^i = & \sum_{c=1}^{C}\sum_{n=1}^{N} \left(x_n - \mu_k^i\right)^2 w_{n,c} \end{cases} \tag{9}$$

The new estimates are compiled in θ^i. Equations are evaluated again with the new estimates until the convergence criterion is satisfied.

3.3 Length Modification

Most of the orchestral instruments producing sustained sounds, it seems mandatory that the comparison between the target and the instrumental mixture is made on an identical period. Otherwise, the similarity index would inevitably be distorted by such differences in duration. Using the parameters obtained from the model computation, it becomes effortless to infer the structure of an identical instrumental sound but of different duration. We consider modification by expansion or compression of the temporal envelope on the sustained part of the sounds only. Thus attack and release segments are kept unmodified by calculating their respective locations and then applying changes only to parameters belonging to the sustained segment. This choice stems from the fact that excitation over the attack and release segments are generally the same regardless of the note duration. The modified Gaussian are then those whose mean satisfies

$$pos_{att} + \epsilon < \mu_k < pos_{rel} - \epsilon \tag{10}$$

with pos_{att} the end of the attack segment, pos_{rel} the beginning of the release segment and ϵ a time tolerance factor. We thus obtain the set of Gaussians to be modified

$$k \in K = \left\{\mathcal{N}\left(\mu_k, \sigma\right) / pos_{att} + \epsilon < \mu_k < pos_{rel} - \epsilon\right\} \tag{11}$$

Two different methods may then be applied in order to modify accurately the duration of the structure.

Modification by dilation. In our model, all Gaussians are linked together by a common diffusion parameter σ_k. This parameter can be used in a straightforward manner to perform expansion or compression on the gaussian mixture. If we want to change the length of the sound by a factor γ, we just have to multiply the variance of each gaussian by this factor thus $\sigma_k^{'} = \gamma\sigma_k$. We must then update the weights of the modified Gaussians accordingly in order to compensate for the modification of variance $\alpha_k^{'} = 2.\gamma.\alpha_k$ and then normalize all weights. The means of the gaussians being linked by a common diffusion factor, it is sufficient to calculate the new means for $n \geq min(k)$

$$\mu_n = \mu_{n-1} + \left(\sigma_{n-1}^{'} + \sigma_n^{'}\right) \tag{12}$$

Modification by repetition. It is possible for special cases such as vibrato and tremolo to perform a more pertinent modification by repetition. This is done by keeping the original parameters but changing the number of gaussians and then updating the weights of the corresponding components. We must first calculate the new number of components

$$N_{new} = \lceil \gamma . N_{k \in K} \rceil + N_{g \notin K} \tag{13}$$

The parameters of pre-existing components remain unchanged and we consider that the weights of the newly added components follow a cyclical behavior

$$\alpha_n = \alpha_{n-card(K)} \tag{14}$$

It is necessary to note that both approaches are applicable to specific cases. Indeed, if dealing with a linearly decreasing envelope, modification by repetition appears irrelevant. Conversely, for tremolo or vibrato modes, even if modification by dilation seems to infer a valid structure, it is obvious that the repetition approach is much more relevant. Thus, it would be optimal to choose the method of duration modification by following a taxonomy of the samples being analyzed.

3.4 Results

We discuss here the results of our model. For this we compare it with Legendre polynomials which have already been applied to harmonics modeling [5] and GMM with free variance. It is possible to observe their performance from different angles. Computation time is not a primary criterion as the analysis is done once and then stored in a database. However, we emphasize the importance of model accuracy by calculating the divergence between the original spectrum and a spectrum resynthesized from reduced data. This precision is confronted with the data reduction induced by the model. This allows to evaluate the performance of the model to capture the essential characteristics of a sound. The accuracy measurement is performed using two different error measures, the standard L^2 and the generalized *Kullback-Leibler* divergence. We compute these two distance measures between the original spectrum and the resynthesized spectrum. However, as duration and content of the samples may be very different,

Table 1. Mean value and variance for synthesis time, reduction factor and accuracy of the different models

	Legendre polynomials	Gaussian Mixture Models	
		Free variance	Fixed variance
Synthesis time (s)	**0.0031** (+/– 0.0037)	0.0212 (+/– 0.0032)	0.0250 (+/– 0.0138)
L^2	39.749 (+/– 836.91)	19.429 (+/– 218.31)	**8.3071** (+/– 37.341)
$L^2/Silence$	0.5690 (+/– 0.1529)	0.2143 (+/– 0.0267)	**0.0897** (+/– 0.0223)
KL	335.79 (+/– 6987.8)	272.82 (+/– 458.97)	**201.72** (+/– 362.76)
$KL/Silence$	0.9650 (+/– 0.5047)	0.6176 (+/– 0.0043)	**0.3811** (+/– 0.0154)
Reduction factor	**77.236** (+/– 498.75)	52.202 (+/– 339.28)	48.687 (+/– 246.69)

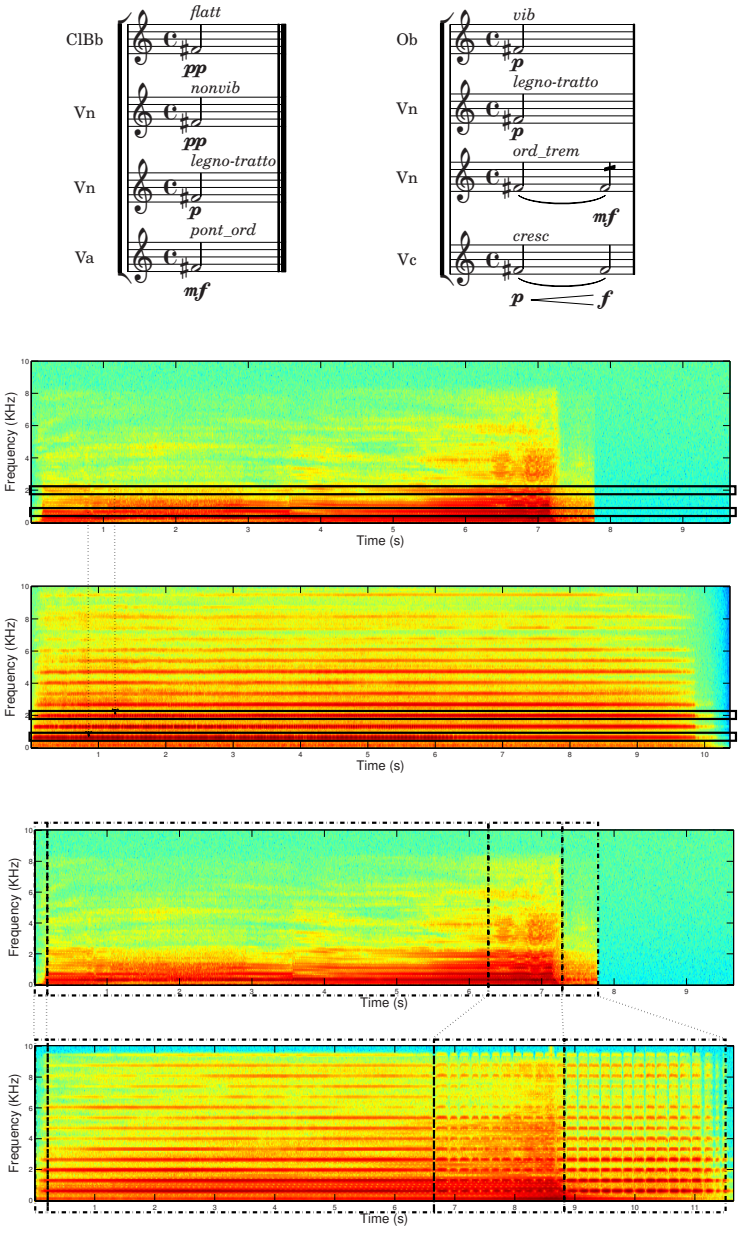


Fig. 2. Comparison of the results from static and dynamic orchestrations on a time-evolving target (extract from *Poetry* - Marco Suarez). When using only static descriptors (left score and top spectrums), although the average harmonic structure is preserved, the temporal information is completely omitted. After integration of the temporal descriptor however, we see very clearly on the results that the overall temporal structure of the target is met.

we normalize these measures with the divergence between each sample and an equivalent period of silence. In this way we can compare results with sounds of various nature. Table 1 summarizes the results of analysis on all 21,381 samples of the database.

As hoped, we obtain the best results with the fixed-variance model which allows to obtain greater accuracy with an almost equivalent data reduction. The accuracy of the free variance model could be increased by allowing a higher number of components but that would proportionally increase the chances of obtaining singular estimates.

4 Experiments

By introducing the temporal model, we allow the orchestration research to use not purely stationnary sounds. However, the system being programmed around this central hypothesis, several structural modifications have been performed. First, all descriptors were expected to have the same size, which is irrelevant for efficient temporal modeling. Moreover, the descriptor system has been extended to use matrix representations. Finally, the computation of descriptors is now done with respect to the target in order to use duration modification as stated above. We describe here the results of the experiment with system *Orchidée* after integration of our temporal model. The target sound is a voice extract from *Poetry* of Marco Suarez. That voice produce a note consisting of 3 distinct parts. First, a noisy component and then an amplification of the harmonic intensity that ends with a tremolo. As shown in Fig.2, when using only static descriptors, although the average harmonic structure is preserved, the temporal information is completely omitted. After integration of the temporal descriptor however, we see very clearly on the results the different parts of the original sound. The overall temporal structure of the target is met, we thus confirm the validity of our model and the improvement that it brings to the existing system.

5 Conclusion and Future Work

In this paper, we have extended the orchestration system *Orchidée* in order to cope with evolving sounds and use the temporal structure of timbre. We have presented an instrumental model based on GMM that allows to account for the temporal evolution of sounds in the orchestration process.

Several directions of future work can be seen on the temporal model. The type of distributions can be modified in order to increase the overall accuracy of the model and of the duration modification. The mixture approximation should also be done according to a peak tracking rather than being applied to frequency bins in order to improve data reduction. The similarity measure between an instrumental mixture and the target should also be enhanced in order to cope with the complexity of the spectro-temporal structure. Finally, the genetic algorithm should be extended to account for temporal combinatorics. It is up to now impossible to induce a temporal shift between the instruments which are supposed

to start at the same time. But this suppose not only choosing a starting point for each sound but also how do we scale sounds. This means the insertion of two additionnal heuristic parameters in the genome of solutions.

References

1. Brown, J.C., Puckette, M.S.: An Efficient Algorithm for the Calculation of a Constant Q Transform. Journal of the Acoustic Society of America 92(5), 2698–2701 (1992)
2. Carpentier, G., Assayag, G., Saint-James, E.: Solving the Musical Orchestration Problem using Multiobjective Constrained Optimization with a Genetic Local Search Approach. Heuristics (2009) (in Print)
3. Carpentier, G., Bresson, J.: Interacting with Symbolic, Sound and Feature Spaces in Orchidée, a Computer-Aided Orchestration Environment. Computer Music Journal 34(1) (2010)
4. Figueiredo, M.A.T., Jain, A.K.: Unsupervised Learning of Finite Mixture Models. IEEE Transactions on Pattern Analysis and Machine Intelligence 24(3), 381–396 (2002)
5. Jensen, K.: Musical Instruments Parametric Evolution. In: Proceedings of the ISMA, Mexico City, Mexico (2002)
6. Kameoka, H., Nishimoto, T., Sagayama, S.: A Multipitch Analyzer Based on Harmonic Temporal Structured Clustering. IEEE Transactions on Audio, Speech and language processing 15(3) (2007)
7. McAdams, S., Beauchamp, J.W., Meneguzzi, S.: Discrimination of Musical Instrument Sounds Resynthesized with Simplified Spectrotemporal Parameters. Journal of the Acoustical Society of America 105(2), 882–897 (1999)
8. Nouno, G., Cont, A., Carpentier, G., Harvey, J.: Making an Orchestra Speak. In: Proceedings of the Sound and Music Computing Conference, Porto, Portugal, pp. 277–282 (2009)
9. Peeters, G.: Automatic Classification of Large Musical Instrument Databases using Hierachical Classifiers with Inertia Ratio Maximization. In: Proceedings of the 115th Audio Engineering Society Convention, New York, USA (2003)
10. Sukittanon, S., Atlas, L.E., Pitton, J.W.: Modulation-Scale Analysis for Content Identification. IEEE Transactions on Signal Processing 52(10), 3023–3035 (2004)
11. Tardieu, D., Peeters, G., Rodet, X.: An Instrument Timbre Model for Computer Aided Orchestration. In: IEEE Workshop on Applications of Signal Processing to Audio and Acoustics, New Paltz, New York (2007)
12. Verbeek, J.J., Vlassis, N., Kröse, B.: Efficient Greedy Learning of Gaussian Mixture Model. Neural Computation 5(2), 469–485 (2003)
13. Virtanen, T., Klapuri, A.: Analysis of Polyphonic Audio Using Source-Filter Model and Non-Negative Matrix Factorization. In: Advances in Models for Acoustic Processing, Neural Information Processing Systems Workshop (2006)

Evolutionary Sound Synthesis: Rendering Spectrograms from Cellular Automata Histograms

Jaime Serquera and Eduardo R. Miranda

ICCMR - Interdisciplinary Centre for Computer Music Research
University of Plymouth, UK
{jaime.serquera,eduardo.miranda}@plymouth.ac.uk

Abstract. In this paper we report on the synthesis of sounds using cellular automata, specifically the multitype voter model. The mapping process adopted is based on digital signal processing analysis of automata evolutions and consists in mapping histograms onto spectrograms. The main problem of cellular automata is the difficulty of control and, consequently, sound synthesis methods based on these computational models normally present a high factor of randomness in the output. We have achieved a significant degree of control as to predict the type of sounds that we can obtain. We are able to develop a flexible sound design process with emphasis on the possibility of controlling over time the spectrum complexity.

Keywords: Sound Synthesis, Cellular Automata, Histogram Mapping Synthesis, Additive Synthesis, Multitype Voter Model.

1 Introduction

A number of musicians, in particular composers, have started to turn to evolutionary computing for inspiration and working methodology. This paper focuses on cellular automata (CA), a class of evolutionary algorithms widely used to model systems that change some feature with time. They are suitable for modelling dynamic systems in which space and time are discrete, and quantities take on a finite set of discrete values. CA are highly suitable for modelling sound and music, which both are fundamentally time-based and can be thought of as systems in which a finite set of discrete values (e.g., amplitudes, frequencies, musical notes, rhythms, etc.) evolve in time [10].

In the 1950s several different kinds of systems equivalent to CA were independently introduced. The best-known way in which CA were introduced (and which eventually led to their name) was through work by John von Neumann in trying to develop an abstract model of self-reproduction in biology –a topic which had emerged from investigations into Cybernetics [13].

Cellular automata are normally implemented as a regular grid of cells in one or more dimensions. Each cell may assume any state from a finite set of n values. CA evolve in successive generations at every time unit. For each generation, the values of all cells change simultaneously according to a set of transition rules that takes into account the states of the neighbouring cells. The transition rules can be deterministic

C. Di Chio et al. (Eds.): EvoApplications 2010, Part II, LNCS 6025, pp. 381–390, 2010.

or non-deterministic. With a deterministic rule, for a given configuration of cell states, the updated cell state is always the same. With non-deterministic rules the next state is not only dependent on the neighbourhood but also on some random inputs and/or probabilistic components. A probabilistic rule gives the probabilities that each cell will transition to the next possible state. The states of the cells may represent different colours and therefore, the functioning of a two-dimensional cellular automaton may be displayed on the computer screen as a sequence of images, like an animated film.

Cellular automata have been of interest to computer musicians because of their emergent structures –patterns not created by a single rule but through the interaction of multiple units with relatively simple rules. This dynamic process leading to some order allows the musician to explore new forms of organization. In sound synthesis, CA are normally used for controlling over time the parameters of a synthesis instrument. Many of the synthesis techniques demand enormous amounts of control data for obtaining interesting results, making it difficult to be controlled manually. CA represent a solution to this problem because with few parameter specifications it is obtained massive amounts of structured data. The goal is to transfer the structured evolution of CA onto the sound synthesis domain. This is always done through a mapping, a set of correspondences between different domains.

There have been different mapping attempts [1] ranging from direct assignments of CA values, like in Lasy [2], to higher-level approaches intending to map the overall CA behaviour, like in Chaosynth [9]. We are interested in the second type of approach. Our research strategy is based on the analysis of CA evolutions by means of digital signal processing techniques in order to discover structural information of their organization. Then we proceed with the mapping of the analysis results onto appropriate synthesis parameters.

This paper is organized as follows. In Section 2 we present the automaton chosen for this study, which is based on the multitype voter model. In section 3 we explain the mapping process adopted, which is based on CA analysis with the histogram technique and, aims at the design of sound spectrograms. In section 4 and 5 we describe the musical features revealed from the histogram analysis of the multitype voter model and, we suggest solutions for the drawbacks found. As the control is the main problem of CA, in section 6 we comment controllability aspects of the sound design process. Sounds are rendered using additive synthesis. Finally, Section 7 concludes this paper.

2 The Multitype Voter Model

The automaton chosen for this research is based on the voter model. In 1953, geneticist Kimura introduced the stepping stone model [7]. This process was studied extensively by other geneticists over twenty years before being rediscovered by probability theorists Clifford and Sudbury in 1973 [3] where it was called the invasion process and by Holley and Liggett in 1975 [6] under the name the voter model [4]. Nowadays, the voter model is considered one of the standard models of interacting particle systems [8].

The voter model is interpreted as a model of opinion formation. A collection of individuals is defined, each of which has one of two possible opinions on a political issue. These possible opinions are denoted by 0 and 1. An individual reassesses his view in a rather simple way: he chooses a "friend" at random with certain probabilities and then adopts his opinion [8]. When the voter model is generalized to more than two opinions it is known as multitype voter model.

It can be also seen as a model of competition. The interpretation is clear from the point of view of the invasion process; different species compete for the territory and the result of conflict is the invasion by one of the species of territory held by the other.

These models can be simulated by means of a probabilistic cellular automaton in two dimensions. The multitype voter model can be implemented with the following transition rule: a number between 0 and 1 is chosen as to be the update probability for all cells. Then, for each cell in the grid, a random number between 0 and 1 is generated at every time step. If the random number generated for the given cell is higher than the update probability, then the state of the cell changes to that of one of its neighbours selected uniformly at random. (Neighbour is defined as the four orthogonally adjacent cells: north, east, south, west) [5].

From a uniform random distribution of cell values, or colours, as the initial configuration, the automaton self-organizes in areas of single colours (Figure 1). As the rule is iterated, some areas will increase their space while others will decrease –to the extent that they can disappear. In the end, one colour will prevail over the others when, according to the voter interpretation, consensus occurs.

The random inputs and probabilities in the rule make that different runs with the same settings result in different evolutions.

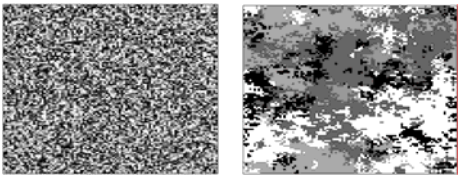


Fig. 1. Two configurations of a multitype voter model evolution. From a random input (left) it self-organizes in coloured areas (right).

3 Mapping Process: From Histograms to Spectrograms

The mapping process adopted in this study is based on a statistical analysis of the CA evolution. The functioning of a two-dimensional automaton is considered as a sequence of digital images and it is analysed by histogram measurements of every CA image. Such a CA analysis gives a histogram sequence.

The histogram of a grey-level digital image is a graphical representation of the number of occurrences of each grey level[1] in the image. By dividing the number of occurrences by the total number of pixels of the image, the histogram is expressed in

[1] Apart from this definition, in this paper we refer to colours instead of grey levels because we usually display the CA in the computer screen using a palette of different colours.

probabilistic terms giving an estimate of the probability of occurrence of each grey-level in the image –the sum of all the histogram bins is equal to one.

The mapping process is very simple; the bins of the histogram sequence are considered to be bins of a spectrogram. With an appropriate automaton, in the histogram sequence it is possible to find structural elements resembling spectral components of a sound. For example, from a histogram analysis of the hodge podge automaton there were discovered structural elements similar to sinusoidal components and others similar to noise components such as noise bands and transients [12]. This makes such mapping process distinctive; in most other cases there is not an intuitive correspondence between the components of the automaton and the components of a sound.

With these structural elements we can design the time varying frequency content of a sound; we can build a spectrogram. This spectrogram can be rendered into sound using different synthesis techniques –the structural elements of the histogram sequences become control data for the synthesis program.

4 Features of the Multitype Voter Model Histogram Sequences

We have found several music-like features that make the histogram sequences of the multitype voter model interesting. Figure 2 shows the histogram sequence of an automaton with 100x100 size and 20 colours through 12000 iterations.

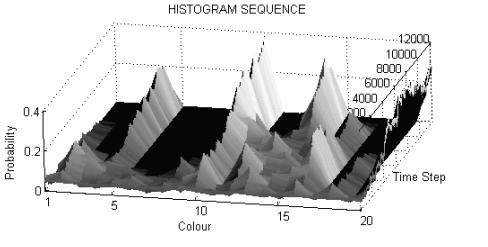


Fig. 2. A histogram sequence of the multitype voter model

The first important characteristic is that the bins of the histogram sequence may represent the time varying amplitudes of sound partials[2]. In addition, note that the multitype voter model allows us to work with as many colours (i.e., partials) as we want.

Secondly, the disappearance of colours during the run attracts our attention. The sounds of acoustic instruments present a similar behaviour; they usually produce more partials in the attack than in the rest of the sound. In the previous example the automaton goes from having 20 colours to 4 in 12000 time steps. We can favour this phenomenon if we work with more colours in a smaller automaton. According to the invasion interpretation, there would be more species competing for less territory, a fact that will provoke a sooner extinction of many species just at the beginning of the run. Figure 3 shows how, in a 30x30 automaton with 50 colours, there is a disappearance of many colours in only 200 time steps.

[2] Assuming this premise, for the rest of the paper we may use the term 'partials' for referring to histogram bins.

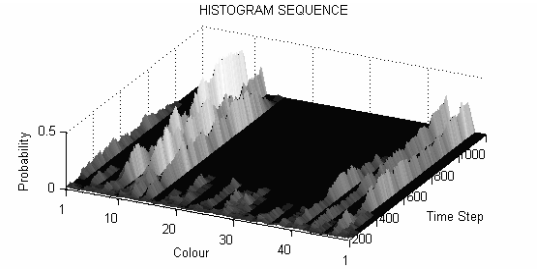


Fig. 3. Inducing disappearance of partials at the beginning of the CA run

The problem now is that the automaton can achieve consensus very soon after a quick disappearance of most of the partials. With this, there will not be an interesting structure for the sustain of the sound. We have devised a solution to this problem. When there remain a determined number of partials (determined in advance) to constitute the sustain structure, the automaton is automatically replicated several times in order to build a bigger one (which will have the same histogram). With more exemplars per species and, what is more important, having provided more space, the remained species can coexist for longer (Figure 4).

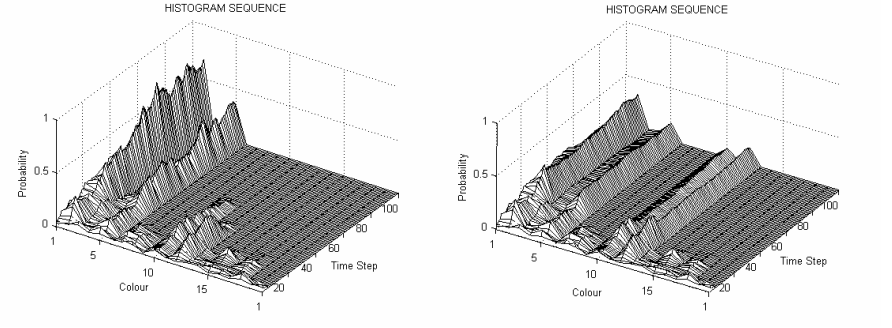


Fig. 4. Controlling spectrum complexity by controlling extinction and coexistence. Histogram sequence of a 10x10 automaton (left). Same evolution with replication in the 30th generation to build a 40x40 automaton (right).

Finally, note that since the automaton has a finite size and all the cells are occupied, when the total area covered by one colour increases then it means that the areas of other colours have decreased. In the histogram sequence it means that when some partials grow, other partials decrease (Figure 5). This is of interest because it reminds us of opposite movements typically fond in polyphonic music. When some voices in the background become important they go to the foreground increasing their intensity, while at the same time, previous foreground voices become less important and go to the background decreasing their intensity.

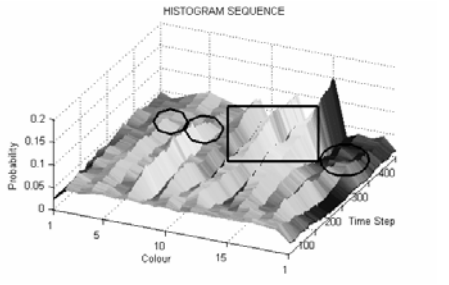


Fig. 5. Opposite movements. The circles show decreasing partials and the square shows increasing partials at the same moment.

Because of the characteristics described above, we find the multitype voter model suitable for rendering single sounds and also polyphonic sound textures.

We still identify two drawbacks. Firstly, before the automaton reaches consensus there will be an increasing prominence of one colour over the others, which may be not desirable. Normally, before this happens we will have enough structure for rendering a sound. Otherwise, as we will see later, it will be possible to exclude a prominent partial or modify its amplitude before rendering the sound.

The second drawback lies in the fact that the histogram sequences do not provide patterns of sound attacks and releases. We treat this matter in the following section.

5 Attacks and Releases

In Figure 6 we can see that the histogram sequence does not start from zero and therefore the synthesized sound does not have an attack pattern. It is also clear that the automaton does not provide either release patterns for all the partials.

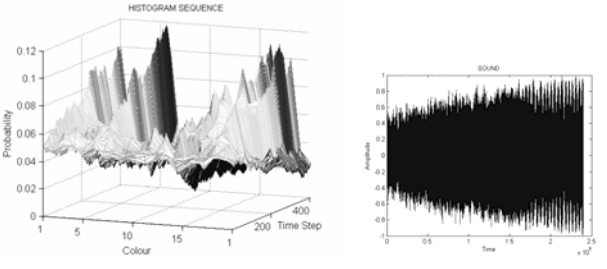


Fig. 6. No attack/release patterns in the histogram sequence (left), neither in the sound (right)

We can always apply external envelopes for creating attacks and releases. But we are interested in giving the automaton complete control over the time varying amplitudes and we have developed a solution by extending the model.

We start the automaton with just one instance of each colour, placed at random locations. With this we guarantee a beginning of the histogram sequence that can be

approximated to zero. We define empty cells for the rest of the automaton. The functioning is the same (the same rule), but in order to ensure a growth (it could be a growth model) we impose that occupied cells can not become empty. With this, the occupied cells will expand covering the whole automaton (Figure 7). At that point the attack is finished.

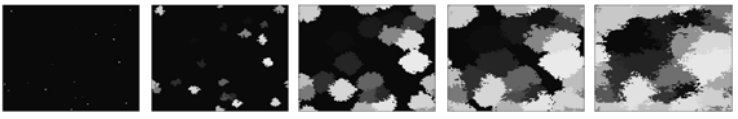


Fig. 7. CA evolution for creating attack envelopes

In Figure 8 we can see, in the histogram sequence, the beginning from zero and the attack patterns. The synthesized sound has as a result a sigmoidal-like attack.

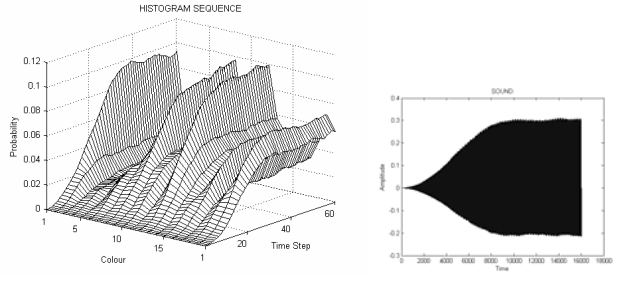


Fig. 8. Attack pattern in the histogram sequence (left), and in the sound (right)

An interesting characteristic of these attacks is that each partial reaches a different amplitude value. This is because the initial colours are located at random locations, and therefore they start to compete for the space at different times (the competition for the space occurs when different colours collide and has as a result a decrease of their growth).

Also, working with relatively big CA we observed that with this solution, partials present more stable amplitudes. This is probably so because the CA start with already established areas of colours, and then during the run, it is more difficult for established areas to experience changes in their sizes. We find this behaviour very interesting for sound synthesis, hence the reason we attempted to capture it in our system.

We have devised an alternative solution that, starting from single dots, and regardless of the amount of them, creates increasing areas but not with specific colours, but with random colours. With this, we fill the automaton with a random distribution of colours.

In order to create releases we have devised a method based on the opposite idea. We introduce sources of "epidemics" at random locations, which will expand "killing" all the cells. The curves obtained look like sigmoidals, with different release times for each partial (due mostly to the random locations of the epidemics), having as a result that the strongest partial is not necessarily the last one that disappears (Figure 9).

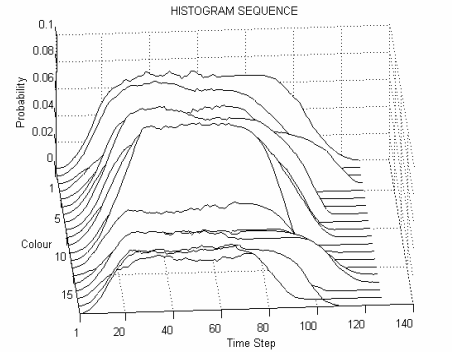


Fig. 9. Histogram sequence of a 100x100 automaton, with attacks and releases. Note that there is not disappearance of colours in the attack due to the relatively big size of the automaton.

In terms of implementation, the empty cells of the attacks and the dead cells of the releases may correspond to negative cell values not considered in the computation of the histograms.

6 Control

The multitype voter model is controlled with four input parameters: the size, the number of colours, the initial configuration, and the update probability. However, the predictability of the outcome of CA is an open problem; it is not possible to predict the value that a specific cell would hold after a number of generations [14]. This is even more obvious if we have random inputs and probabilities in the rules. Therefore, although a level of unpredictability is accepted and often desired in systems for generating music and sound, being under unpredictability conditions implies limited controllability. A lack of a reasonable level of control restricts the music or sound design process [11]. Our work alleviates this limitation in many respects.

Firstly, it is possible to find direct and intuitive relations between the multitype voter model input parameters values and their effects in the histogram sequence. For example, in section 4 we have seen how to cause the disappearance of partials and enable coexistence by controlling the relationship between the size and the number of colours. It is also clear that the update probability controls the amount of cell-colour updates that occur at each generation. Therefore, it controls the rate at which the automaton and thus, the histogram sequences, evolve towards consensus. The initial configurations provide control over the attacks. For example, starting with more than one instance per colour we can make the attack structure shorter –the same idea can be applied to the releases. With different number of instances of each colour we can control, to a certain extent due to the random evolutions, the relative amplitude of the partials (Figure 10).

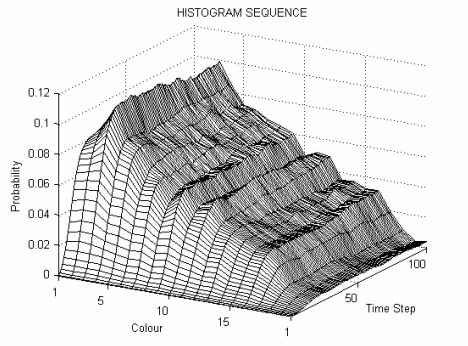


Fig. 10. Controlling relative amplitudes. The number of initial instances of each colour was inversely proportional to the colour value.

We can also model attack delays of the partials by introducing the different colours at different generations (Figure 11).

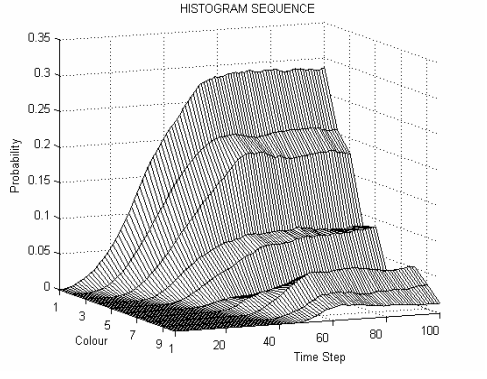


Fig. 11. Controlling spectrum complexity in the attack. Successive colours appear with a delay of 5 CA generations.

Finally, another important aspect of controllability is the possibility of developing a sound design process from the structural elements of the histogram sequences. Conceptually, the first steps to be performed are the assignment of frequencies to the partials and, the specification of the sound duration. From here, other spectral transformations are possible. Time stretching, pitch shifting and amplitude modifications of each partial are straightforward to implement. With all this we are able to design different spectrograms.

For this research we have chosen additive synthesis of sinusoidal components, but other synthesis techniques can be considered to be controlled with these histogram sequences. This is a venue we might explore in the future.

7 Conclusion and Further Work

In this paper we have reported on the synthesis of sounds by the computer simulation of natural systems with CA. The multitype voter model exhibits rich dynamics from a very simple rule and few input parameter specifications. From a histogram analysis we have obtained complex structures endowed with musical features, suitable for the design of spectrograms. A sound design process is possible thanks to the controllability achieved.

We have synthesized single sounds (with durations in the order of seconds) with dynamic spectra and controlled complexity, and also polyphonic sound textures (with durations in the order of tens of seconds) with interesting internal evolutions.

We are currently investigating the inclusion of a mutation process –mutations are often considered in genetic models. The possibility that new species can enter the system through genetic mutation suggests potential applications in the synthesis of polyphonic sound textures.

Examples of sounds synthesised by our new method will be (or were) played at the conference and they are available by request.

References

1. Burraston, D., Edmonds, E.: Cellular Automata in Generative Electronic Music and Sonic Art: A Historical and Technical Review. Digital Creativity 16, 165–185 (2005)
2. Chareyron, J.: Digital Synthesis of Self-Modifying Waveforms by Means of Linear Automata. Computer Music Journal 14, 25–41 (1990)
3. Clifford, P. and Sudbury, A.: A Model for Spatial Conflict. Biometrika. 60, 581–588 (1973)
4. Cox, J.T., Durrett, R.: The Stepping Stone Model: New Formulas Expose Old Myths. The Annals of Applied Probability 12, 1348–1377 (2002)
5. Cox, J.T., Griffeath, D.: Recent results for the stepping stone model. In: Percolation Theory and Ergodic Theory of Infinite Particle Systems. Springer, New York (1987)
6. Holley, R.A., Liggett, T.M.: Ergodic Theorems for Weakly Interacting Infinite Systems and the Voter Model. The Annals of Probability 3, 643–663 (1975)
7. Kimura, M.: Stepping-stone Model of Population. Annual Report of the National Institute of Genetics 3, 62–63 (1953)
8. Liggett, T.M.: Interacting Particle Systems. Springer, New York (1985)
9. Miranda, E.R.: At the Crossroads of Evolutionary Computation and Music: Self-Programming Synthesizers, Swarm Orchestras and the Origins of Melody. Evolutionary Computation Journal 12, 137–158 (2004)
10. Miranda, E.R., Biles, J.A.: Evolutionary Computer Music. Springer, London (2007)
11. Miranda, E.R., Wanderley, M.M.: New Digital Musical Instruments: Control and Interaction beyond de Keyboard, A-R edn., Middleton, WI (2006)
12. Serquera, J., Miranda, E.R.: Spectral Synthesis and Control with Cellular Automata. In: Proceedings of the International Computer Music Conference ICMC, Belfast, UK (2008)
13. Wolfram, S.: A New Kind of Science Online, `http://www.wolframscience.com/reference/notes/876b` (accessed on February, 2009)
14. Wolfram, S.: Computational Theory of Cellular Automata. Communications in Mathematical Physics 96, 15–57 (1984)

Sound Agents

Philippe Codognet[1] and Olivier Pasquet[2]

[1] JFLI – CNRS / UPMC / University of Tokyo
Information Technology Center, 2-11-16 Yayoi, Bunkyo-ku, Tokyo 113-8658, Japan
[2] Independent sound artist & IRCAM, 1 place Igor Stravinsky, 75004 Paris, France
codognet@jfli.itc.u-tokyo.ac.jp, olivier.pasquet@ircam.fr

Abstract. *Sound Agents* is a media intallation relating real space and virtual sound space. Each agent is a virtual entity producing sound, which has its own autonomous behavior. This sound is spatialized in the real installation space through many loudspeakers (24 loudspeakers + 1 subwoofer), creating thus an ever-changing ambient music which is dynamically spatialized and modified by the movement of the virtual agents. We implemented a first propotype of this general scheme by using swarm intelligence and the classical ant foraging simulation to generate ambient soundscape, associating sounds to ants movements and pheromone levels. We further designed a declarative high-level language for describing autonomous behaviors of the virtual sound agents.This language is based on the notion of goal constraints and simple constraint-based local search techniques are defined as a behavior engine.

1 Introduction

The basic idea of the *Sound Agents* system is to create an immersive sound-space by relating real space and virtual sound space. In can be seen in the tradition of immersive Virtual Reality systems where video projections and computer graphics will recreate in a real space a 3D virtual world, cf. for instance the well-known CAVE system develop by University of Illinois in the early 90's. In the last two decades, advances in computer graphic rendering techniques and efficiency of specialized hardware, improved visual immersion, but to further reinforce the immersive aspects of such virtual environments, the idea of populating virtual spaces with virtual creatures or agents has been growing in the recent years a major focus of research. The motivation of the *Sound Agents* system is rooted in the development of virtual autonomous entities for immersive environments. However in *Sound Agents*, each virtual entity will not be a visual character but an invisible sound agent producing sound, which will have its own autonomous behavior. This sound will actually be localized in the actual 3D space of the installation, much as a CAVE-like VR system. Sound agents will be like bees or butterflies flying around spectators, but you cannot see them, just hear them. Therefore the ambient sounds will be dynamically spatialized in the actual installation space and will be modified by the movement of the virtual agents, providing thus an ever-changing musical soundscape.

The media installation *Sound Agents* aims at creating in real-time ambient electronic music from a multi-agent simulation, associating agents with sounds and

C. Di Chio et al. (Eds.): EvoApplications 2010, Part II, LNCS 6025, pp. 391–400, 2010.

associating various sound parameters with the internal variables of an agent at a given time (position, orientation, internal state, etc). We consider sound entities to be reactive agents. That is, simple entities that receive percepts from their environment and can act on the environment by performing actions, e.g. to navigate in a 3D world and produce sounds. Reactive agents have no symbolic model of the world they live in, but rather use sensory-action control loops in order to perform tasks in a robust manner. In its current version, the *Sound Agent* system uses Nature-inspired multi-agent simulations to drive the musical processes, and in particular the swarm intelligence metaphor [3]. The key interest of swarm intelligence lies in the fact that only simple and easy to understand local behavior rules have to be programmed for each agent, while a global complex behavior will be exhibited by the overall population of agents, the so-called emergent behavior. Therefore this seems to us quite intuitive and easily understandable by non-specialists, e.g. music composers, who could use this metaphor in order to construct musical works with some part of randomess (the actual movement of each agent) but nevertheless subject to a predictable emergent behavior. However for more versatile or precise simulations, we need a high-level language for describing more complex, goal-oriented behaviors. We have thus defined a simple but effective high-level language to describe the autonomous behaviors of the sound agents. We propose to use the formalism of CSP (Constraint Satisfaction Problems) as a general behavior description language. Constraints are used to state goals, or more exactly partial goals, that the agent has to achieve. Each agent thus manages a set of current goals that represent its possibly competing or even conflicting goals. Constraint solving occurs at each time step in order to reduce the conflict between the current situation and ideal goal satisfaction (where all constraints are solved) and thus optimize the (partial) realization of the goals. We hope that the abstraction level of goal constraint is high enough to be easily understandable and managed by users with a minimal programming background.

The paper is organized as follows. Section 2 details the general architecture of the Sound Agents system and the basic hardware. Section 3 presents the swarm intelligence simulation and the current prototype, which is based on ant foraging. To further develop high-level multi-agent simulations, Section 4 proposes a declarative language for describing agent behaviors, based on the notion of goal constraints. A short conclusion end the paper.

2 Hardware Implementation of the Sound Agents System

The current prototype of *Sound Agents* consists of integrating a Java-based multi-agent simulation engine, the Mason system [15], in the Max/MSP real-time sound generation software. Max/MSP is controlling the system parameters and is giving the timing, in order to have the simulation system iterations synchronized with the rest of the audio engine. It is thus possible to use a time grid for giving the agents a pulse to develop their step-time behaviors and progress to the next time-step. Concerning the actual rendering in real-time of the spatialized, moving sounds, the best would be to use Wave Field Synthesis (WFS) technology [24]. Such systems are commercially available but very costly. We have thus rather designed a low-cost approximation by using many small loudspeakers located in the installation space (using three 8-channel

sound cards and thus 24 speakers), organized as a 6 x 4 matrix, plus one global subwoofer. The system configuration is composed of one Apple MacBook Pro, two M-Audio Profire 2626 soundcards (2 x 8 outputs), one Behringer ADA8000 ADAT soundcard (8 outputs), 24 loudspeakers (M-Audio Studiophile AV30), and one subwoofer (Tapco SW-10).

Fig. 1. The installation space with 24 loodspeakers for sound spatialization

3 Swarm Intelligence

In the recent years, multi-agent systems have however attracted the attention of computer-music researchers. The framework of [2] proposed a general context for self-organized music and a particular application of swarm music for improvisation performances. [9] briefly presents without much details a composition system based on agents who can alter a "musical space" by adding, removing or moving sound sources (not performing sound synthesis themselves), but the authors say themselves that the agent behaviors and the overall system seems quite difficult to program. Also [10] presents an interactive system which is based on the simulation of particle swarms (e.g. like fish schooling or bird flocking), which is interactive: the general movement of the swarm can be controlled in real-time by a musician. It is thus more an electronic musical instrument for live performances rather than a generative music system. We use a simple swarm intelligence simulation, namely ant foraging, for our *Sound Agent* prototype. This generative music system is fully detailed in [7] and will be summarized in the rest of this section.

Ant foraging is a classical example of swarm intelligence that has been studied by entomologists and then simulated in the computer since the early 90's [1]. The basic idea is that the ants deposit pheromones on the paths that they cover and as ants going on the shortest path are getting back to the nest quicker and will then go again for more food searching, this builds an optimal path that will contain more pheromone than others. The behavior of each agent is simple and can be described as follows:

while walking and searching for food, ants may (1) deposit a pheromone on the ground, (2) follow with high probability pheromone trails that they sense on the ground. The emergent behavior is that optimal or near-optimal paths will be created because ants using those paths are going back faster to the nest and therefore will go again and deposit more pheromones. Recent research in insect communication show that maybe several types of pheromone are indeed used, e.g. a 'no-entry' pheromone might be deposited to block "dead-end" paths [20]. This is not however implemented in our simulation, which is the one implemented in the Mason system.

Ant foraging simulation is thus used to produce generative ambient music in real time. About one thousand agents are used in the simulation but it would be too costly in computation time to associate one sound source to each agent, as the sound has to be generated in real-time for each agent. With our current computing power (single computer) we have to limit to about 25 to 30 the number of sound sources, which will be randomly chosen among all the agents. Observe however that, in a simulation with 1000 agents, the 975 « non-audible » agents are nevertheless useful because they interact and will influence the behaviors of the 25 audible agents through stymergy (pheromone deposit). Sound sources associated to audible agents are synthetized sounds created by simple oscillators whose parameters depend on the position and orientation of the agents. We used simple oscillators for performance reasons, but more complex treatments are of course possible with Max/MSP and will be experimented if we can have extra computing power. It is interesting to note that in order to have better performances, we also had to go down to the parallel processing aspects of the dual core processor of the Powerbook Pro and explicitely program Mason and Max/MSP to run on different cores and communicate via the main memory. This has greatly improved the performances of the system. Concerning the sound spatialization, we could have done a naive mapping of the 2D simulation space to 2D matrix of loudspeakers, but we prefered to have a different one, linked to the distance of each agent to the optimal path. Therefore the overal movement of the simulation, which is that all ants first walk randomly in the 2D space and eventually converge to a path close to the optimal (i.e. the line linking the nest to the food, as we have no obstacle in this simulation), will be musically reflected by the fact that all audible agents will eventually converge to a common rhythm, with minor variations. We also decided to associate a musical meaning and a specific sound generation to the pheromone deposit itself and add an extra sound source which produce some continuous bass chords. The volume of this drone depends on the level of pheromone, and this sound is further modified by a distortion effect whose parameters depends on the location of the pheromone deposit.

The whole idea of using multi-agent simulation and swarm intelligence for generating music is to be able to associate to each agent some sound parameters that will be sumitted to variations depending on the position of the agent. Then the emergent property will ensure that agents will eventually converge towards some optimal path and therefore that some musical movement can be achieved, even if the actual timing or exact value the position of each agent of the swarm cannot be precisely defined in advance. However it is up to the musician to define how to use the emergent property in his musical movement. Therefore the mapping between the swarm agents and the sound parameters is of key importance. Although best appreciated in an exhibition space with 24+1 loudspeakers, the resulting ambient

music can be appreciated by looking at the following website, which include mp3 recordings and videos : http://webia.lip6.fr/~codognet/SA/

4 A Declarative Language for Describing Agent Behaviors

In the current version of *Sound* Agents, we use the Mason multi-agent system [15] in order to drive the simulation. Although a powerful and efficient system, Mason nevertheless requires serious programming skills for describing agent behaviors and it is thus difficult for a composer to directly modify the simulation code, or to experiment a new type of multi-agent simulation by himself. Therefore our goal is to develop a high-level language to describe agent behaviors that could be easily understood by someone with limited programming skills. Simple but interesting life-like behaviors with emergent properties should be easily implemented, such as those described in [4]. In computer graphics and animation systems, the most common formalism for representing behaviors of high-level agents, such as virtual humans is some extension of finite state automaton (FSA) [17,26] or more complex hierarchical models [23,14]. For low-level agents, such as the swarm agents in flocks or herds and reactive agents, two basic approaches are classically used:

1. Steering behaviors, where the different low-level goals (such as grouping or escaping) are stated as forces that are then added to produce the actual behavior of the agent in a time-step manner. This approach has been pioneered by Reynolds since the late 80's [19,20], but it still active now and various extensions have been proposed [18,23,8].
2. Particle systems [25] or potential fields [12] treating the swarm as a complex physical system.

We are obviously closer to the first approach, but we propose to use the formalism of CSP (Constraint Satisfaction Problems) as a general behavior description language. Constraints are used to state goals, or more exactly partial goals, that the agent has to achieve. This can be seen as an extension of the steering behavior approach where constraints are solved logically instead of forces added numerically. One interesting point however is that the constraint formalism is naturally nondeterministic, as opposed to any force-based formalism such as steering behaviors, which is intrinsically deterministic. Indeed we find here again the classical dichotomy between declarative and procedural languages. We believe that a declarative, nondeterministic formalism such as that of goal constraint is more powerful and easier to use than a procedural one.

4.1 Goal Constraints and Local Search Constraint Solving

Constraints, i.e. logical relations, are an interesting tool at a declarative level to represent goals, but we also need to solve these constraints for achieving the goals. We have developed in previous work an efficient constraint-based combinatorial optimization algorithm named "adaptive search" [6], core ideas of which are:

1. to consider for each constraint a heuristic function that is able to compute an approximated degree of satisfaction of the goals (the current "error" on the constraint);

2. to aggregate constraints on each variable and project error on variables thus trying to repair first the variable with worst "error"; and
3. to update and refine these heuristic functions as the agent explores the environment and discovers new information, such as the position of some obstacles.
4. to use some sort of "Tabu list" in order to give the search engine a short-term memory and avoid having it trapped in loops and local minima.

Consider an n-ary constraint $c(X_1, \ldots, X_n)$ and associated variable domains $D_1, \ldots, D_n$. An error function f_c associated to the constraint c is a real-valued function from $D_1 \times \ldots \times D_n$ such that $f_c(X_1, \ldots, X_n)$ has value zero if $c(X_1, \ldots, X_n)$ is satisfied. The error function will in fact be used as a heuristic value to represent the degree of satisfaction of a constraint and will thus give an indication on how much the constraint is violated. For instance in path-planning applications and spatial goal constraints, the error function can be seen as (an approximation of) the distance of the current configuration to the closest satisfiable region of the constraint domain. Since the error is only used to heuristically guide the search, we can use any simple approximation when the exact distance is difficult (or even impossible) to compute. It is worth noticing that the idea of using simple heuristics to guide the behavior of humans or animals has been recently proposed in many cases by both psychologists and biologists [11].

Adaptive Search is a simple algorithm but it turns out to be quite efficient in practice. Considering the complexity/efficiency ratio, it can be a very effective way to implement constraint solving techniques in larger software tools, especially for *anytime algorithms* where (approximate) solutions have to be computed within a limited amount of time. The efficiency of the Adaptive Search algorithm for large combinatorial problems is detailed in [6]. As we do not expect that the use of Adaptive Search for solving goal constraints will give rise to hard combinatorial problems, we will present in the following a simplified version of the algorithm, that will just compute an iterative improvement for the next time-step. As this procedure will be called at each time-step by the agent simulation loop, this will converge towards a solution that is the realization of the goals. We also consider that this algorithm will be used in the context of generating a continuous behavior and thus the modification of the variables is subject to some limitation (e.g. the agent has a maximal speed for its movements). This can be modeled by considering a modification limit for each of the agent variables. If the limit is reached, then the algorithm will stop and output the current best partial solution.

Input: current configuration with
- a set of variables $V=\{V_1, V_2, \ldots, V_n\}$ with associated domains
- a set of constraints $C=\{C_1, C_2, \ldots, C_k\}$ with error functions
- functions to combine constraint errors on variables (e.g. simply the sum)
- a (positive) cost function to minimize (usually a linear combination of errors)
- a parameter T : the Tabu tenure (number of iterations a variable is frozen)

Output: next configuration with minimal cost

Algorithm:
repeat

1. Compute errors of all constraints in C and combine errors on each variable (by considering only the constraints in which a variable appears)
2. select the variable X (not marked Tabu) with highest error
3. evaluate costs of possible moves from X
4. **if** no improvement move exists
   then mark X as Tabu for T iterations
   else select the best move and change the value of X accordingly

until solution is found **or** modification limit from initial configuration is reached

4.2 Goal Constraints for Navigation

In the *Sound Agent* project, we are mainly concerned with the motion of agents in a 2D or 3D space; therefore we should focus on the specific goal constraints related to navigation. Indeed we want to generate the trajectory of an agent as the (iterative improvement) solving of the goal constraints at each time-step, which will generate the actual movement of the agent. Therefore we can simply consider a single variable for each agent, which is its position in the virtual space. As we are in a combinatorial search setting, we will consider that the domains of the variables are representing some discrete approximation of the real space. The key idea is that the agent will look at the possible positions in his neighborhood, check the combination of errors of his own goal constraints and choose the position minimizing the error. Moreover the use of a Tabu mechanism makes it possible to forbid some areas for a given period of time (Tabu tenure) and thus helps to prevent being trapped in a local minimum (e.g. a spatial dead-end) or short-term oscillating behaviors. Indeed comparing to the steering behavior paradigm, we have here non-deterministic behaviors as we do not try to define a combination of forces that will bring the agent to the desired position but just check possible locations and choose the best one, which can be done efficiently in the local search approach. Also observe that we can cope with dynamically changing priorities between behaviors, as the error function are reevaluated at each time steps.

To be more precise about declarative navigation goals, we can consider the constraints defined in [5] for path-planning:

Table 1. Examples of Goal Constraints

Constraint	**Declarative meaning**
In(Region)	Stay within the zone define by *Region*
out(Region)	Stay outside the zone define by *Region*
Go(Object)	move towards the location of *Object*
away(Object)	move away from the location of *Object*
Attraction(Stimulus)	Move towards source of stimulus
Repulsion(Stimulus)	Move away from tsource of stimulus

These declarative constraints will reduce to (or, for efficiency, be approximated by) some simple arithmetic constraints.

For instance the constraint *In(Region)* states that an agent should stay within a circular *Region* will reduce to : *Agent.position - Region.Center < Region.Radius* and the constraint *go(Object)* will reduce to: *Agent.position - Object.position < 0.01*

It is clear that a combination of such goal constraints could produce quite complex behaviors, e.g. that the agent should go towards some object, avoid all objects it perceives and stay away from some predefined regions. For instance a "follow" behavior can be simply obtained as a combination of a *go* constraint (to move toward the followed agent) and an *out* constraint (to stay at a certain distance). Observe however that for the last two constraint goals, the agent does not know the location of the source of the stimulus, but it can only sense the amount of stimulus received at some location by one or more sensors, using either a *temporal difference* or a *spatial difference* method [22]. The temporal difference method consists in considering a neighborhood reduced to a single location (just ahead of the agent), while the spatial difference method will consider two or more neighbors, although it seems that considering more than two sensors does not improve much the performances. If none of the neighbors improve the current amount of stimulus (objective function), then a random move will be performed.

5 Conclusion

The architecture of *Sound Agents* has been designed and preliminary experiments conducted. They consist in (1) transferring a multi-agent architecture that has been developed for 3D virtual agents to sound-based agents and (2) developing a sound rendering hardware to realize the spatialization of the sound agents in the actual installation space. *Sound Agents* is a sound-based installation relating a virtual space and the actual 3D space of the exhibition. Agents are associated to sound sources whose parameters are modified in real-time depending on the values of the agent's state (position, internal variables, etc). Therefore the ambient music generated by the multi-agent simulation produces an ever-changing music soundscape. The first experimental instance consists in an ant foraging simulation and 24 loudspeakers (+ 1 subwoofer) organized as a 4x6 matrix on the floor. An interesting extension would be to realize this installation in a bigger space (e.g. 10m x 10m room) and to put the loudspeakers on small columns, in order to have a better spatialization and audio perception throughout the installation.

Concerning the modeling aspects, we have designed a general framework for describing autonomous agent behaviors based on the goal constraint formalism and a local-search optimization algorithm. We hope that this could be a basis for a more general use of multi-agents systems for generative music and interactive installations.

References

1. Beckers, R., Deneubourg, J.L., Goss, S.: Trail laying behaviour during food recruitment in the ant Lasius niger (L). Insectes Sociaux 39, 59–72 (1992)
2. Blackwell, T., Young, M.: Self-organized music. In: Organised Sound, vol. 9(2), pp. 123–136. Cambridge University Press, Cambridge (2004)

3. Bonabeau, E., Dorigo, M., Theraulaz, G.: Swarm Intelligence: From Natural to Artificial Systems. Oxford University Press, Oxford (1999)
4. Braitenberg, V.: Vehicles - Experiments in Synthetic Psychology. MIT Press, Cambridge (1984)
5. Codognet, P.: Animating virtual creatures by constraint-based adaptive search. In: Proceedings of VSMM 2000, 4th Int. Conf. on Virtual Systems and Multimedia, Gifu, Japan. IOS Press, Amsterdam (2000)
6. Codognet, P., Diaz, D.: An Efficient Library for Solving CSP with Local Search. In: Ibaraki, T. (ed.) Proc. MIC 2003, 5th International Conference on Metaheuristics, Kyoto, Japan (2003)
7. Codognet, P., Pasquet, O.: Swarm Intelligence for Generative Music. In: Proceedings of ISM 2009, IEEE International Symposium on Multimedia. IEEE Press, San Diego (2009)
8. Couzin, I., Krause, J., James, R., Ruxton, G.D., Franks, N.R.: Collective Memory and Spatial Sorting in Animal Groups. Journal of Theoretical Biology 218(1), 1–11
9. Dahlstedt, P., McBurney, P.: Musical Agents: Towards Computer-aided Music Composition using Autonomous Software Agents. Leonardo 39(5), 469–470 (2006)
10. Davis, T., Karamanlis, O.: Gestural Control of Sonic Swarms: Composing with Grouped Sound Objects. In: Proc. SMC 2007, 4th Sound and Music Conference, Lefkada, Greece (2007)
11. Hutchinson, J., Gigerenzer, G.: Simple heuristics and rules of thumb: Where psychologists and behavioural biologists might meet. Behavioural Processes 69(2), 97–124 (2005)
12. Jin, X., Wang, C.C.L., Huang, S., Xu, J.: Interactive control of real-time crowd navigation in virtual environment. In: Proc. 2007 ACM symposium on Virtual Reality Software and Technology, pp. 109–112. ACM Press, New York (2007)
13. Krause, J., Ruxton, G.D., Krause, S.: Swarm intelligence in animals and humans. Trends in ecology & evolution 24(9) (2009)
14. Lamarche, F., Donikian, S.: Crowd of Virtual Humans: a New Approach for Real Time Navigation in Complex and Structured Environments. Computer Graphics Forum 23(3), 509–518 (2004)
15. Luke, S., Cioffi-Revilla, C., Panait, L., Sullivan, K.: MASON: A New Multi-Agent Simulation Toolkit. In: Proc. SwarmFest 2004, 8th Swarm Users/Researchers Conference, Ann Arbor, USA (2004)
16. Musse, S.R., Thalmann, D.: Hierarchical Model for Real Time Simulation of Virtual Human Crowds. IEEE Transactions on Visualization and Computer Graphics 7(2), 152–164 (2001)
17. Noma, T., Zhao, L., Badler, N.I.: Design of a Virtual Human Presenter. IEEE Computer Graphics and Applications 20(4), 79–85 (2000)
18. Pedica, C., Vilhjalmsson, H.: Social Perception and Steering for Online Avatars. In: Prendinger, H., Lester, J.C., Ishizuka, M. (eds.) IVA 2008. LNCS (LNAI), vol. 5208, pp. 104–116. Springer, Heidelberg (2008)
19. Reynolds, C.W.: Flocks, Herds, and Schools: A Distributed Behavioral Model. Computer Graphics 21(4), 25-34 (1987); SIGGRAPH 1987 Conference Proceedings
20. Reynolds, C.W.: Steering behaviors for autonomous characters. In: Proc. of 1999 Game Developers Conference, pp. 763–782. Miller Freeman Group, San Francisco (1999)
21. Robinson, E., Jackson, D., Holcombe, M., Ratnieks, F.: 'No entry' signal in ant foraging. Nature 438(7067), 442 (2005)
22. Schmajuck, N. (ed.): Special issue on Biologically-inspired models of Navigation, Adaptive Behavior, vol. 6(3/4) (1998)

23. Shao, W., Terzopoulos, D.: Autonomous pedestrians. In: Proc. of the ACM SIGGRAPH 2005, Symposium on Computer Animation. ACM, New York (2005)
24. Theile, G.: Wave Field Synthesis – A Promising Spatial Audio Rendering Concept. In: Proc. DAFX 2004, 7th Int. Conference on Digital Audio Effects, Naples, Italy (2004)
25. Treuille, A., Cooper, S., Popovic, Z.: Continuum crowds. ACM Transactions on Computer Graphics 25(3), 1160–1168 (2006); SIGGRAPH 2006 Conference Proceedings
26. Xiao, H., He, C.: Behavioral Animation Model for Real-Time Crowd Simulation. In: Proc. of 2009 International Conference on Advanced Computer Control. IEEE Press, Los Alamitos (2009)

From Evolutionary Composition to Robotic Sonification

Artemis Moroni[1] and Jônatas Manzolli[2]

[1] CTI Renato Archer, Robotics and Computer Vision Division, Rod. D. Pedro I, km 143, 6, 13069-901 Campinas, SP, Brazil
[2] Unicamp, Interdisciplinary Nucleus of Sound Studies, Rua da Reitoria, 165, Cidade Universitária "Zeferino Vaz"13083-872 Campinas, SP, Brazil
Artemis.Moroni@cti.gov.br, Jonatas@nics.unicamp.br

Abstract. A new approach is presented which integrates evolutionary computation and real world devices such as mobile robots and an omnidirectional vision system. Starting with an evolutionary composition system named JaVOX, a hybrid environment named AURAL evolved. In the AURAL, the behavior of mobile robots in an arena is applied as a compositional strategy. It uses trajectories produced by mobile robots to modify the fitness function of a real time sonic composition. The model is described, its evolutionary design and how the interaction between the real world devices was implemented. This research is oriented towards the study of automatic and semi-automatic processes of artistic production in the sound domain.

Keywords: Algorithmic composition, evolutionary computation, robotics, sonification.

1 Introduction

Evolutionary Computation applied to real time sonification is a paradigm for creative evolution in music systems, particularly in algorithmic composition. Interaction between music and the evolutionary algorithm has been studied following human improvisation systems such as Todd & Werner [1], Biles [2] and Yee-King [3]. Manzolli & Verschure [4] created the Roboser System, a combination of the behavior of the Khepera robot with a large neuronal interaction system IQR. Subsequently, the Roboser system was used in the interactive installation "Ada: intelligent space." In Ada, human collective behavior and a set of real world devices were used to express synthetic emotions [5]. Besides providing paths or behavior to be mapped into sound, the robotic systems may be guided by sound signals. In this case, the goal is to study the control of the robot positioning through a sound stimuli, as presented by Murray [6].

AURAL environment, which uses real world devices for evolving music material using Evolutionary Computation is presented in this paper. Like the foregoing works, AURAL attempts to autonomously generate complex sonic structures by exploring the dynamics of the real world interaction between artefacts and their environments, including human beings. Similar to the systems developed by Manzolli & Verschure [4] and Murray [6], the AURAL system organizes a sequence of sound events based on the interaction of mobile robots in an arena. Unlike these two systems, the sonification is

C. Di Chio et al. (Eds.): EvoApplications 2010, Part II, LNCS 6025, pp. 401–410, 2010.

controlled by robotic trajectories associated with the fitness function of the evolutionary sound environment, JaVOX [7]. Conceptually, AURAL is related to arTbitrariness, defined as a theoretical perspective for studying automatic and semi-automatic processes of artistic production [8].

The next section presents the basic elements of the system: the interactive evolutionary graphical interface, the omnidirectional vision system and the robotic control. In Section 3, experiments are described, followed by the results and a conclusion.

2 AURAL Architecture

AURAL is a system of third generation in which evolutionary computation was applied to algorithmic composition. The first one, VOX POPULI, was based on a mathematical model for automatic fitness evaluation of music structures described by MIDI events. The musical fitness was defined by heuristics for harmonic, melodic criteria and a fuzzy set modeling, as described in [9, 10]. Since then, this research has explored the parametric manipulation of that fitness criterion. A graphical interface was developed allowing the user to draw curves interfering in the fitness result, fully implemented in JaVOX [7]. In AURAL, the curves previously drawn by the user are combined with robotics trajectories. Thus, it is possible to have control curves drawn by humans beings or curves "drawn" by the robots (i.e. trajectories performed in an arena) and even both.

The AURAL system, depicted in Figure 1, is made up of the following elements: a) the evolutionary composition interface JaVOX; b) the OmniEye, the artificial vision system, c) the supervisor module, TrajeCt, which receives the trajectory and supervises the master robot, d) a community of robots.

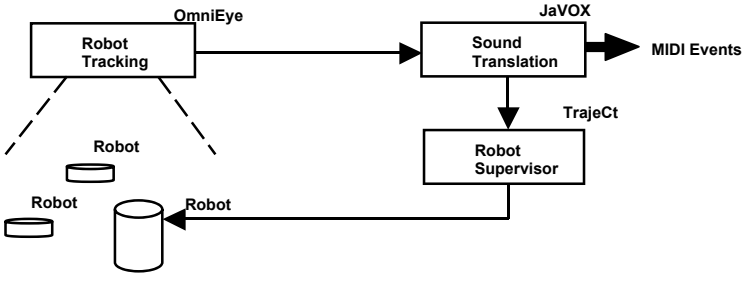


Fig. 1. The AURAL architecture diagram

The sonification produced by the AURAL system is based on a generative cycle, i.e., the user draws control curves on the GUI of the evolutionary sonification module, JaVOX. The curves are transmitted as trajectories to a master robot, Nomad, moving in an arena. The omnidirectional vision system, OmniEye, observes the navigation of the Nomad [11] while interacting with other mobile robots, the Creates [12]. The real time data sensed by the OmniEye is fed back into the JaVOX evolutionary composition system producing new MIDI events. The cycle is repeated until the robots are stopped.

2.1 The OmniEye

The OmniEye is the first module that is activated when the AURAL system is run. It occupies an important role, for it is the "observer" being used to feed back the robot localization. Part the AURAL development, the OmniEye is made up of a camera, a spherical convex mirror and a conical weight to align the camera and stabilize the set up [13]. Figure 2 shows, on the left, the OmniEye; on the right, the CamShift window [14], with an image of the whole arena.

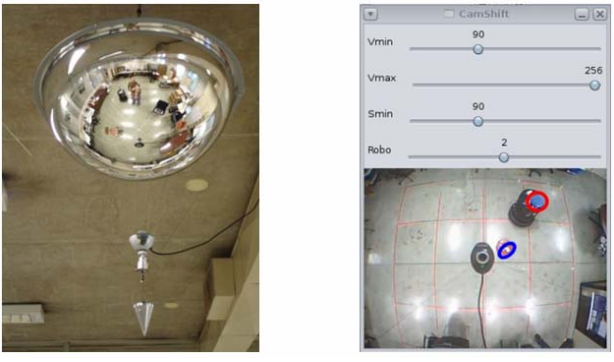


Fig. 2. On the left, the OmniEye with the spherical mirror and the camera. On the right, an image of the arena obtained with the OmniEye.

The CamShift window, the interface of the OmniEye, is the first one to appear when AURAL starts. A strongly colored panel was fixed on each robot for tracking. The user marks each robot with the mouse on the window, assigning a *voice number* to it (to be seen in Section 2.2.1). A colored circle appears. The coordinates of the mass center of the circle are used to estimate the position of the robot.

2.2 The Evolutionary Sound Interface

JaVOX, the evolutionary sound interface, is the kernel of the AURAL system. It is activated just after the OmniEye and has four main areas: a *Parameter Control;* a *Graphic Area; Dynamic Controls and* a *Performance Control,* depicted in Figure 3, on the left.

In the Graphic Area the user may draw a curve (red) to be sent as a trajectory to the Nomad, the master robot. When the Play button (Dynamic Controls) is pushed, different processes are activated, such as, the trajectory is sent to the master robot, Nomad, as a sequence of points; the robots start to move and the sound production is triggered. The paths crossed by the robots, observed by the OmniEye, are plotted as curves in different colors on the Graphic Area.

The Graphic Area is associated with a conceptual sound space having two phase spaces, the "red" one, or *melodic*, and the "blue" one, or *rhythmic*. In the melodic phase space the x-axis is associated with the tonal center ([0, 11], representing the tones and semi-tones of a musical octave), and the y-axis with the octave range ([0, 127] in MIDI protocol). The rhythmic space has a diversing parameter ([1, 30])

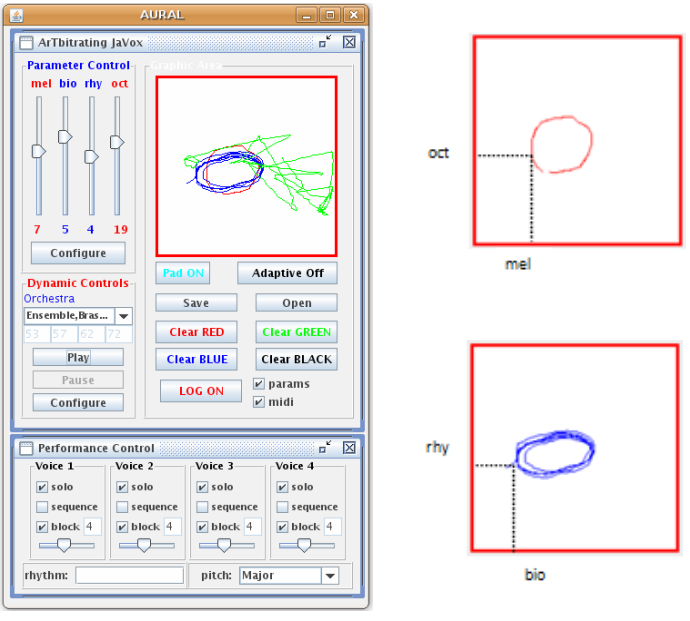


Fig. 3. On the left, JaVOX interface. On the bottom, the Performance Control with Voices 1, 2, 3, 4 (each voice is associated with a robot) and solo, block, sequence and block parameters. On the right, the mappings of the parameters melodic (*mel)* and octave range (*oct)* with the melodic phase space and the parameters rhythm (*rhy*) and diversity (*bio*) with the rhythmic phase space.

associated with the x-axis. The diversity controls the number of eras of the genetic algorithm. The rhythm ([0, 27]) is associated with the y-axis. The mappings of the parameters with the melodic and rhythmic phase spaces are shown on Figure 3, on the right.

The coordinates of the points of the red curve, sent to the Nomad as a trajectory, and of the blue curve, the path traversed by Nomad and observed by the OmniEye, are linked with the *mel*, *bio*, *rhy* and *oct* scrolls in the Parameter Control area. The *mel* and *oct* values are used as parameters in the fitness function [9, 10]. The *bio* scroll bar allows for interference in the duration of the genetic cycle, modifying the reproduction time. The *bio* value determines the slice of time necessary to apply the genetics operators, such as crossover and mutation, and may also be interpreted as the reproduction time for each generation, affecting the *diversity* of the population. The *rhy* value is used in the MIDI cycle and it interferes directly in the rhythm of the music making the rhythm become faster or slower. The points of the other curves are used in the Performance Control.

2.2.1 The Evolutionary Sound Process

In JaVOX, the MIDI protocol representation was used to code a musical genotype. In the evolutionary sound process, the individuals of the population are defined as groups of four voices, or notes. These voices are initially random generated within the interval [0, 127] with each value representing a MIDI event. In each generation, 30 groups are produced.

Two cycles are integrated in the evolutionary sound process. The *reproduction cycle* is the evolving process that generates chords using genetic operators and selecting individuals. The parameters of the reproduction cycle are extracted from the red curve, sent as a trajectory to Nomad, and from the blue curve, the tracked path. In the *MIDI cycle* the interface looks for notes to be played. When a chord is selected, the program puts it in a critical area that is continually being verified by the interface. These notes are played until the next group is selected, considering the status of each voice in the Parameter Control, that are set according the distance among the robots (Section 2.2.2). Figure 4 depicts the reproduction cycle and the MIDI cycle.

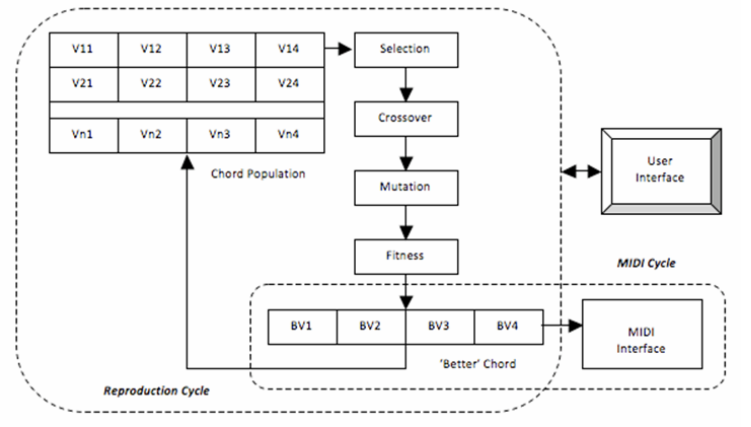


Fig. 4. The reproduction cycle and the MIDI cycle in the evolutionary process for the production of sound

The musical fitness for each chord, described in [9, 10], is a combination of three partial fitness functions: *melody, harmony and vocal range*, each resulting in a numerical value.

$$\text{Musical Fitness} = \text{Melodic Fitness} + \text{Harmonic Fitness} + \text{Vocal Range Fitness} \quad (1)$$

An analogy can be made of each individual of the population with a choir of four voices (or orchestra of four instruments). At each generation of the process a new sonority is created by applying a fitness criteria considering a melodic line (*mel*) and a voice range (*oct*). The choir with the highest fitness is selected and played as a new MIDI event, the duration of the evolutionary cycle (*bio*) and music meter (*rhy*) is taken into account. Based on the order of consonance of musical intervals, the notion of approximating a sequence of notes to its harmonically compatible note, or tonal center, is used. This sequence produces a sound resembling a chord cadence or a fast counterpoint of note blocks.

2.2.2 Music Real Time Performance Control

The Performance Control area offers other possibilities to control the sound production. For each of the four MIDI voices there are three controls: *solo, sequence* and

block. The Performance Control works as delay lines in which MIDI notes from previous generations are played again as solo, melodic patterns or chords. The Performance Control is also modified in real time by the robots. The relative position of the robots is used to select the solo, the sequence or block mode for each voice.

When the *solo control* is selected, the sound events are sent directly to the MIDI board producing a single sequence of MIDI events at each step of the genetic cycle. When the *sequence control* has been selected, MIDI events are played as note sequences, such as arpeggios. When the block control is selected, events are sent to the MIDI board as fast as possible, almost simultaneously, generating overlapped chords. A slider in the GUI controls the number of notes sent to the MIDI board. This control may only be modified by the user, as well as the rhythm and the pitch controls, affecting only the musical performance.

A set of performance rules was used to combine the collective behavior of the robots with sound production in JaVOX. Table 1 describes the voice control considering the proximity between the mobile robots and the solo, sequence and block processes, of the Performance Control.

Table 1. Performance Rules: Proximity and Performance Control

Distance between the Mobile Robots				
Rule	Distance (m)	Solo	Sequence	Block
1	>0,5	X		
2	0,4 < D < 0,5	X	X	
3	0,2 < D < 0,4		X	
4	D < 0,2			X

2.3 Robotic Control

Once a curve is drawn in the GUI, consecutive points of this curve are transmitted to the supervisor module TrajeCt (for trajectory control) so that the Nomad can navigate along the predetermined path. The applied algorithm is based on [15], where allowable navigation paths are defined as a sequence of straight lines between the given pathpoints, by controlling the robot heading. The heading change, at each pathpoint (between consecutive segments), may vary in a ±90° range and the distance between the actual robot position. The path is to be minimized in all cases. The supervisor system, therefore, receives the points from JaVOX and sends the movement commands to the robot. Other mobile robots move in the arena; when there is a collision, the other robot moves away. The flow of information departs and returns to JaVOX to produce the sonification process. The interaction between the path covered by the Nomad and the free navigation of other robots generates a collective robotic behavior.

3 Experiments

Since AURAL is also a platform for experiments in mobile robotics, three types of robots were incorporated to AURAL: Nomad, Pioneer, and Create. Nomad and Pioneer act as master robots, in different contexts (Pioneer was incorporated to a web version of AURAL). All the experiments described in this paper were performed with

Nomad and Create robots. Three runs of the system with two robots are described to illustrate the methodology applied in the experiments. Three different trajectories, named here as Circle, Eight and Arcs, were sent to Nomad, depicted in Figure 5, left column.

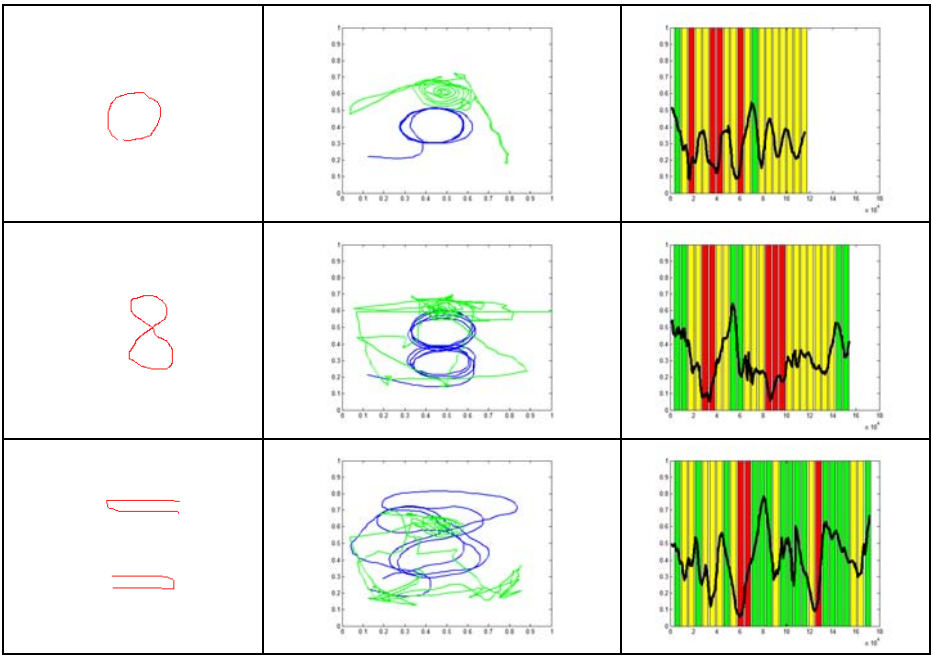


Fig. 5. Graphical results of three experiments

In the center column, the observed trajectory of the Nomad is depicted in blue and the observed trajectory of the Create robot, in green. The right column shows, on a time line the occurrences of the solo (green bars) in milliseconds (x-axis), sequence (yellow bars) and block events (red bars). The arena was scaled to be 1x1 m (y-axis). The black curve shows on a time line the variation of the distance between the robots.

Each row in Figure 5 is associated with an experiment. In Experiment 1 (row #1), the black curve (right column) shows that the Create trajectory was closer to the trajectory of the Nomad, although it produced less block events in real time. The distance between Create and Nomad was higher in Experiment 3 (row #3, right column). The maximum distance in Experiment 1 was around 0.5 m, in Experiment 2 it was around 0.65 m and in Experiment 3, around 0.8 m. Comparing the distance variation among the bar charts, it is possible to verify that changes on Performance Control in real time comply to the performance rules established in Table 1. Experiment 1 was confined to a short range of distance around the Nomad. Experiment 3 generated more solo events (green bars) than Experiments 1 and 2, due to the value of distances that in Experiment 3 was higher.

The next two sub-sections detail the relationship between the behavior of the robots, the evaluation of musical fitness and the sonification results.

3.1 Collective Behavior Affects Performance Control

According to Section 2.2, the trajectories sent to Nomad robot control melodic *(mel)* and octave range (*oct)* parameters and the observed trajectories control the parameters rhythm (*rhy*) and diversity (*bio*). Red curves change fitness evaluation of pitch material (MIDI note numbers) and observed blue curves change duration of the notes played by the system and also the duration of genetic cycles. In this way, original JaVox system produces MIDI notes and Nomad robot provides the rhythmic structure for them.

In real time each robot is related to one performance voice. Nomad is always associated with Voice 1 and the other Create robots are associated with the other voices. The results presented in Figure 6 refer to an experiment with Nomad and three Create robots, moving in the arena according to its area sweeping algorithm. In short, the robot initially performs a spiral. When it meets an obstacle, it turns left or right and moves straight, looking for a free way to avoid the obstacle.

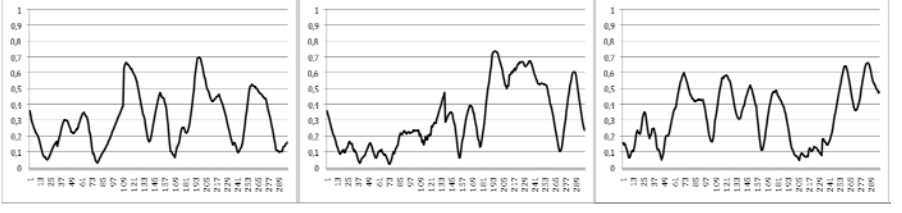


Fig. 6. From left to right the distance between the Nomad and Create robots 1, 2, and 3, while Nomad performed the arcs trajectory

The integration between guided behavior of the Nomad and the sweeping mode of the Creates produces a collective behavior resulting in Performance Control changes (Table 1). In terms of sonification, “circle” and “eight” trajectories produces more repetitive patterns and “arcs” produces more rhythmic variations, illustrating how melodies are produced by the AURAL system, integrating JaVOX evolutionary engine with the trajectories produced by the collective behavior of the robots. The movement of the robots associated with the performance rules of Table 1 introduces an element of pseudo musical improvisation in the system. It is important to note that the Block, Sequence and Solo modes are not exclusive; the three modes may be simultaneously assigned for each voice. The MIDI event associated with the voice is triggered off according to the selected mode, in the order in which it happened. According to the expectation, the occurrence of Solo events tends to decrease as the number of robots increases. The number of Solo events was lower in this series for all trajectories. The highest number was of Block events and occurred during the performance of the Circle trajectory.

The AURAL was presented as an installation at an Art Gallery, where visitors could draw curves in the JaVoX GUI and transmit it as trajectories to the Nomad. The visitors observed the interaction among the robots and the sound output produced by the robots exploring the associated conceptual sound space. On the last day of the exhibition, a dancer, three musicians and the AURAL system itself, with the Nomad

and three Create robots, performed an interactive concert. MIDI files produced by the AURAL system were used as a basic material for generating instrumental compositions. A piece titled “Robotic Variations” for Piano, Marimba and Electronics (computer and robots) was composed and performed at the AURAL installation. A choreography was designed so that a Create robot with a red panel left the room and was substituted by a dancer with a red hat. Her position was tracked by the OmniEye through the red hat and interfered in the performance of the sound. Figure 7 shows pictures of the musicians and the dancer during the performance. Scenes were broadcasted on television under the title "Robots compose music for man” [16].

Fig. 7. From the left to the right, the musicians in a rehearsal, during the performance and the dancer, while substituting a robot, tracked by the red hat

4 Conclusion

All these experiments beared aspects to consider such as the area of the room and the number of robots when adopting robotic collective behavior as a strategy for automatic composition. When the number of robots increases, the distribution of the events Block, Sequence and Solo tends to be uniform. Currently, the collected material is being studied to determine the events that contribute for the generation of interesting sound sequences, in order to teach the system to promote those events.

Acknowledgements

Special thanks to Flavio Kodama, who patiently implemented features for events recording in AURAL. This research work is supported by the Foundation for the Research in São Paulo State (FAPESP) process 05/56186-9. Manzolli is supported by the Brazilian Agency CNPq.

References

1. Todd, P.M., Werner, G.M.: Frankensteinian Methods for Evolutionary Music Composition. In: Griffith, N., Todd, P.M. (eds.) Musical Networks: Parallel Distributed Perception and Performance, pp. 313–340. MIT Press, Cambridge (1999)
2. Biles, J.: GenJam in Perspective: A Tentative Taxonomy for GA Music and Art Systems. Leonardo 36(1), 43–45 (2003)

3. Yee-King, M.: An Automated Music Improviser Using a Genetic Algorithm Driven Synthesis Engine. In: Giacobini, M., et al. (eds.) EvoWorkshops 2007. LNCS, vol. 4448, pp. 567–576. Springer, Heidelberg (2007)
4. Manzolli, J., Verschure, P.F.M.J.: Roboser: a Real world Musical Composition System. Computer Music Journal 3, 5–74 (2005)
5. Wassermann, K.C., Eng, K., Verschure, P.F.M.J., Manzolli, J.: Live Soundscape Composition Based on Synthetic Emotions. IEEE Multimedia, 82–90 (2003)
6. Murray, J., Wermter, S., Erwin, H.: Auditory robotic tracking of sound sources using hybrid cross-correlation and recurrent networks. In: IROS 2005 - International Conference on Intelligent Robots and Systems, pp. 3554–3559 (2005)
7. Moroni, A.S., Manzolli, J., Von Zuben, F.: ArTbitrating JaVOX: Evolution Applied to Visual and Sound Composition. In: Ibero-American Symposium in Computer Graphics, pp. 9–108 (2006)
8. Moroni, A., Von Zuben, F.J., Manzolli, J.: ArTbitration: Human-Machine Interaction in Artistic Domains. Leonardo 35(2), 185–188 (2002)
9. Moroni, A., Manzolli, J., Von Zuben, F.J., Gudwin, R.: Vox Populi: An Interactive Evolutionary System for Algorithmic Music Composition. Leonardo Music Journal 10, 49–54 (2000)
10. Moroni, A., Manzolli, J., Von Zuben, F.J., Gudwin, R.: Vox Populi: Evolutionary Computation for Music Evolution. In: Bentley, P., Corne, D. (eds.) Creative Evolutionary Systems, pp. 205–221. Morgan Kaufmann, San Francisco (2002)
11. http://nomadic.sourceforge.net/production/n200/
12. http://store.irobot.com/shop/index.jsp?categoryId=3311368
13. Moroni, A., Cunha, S., Ramos, J., Cunha, S., Manzolli, J.: Sonifying Robotic Trajectories with a Spherical Omnidirectional Vision System in the AURAL Environment. In: Workshop on Omnidirectional Robot Vision in the International Conference on Simulation, Modeling and Programming for Autonomous Robots (SIMPAR 2008), Venice, Italy (2008)
14. http://isa.umh.es/pfc/rmvision/opencvdocs/appPage/CAMShift/CAMShift.htm
15. Azinheira, J.R., Paiva, E.C., Ramos, J.G., Bueno, S.S.: Mission path following for an autonomous unmanned airship. In: Proc. IEEE International Conference on Robotics and Automation, San Francisco, pp. 147–164 (2000)
16. http://jornalnacional.globo.com/Telejornais/JN/0,,MUL1052269-10406,00-ROBOS+COMPOEM+MUSICA+PARA+O+HOMEM+NA+UNICAMP.html

Musical Composer Identification through Probabilistic and Feedforward Neural Networks

Maximos A. Kaliakatsos-Papakostas, Michael G. Epitropakis, and Michael N. Vrahatis

Computational Intelligence Laboratory (CI Lab), Department of Mathematics,
University of Patras Artificial Intelligence Research Center (UPAIRC), University of Patras,
GR-26110 Patras, Greece
{maxk,mikeagn,vrahatis}@math.upatras.gr

Abstract. During the last decade many efforts for music information retrieval have been made utilizing Computational Intelligence methods. Here, we examine the information capacity of the Dodecaphonic Trace Vector for composer classification and identification. To this end, we utilize Probabilistic Neural Networks for the construction of a "similarity matrix" of different composers and analyze the Dodecaphonic Trace Vector's ability to identify a composer through trained Feedforward Neural Networks. The training procedure is based on classical gradient-based methods as well as on the Differential Evolution algorithm. An experimental analysis on the pieces of seven classical composers is presented to gain insight about the most important strengths and weaknesses of the aforementioned approach.

1 Introduction

The common approach to tackle the composer identification and classification problem is through music theory and involves symbolic or score element analysis, which is done by musicians or musicologists, such as musical motif recognition, note durations and musical interval analysis. The main domain in computer music data extraction is based on statistical refinement through the wave forms of musical pieces [1]. In the literature, significant results have been demonstrated for musical genre recognition [2,3], key signature [4,5], chord identification [5,6] and composer identification [7] among others. A vital question would be: which symbolic features of a musical score embody the required information for computer based composer identification?

Data extraction through score analysis has been proven to be useful for chordal analysis [8], epoch classification of musical pieces [9] as well as composer identification [1,10,11] among others.

In the paper at hand, we study the information capacity of a simple and compact score-based data extraction technique, the *Dodecaphonic Trace Vector* (DTV), for the Musical Composer Identification task (MCI). The DTV roughly represents densities of degrees in a diatonic major scale in a musical piece and is analogous to the *Pitch Chroma Profile* proposed in [6]. To this end, based on the DTV we examine the composer identification task, by firstly discovering similarities between composers through supervised classifiers, namely the Probabilistic Neural Networks, and secondly investigating the DTV representation identification capabilities through Feedforward Neural Networks.

C. Di Chio et al. (Eds.): EvoApplications 2010, Part II, LNCS 6025, pp. 411–420, 2010.

The rest of the paper is organized as follows. In Section 2, the dataset acquisition and the tools that have been applied to extract certain symbolic features of a score are described. In Section 3, we briefly demonstrate the probabilistic and feedforward neural networks, while Section 4 is devoted to the presentation of the methodology and the produced experimental results. The paper ends with concluding remarks and some pointers for future work.

2 Data Set and Data Extraction

This section is devoted to describe the methods and techniques utilized for data acquisition and refinement to capture the desired information from musical pieces.

To this end, we have collected a dataset consisting of 350 musical pieces, composed by seven classical music composers. Specifically, musical pieces of the composers Bach, Beethoven, Brahms, Chopin, Handel, Haydn and Mozart, have been collected in MIDI format from [12]. Fifty musical works have been collected from each composer. To comply with constrains regarding composition styles for different musical instruments, an effort has been made so that most of the works collected by each composer were already transcribed for piano and correspond to an almost uniform collection of musical forms. Furthermore, in order to formulate a scale-tolerant collection, an almost equal number of major scale and minor scale pieces have been included.

To process each work, a conversion to a more understandable file format had to be made. Hence, we have incorporated the MSQ tool to transform the MIDI file format to a simple text file format [13]. MSQ converts each note to a text symbol preserving information about duration, time onset, pitch and velocity. Through the above encoding each Pitch Height can be described using an integer, but the information on the identity of unison notes is not maintained, which has been named *enharmonic equivalence* [5,14], e.g. whether a number corresponds to $C\sharp$ or $D\flat$.

Here, we investigate whether a compact information index such as the *Dodecaphonic Trace Vector*, which is briefly described in the following paragraph, may incorporate sufficient information from a musical piece, to tackle the MCI task.

Dodecaphonic Trace Vector. A musical piece is like a journey, it begins and ends following a certain path. This path could be based on notes of either a certain scale (tonal music), or more than one scales, depending on the form of each piece and its composer's personal style. A collection of traces of this path can be created with the *Dodecaphonic Trace Vector* described below.

We consider a 12-dimensional vector, the *Chroma Profile* [15] vector of a musical piece denoted by $CP = (CP(1), CP(2), \ldots, CP(12))$, the elements of which can be defined by the following equation:

$$CP\big(n \bmod (12) + 1\big) = \sum_{n \bmod (12) \in M} 1,$$

where n denotes a note in the musical piece M. This equation simply states that the first element, $CP(1)$, is the summation of all the notes n that satisfy the equation $n \bmod (12) \equiv 0$, which has been characterized as *octave equivalence* [14]. In our example

$CP(1)$ is the summation of all Cs, while $CP(2)$ is the summation of all $C\sharp$ or $D\flat$ in any octave.

The norm of the CP vector of a musical piece depends on its length. A very short piece is expected to have less notes than a very long one, under similar circumstances, i.e. both being in the same tempo and composed by the same composer. To extract useful data concerning the composer out of the CP vector, regardless the length of the piece, a normalization has to be done. The normalized vector $N = (N(1), N(2), \ldots, N(12))$ can be defined by the following equation:

$$N(i) = \frac{CP(i)}{\max\limits_{1 \leqslant j \leqslant 12} CP(j)}, \qquad \text{for} \quad i \in \{1, 2, \ldots, 12\}.$$

Finally, to follow the traces of the melodies and the harmony (chord changes) of a composer, the utilized methods should be *key tolerant*, in a sense that the key of the composed piece is not essential. What is considered to be important information, is the way that a composer manipulates the *degrees* of the major or minor scale in his/hers compositions and the deriving expansions occurring on scale changes through out the piece. A procedure to capture that information of a collection of musical pieces is, first, to transpose all these pieces in the key of C major and then to calculate their normalized N vector. Additionally, the minor scale pieces were transposed in the key of A minor. Thus, the *Dodecaphonic Trace Vector* is defined by the normalized vectors of musical pieces transposed to the key of C major or A minor.

3 Classification Methods Tested

Our methodology is based on two simple steps and provides insights on the uniqueness of the DTV in the compositions of each composer. Firstly, a classification is initiated for the construction of a similarity matrix between the composers using a Probabilistic Neural Network and secondly, a Feedforward Neural Network is utilized to identity each composer from another. It has to be noted that the experimental results should be discussed with musical experts to further verify the uniqueness of the DTV for each composer.

For completeness purposes let us briefly describe the classification methods used for the composer identification task.

Probabilistic Neural Networks. Probabilistic Neural Networks (PNNs) introduced by Sprecht in 1990 [16] as a new neural network structure. PNNs are utilized to classify objects in a predetermined number of classes. PNNs are based on kernel discriminant analysis and incorporate the Bayes decision rule and a non-parametric density function [17]. A PNN's structure consists of four layers, the *input*, the *pattern*, the *summation* and the *output* layer [16,18]. To this end, a pattern vector, $x \in \mathbb{R}^p$ is applied to the p input neurons and propagates to the pattern layer. The pattern layer is fully interconnected with the input layer, organized in K groups, where K is the number of classes present in the data set. Each group of neurons in the pattern layer consists of N_k neurons, where N_k is the number of training vectors that belongs to class k where

$k = 1, 2, \ldots, K$. Hence, the i-th neuron in the k-th group of the pattern layer calculates its output utilizing a Gaussian kernel function defined by the following equation:

$$f_{ik}(X) = \frac{1}{(2\pi)^{p/2}|\Sigma_k|^{1/2}} \exp\left(-\frac{1}{2}(X - X_{ik})^\top \Sigma_k^{-1}(X - X_{ik})\right),$$

where $X_{ik} \in \mathbb{R}^p$ defines the center of the kernel and Σ_k is the matrix of smoothing parameters of the kernel function. Furthermore, the summation layer consists of K neurons and estimates the conditional probabilities of each class by,

$$G_k(X) = \sum_{i=1}^{N_k} \pi_k f_{ik}(X), \qquad k \in \{1, 2, \ldots, K\},$$

where π_k is the prior probability of each class k ($\sum_{k=1}^{K} \pi_k = 1$). Thus, a new pattern vector X, not included in the training set, is classified to the class which produces the maximum output of its summation neurons. It has to be noted here that PNN's classification ability is strongly affected by the value of the *smoothing parameter*, σ, that, roughly speaking, determines the range of each class and consequently its performance [16,19]. Probabilistic Neural Networks have been recently utilized in musical applications [3].

Feedforward Neural Networks. Although, many different models of Artificial Neural Network have been proposed, the Feedforward Neural Networks (FNNs) are the most common and widely used. FNNs have been successfully utilized to tackle difficult real-world problems [20,21,22,23]. For completeness purposes let us briefly describe their structure and their supervised training methodology.

A Feedforward Neural Network consists of simple processing units called neurons, which are arranged in layers, the *input*, the *hidden* layers and the *output* layer. The neurons between successive layers are fully interconnected, and each interconnection corresponds to a *weight*. The training process is an incremental adaptation of the connection weights which acquires the knowledge of the problem at hand and stores it to the network's weights. More specifically, consider a FNN, net, whose l-th layer contains N_l neurons, where $l = 1, 2, \ldots, M$. When a pattern appears in its *input* layer the signal deriving by the multiplication of the input and the weights of the first layer neurons is summed and passed through a nonlinear activation function such as the well known logistic function $f(x) = (1 + e^{-x})^{-1}$. Those signals are then propagated to the next layer of neurons, l, multiplied by the weights of the next neuron layer, and the procedure continues until the signal reaches the output layer. Hence, the j-th layer, can be described by $net_j^l = \sum_{i=1}^{N_{l-1}} w_{ij}^{l-1,l} y_i^{l-1}$, where $w_{ij}^{l-1,l}$ is the weight from the i-th neuron at the $(l-1)$ layer to the j-th neuron of the l-th layer, while $y_j^l = f(net_j^l)$ is the activation function of the j-th neuron in the l-th layer. The training procedure can be accomplished by *minimizing* the *error function* $E(w)$ which can be defined by the *sum of the squared differences* between the actual output of the FNN, denoted by $y_{j,p}^M$, and the desired output, denoted by $d_{j,p}$, relative to the appeared input,

$$E(w) = \frac{1}{2} \sum_{p=1}^{P} \sum_{j=1}^{N_M} \left(y_{j,p}^M - d_{j,p}\right)^2 = \sum_{p=1}^{P} E_p(w),$$

where $w \in \mathbb{R}^n$ is the vector of the network weights and P represents the number of patterns used in the training data set.

Traditional methods for minimizing the above error function require the estimation of the partial derivatives of the error function with respect to each weight. These are called *gradient-based descent methods* and information concerning the gradient are estimated by back propagating the error from the output to the first layer neurons using the *delta rule*, a procedure thoroughly described in [20,24]. Furthermore, novel methods proposing stochastic evolution of the weight vector do not require the computation of the gradient of the error function [21,22,25]. These methods are beneficial against gradient-based methods not only in terms of *computational cost* but also their global optimization characteristics enhance the training procedure which is less likely to be trapped in local minima [21,22,23].

Here we incorporate, both classical and stochastic training methodologies to enhance the training procedure. Hence, we have used three well–known and widely used classical methods, namely the Levenberg-Marquardt (LM) [26,27], the Resilient Backpropagation (Rprop) [28], and the Broyden-Fletcher-Goldfarb-Shanno (BFGS) [29], as well as, we have used an evolutionary approach, namely the Differential Evolution method [25]. Due to space limitations, we are not in a position to briefly describe here the above methods. The interested reader is referred to [25,26,27,28,29].

4 Methodology and Experimental Results

This section demonstrates the experimental results and the experimental procedure utilized to tackle the Musical Composer Identification task. Hence, based on the DTV, the first step of our methodology is to calculate the similarity of the composers through the utilization of Probabilistic Neural Networks. To this end, we construct a similarity matrix for all musical composers. The construction of the similarity matrix is based on a simple method. To the best of our knowledge, this method is utilized for the first time and it is described below.

The set of the seven composers that we have collected, allows the construction of a square matrix with seven rows and seven columns. Each row and column represents one composer. Each entry of the matrix would be wished to produce an indication about the similarity between the composers corresponding to the respective row and column in dependence to the similarity along the rest of the composers. It has to be noted here that the diagonal elements of the aforementioned matrix, represents the similarity between a composer with himself, and would have no entry to be computed. A final and notable comment is that a consistent similarity matrix of this form should be symmetric.

We will demonstrate the PNNs role in the similarity matrix through a simple test case example. Let us consider that we want to construct the entries of the first column, which are related to Bach[1]. We utilize a PNN with six classes, one for each composer except Bach, and we train it by incorporating all pieces from the six composers, i.e. fifty pieces at the current data set. Afterwards, the fifty pieces of Bach are given to the PNN for classification. Thus, the utilized network will classify them to the classes of

[1] Since the first column is related to Bach, then the first row is related also to Bach.

the other six composers. In this way it can be assumed that the more pieces classified to a composer's class, the more similar his composition style is related to Bach's.

The construction of the first column of the similarity matrix is completed if we assign no value to the first row (representing the similarity between Bach and himself) and normalize the classified cases to probabilities per column by dividing every element of the first column by the total of the pieces classified (50). We observe that the sum of the column elements is one. To this end, Table 1 exhibits the similarity matrix obtained by the described procedure. The PNNs have been implemented in MATLAB© platform with $\sigma = 0.1$.

Table 1. Musical Composers Similarity Matrix through Probabilistic Neural Networks

	Bach	Beethoven	Brahms	Chopin	Handel	Haydn	Mozart
Bach	—	0.08	0.14	0.08	0.36	0.08	0.06
Beethoven	0.12	—	0.26	0.22	0.02	0.20	0.34
Brahms	0.16	0.10	—	0.30	0.06	0.08	0.08
Chopin	0.00	0.18	0.36	—	0.04	0.12	0.08
Handel	0.30	0.00	0.04	0.08	—	0.18	0.26
Haydn	0.26	0.22	0.08	0.26	0.22	—	0.18
Mozart	0.16	0.42	0.12	0.06	0.30	0.34	—

One can observe that the sum of the entries of each row is not one as well as the matrix is not symmetric. For example, it can been seen that 26% of the pieces composed by Bach were more similar to the composing traces of Haydn, while only 8% of Haydn's pieces fitted best Bach's composing traces. The asymmetric property of the Table 1 is discussed in the final section.

Other notable results have also been given through another classification task. For each composer we have constructed two sets, (a) the set of training pieces and (b) the set of test pieces. The former set consists of 35 and the latter of 15 pieces out of the 50 of each composer. We have used the 35 training pieces of all seven composers to create a PNN with seven classes. The remaining 15 pieces of a composer have been presented to the PNN and the percentage of the pieces classified to each of the seven composer, has been recorded in the composer's related column. The results shown in Table 2 are the average results of 100 classification tasks as described, each task has been accomplished with different, randomly selected, training and test sets. For example, in the first column we can see the percentage of the pieces of Bach classified to each composer, including Bach.

It should be commented here that the fact that the diagonal elements are greater than any other element in the respective row or column (with an exception made to the sixth diagonal element), provides evidence for the compositional information capacity of the Dodecaphonic Trace Vector. Furthermore, it has to be noted that the uniformly selected musical forms of the pieces of each composer is reflected by the high values of some non-diagonal elements compared to their corresponding diagonal element values.

FNN Identification Success Table. Each element of the Identification Success Table (IST) (Tables 3–6) shows the mean value of the successful composer identification

Table 2. PNN Verification Table

	Bach	Beethoven	Brahms	Chopin	Handel	Haydn	Mozart
Bach	64.6%	6.1%	4.5%	6.6%	14.3%	6.4%	5.5%
Beethoven	3.4%	33.3%	13.9%	11.1%	0.5%	13.6%	10.1%
Brahms	1.4%	4.7%	39.1%	15.0%	0.7%	5.0%	3.5%
Chopin	0.2%	12.4%	20.2%	53.0%	2.0%	5.6%	2.0%
Handel	13.4%	0.5%	4.7%	1.7%	66.1%	18.8%	10.6%
Haydn	6.1%	12.4%	6.5%	8.8%	6.6%	24.3%	11.6%
Mozart	10.9%	30.6%	11.1%	3.8%	9.8%	26.3%	56.7%

efforts of a FNN to distinguish whether a piece belongs to the composer of the respective row or column.

These are the mean values computed over 50 different training-testing identification tasks for each pair of composers. During each of the 50 identification tasks between two composers, of row A and column B, the network has been trained to respond 1 to the 35 training pattern of pieces composed by A and 0 to the 35 training pattern of pieces composed by B.

To test the network's performance we have used the 15 remaining pieces of A, for which we know the network should respond a value near 1 (desired output), and the 15 remaining pieces of B, for which the network should respond a value near 0 (desired output). When an unknown piece (a piece that does not belong to the training set) has been presented to the network, the network's response, x, has been considered as 1, if $x > 0.5$ and 0 in any other case.

The success rate of an identification task has been estimated as the percentage of the desired network outputs over all 30 test pieces, which is the sum of the right network responses divided by 30. For each one of the 50 identification tasks a different set of 35 training pieces for each composer has been used, which it also holds for the 15 pieces of each composer that have been used for testing. Moreover, a 5-fold cross-validation methodology has also been conducted with similar results.

The presented values on the IST correspond to the mean success value of the 50 identification tasks for each composer pair. A final comment about the training procedure would be that the sequence of the presented training patterns was randomized so that their targets would not include more than four continuously presented patterns targeted with 1.

Experimental results for the trained FNNs. In Tables 3–6, we exhibit the IST for FNNs that have been trained using the LM, Rprop and BFGS methods, respectively, as well as, with the Differential Evolution algorithm [25]. These results have been produced by the FNNs of the Neural Networks Toolbox of MATLAB [31] using the default toolbox parameters. Additionally, for the Differential Evolution algorithm we have utilized a population of twenty potential solutions and have evolved them for 500 generations with the DE/best/1/bin strategy by incorporating the default parameters for its control parameters i.e. $F = 0.5$, and $CR = 0.9$ [25].

Table 3. Identification Success Table for FNN trained with the LM method

	Bach	Beethoven	Brahms	Chopin	Handel	Haydn	Mozart
Bach	—	81.20%	76.40%	87.00%	67.13%	69.33%	82.20%
Beethoven	83.13%	—	66.46%	63.86%	84.06%	65.00%	69.80%
Brahms	75.80%	66.40%	—	54.06%	84.13%	76.20%	78.00%
Chopin	85.46%	64.86%	55.33%	—	85.73%	77.86%	77.00%
Handel	69.80%	84.53%	84.93%	85.33%	—	70.86%	78.06%
Haydn	71.40%	62.33%	76.73%	74.80%	69.13%	—	55.80%
Mozart	82.33%	71.06%	80.00%	79.06%	80.06%	53.60%	—

Table 4. Identification Success Table for FNN trained with the Rprop method

	Bach	Beethoven	Brahms	Chopin	Handel	Haydn	Mozart
Bach	—	82.86%	78.20%	85.00%	68.46%	72.06%	83.20%
Beethoven	83.13%	—	70.00%	65.20%	85.40%	65.60%	71.06%
Brahms	79.86%	66.60%	—	52.73%	85.46%	79.26%	80.60%
Chopin	87.06%	65.00%	54.93%	—	87.40%	76.80%	78.66%
Handel	69.13%	86.60%	86.06%	88.73%	—	71.60%	78.40%
Haydn	72.20%	65.67%	77.67%	76.33%	72.20%	—	55.06%
Mozart	83.53%	73.00%	80.06%	77.86%	80.60%	53.93%	—

Table 5. Identification Success Table for FNN trained with the BFGS method

	Bach	Beethoven	Brahms	Chopin	Handel	Haydn	Mozart
Bach	—	83.46%	79.26%	87.33%	67.60%	72.46%	82.73%
Beethoven	84.00%	—	66.20%	66.06%	86.86%	64.26%	72.40%
Brahms	77.46%	68.33%	—	53.66%	85.13%	78.86%	77.33%
Chopin	86.40%	66.53%	55.13%	—	88.53%	74.93%	79.26%
Handel	65.33%	85.60%	85.86%	88.86%	—	70.53%	79.53%
Haydn	70.13%	63.66%	78.80%	75.33%	67.13%	—	53.86%
Mozart	81.53%	72.46%	79.06%	78.53%	78.33%	54.60%	—

Table 6. Identification Success Table for FNN trained with DE/best/1/bin

	Bach	Beethoven	Brahms	Chopin	Handel	Haydn	Mozart
Bach	—	82.20%	75.13%	83.53%	69.00%	71.80%	81.80%
Beethoven	80.80%	—	62.60%	66.20%	80.86%	66.40%	69.80%
Brahms	75.86%	68.33%	—	50.06%	82.53%	79.40%	77.00%
Chopin	83.80%	64.66%	47.46%	—	84.53%	76.86%	78.73%
Handel	69.66%	86.46%	79.60%	86.40%	—	70.53%	80.93%
Haydn	72.80%	66.66%	77.93%	75.73%	72.00%	—	51.40%
Mozart	82.60%	71.80%	78.00%	75.26%	79.53%	52.26%	—

5 Discussion and Concluding Remarks

Through this work, evidence has been provided that the Dodecaphonic Trace Vector and, consequently, the Chroma Profile vector contain considerable capacity of information for the individuality of a composer. Moreover, experts with sufficient musical background need to amplify the findings of the work at hand and aid the musical analysis by educing related information.

The PNN similarity matrix in Table 1, as well as the IST table of the FNNs imply that as long as the similarity between two composers increases, the identification effectiveness between those two decreases. The latter comment is a sensible assumption, though we can see some exceptions, for example in Table 1 we can see that the similarity between Mozart and Handel is greater than the similarity between Mozart and Haydn, though in Table 3 the identification task is more precise for the first couple.

Future work would incorporate further analysis of the lack of symmetry of Table 1 and inquiry on the perspective for information extraction out of it. These pieces of information, combined with historical facts, could lead to an *influence* diagram between the composers that provides or validates the evidence on the existence of composers of *fundamental influence*.

More accurate results could be reached by refining further the musical structure through the symbolic analysis of musical scores using more sophisticated representation of musical items, or by incorporating pitch transitions and pitch durations. It is also intended to compare the DTV efficiency against other approaches. Moreover, instead of the PNN used here, any other supervised clustering tool, such as Support Vector Machines [32], could be used for the construction of the similarity matrix. The same holds for the identification success table. Finally, it is evident that a widely acceptable musical database should be created, in order to analyze and compare musical data extraction approaches.

References

1. Lebar, J., Chang, G., Yu, D.: Classifying musical score by composer: A machine learning approach, http://www.stanford.edu/class/cs229/projects2008.html
2. Tzanetakis, G., Cook, P.: Musical genre classification of audio signals. IEEE Transactions on Speech and Audio Processing 10(5), 293–302 (2002)
3. Mostafa, M.M., Billor, N.: Recognition of western style musical genres using machine learning techniques. Expert Systems with Applications 36(8), 11378–11389 (2009)
4. Peeters, G.: Musical key estimation of audio signal based on hidden markov modeling of chroma vectors. In: Proc. of the Int. Conf. on Digital Audio Effects DAFx 2006, pp. 127–131 (2006)
5. Gomez, E.: Tonal description of polyphonic audio for music content processing. INFORMS Journal on Computing 18(3), 294–304 (2006)
6. Fujishima, T.: Realtime chord recognition of musical sound: A system using common lisp music. In: Proc. ICMC, pp. 464–467 (1999)
7. Meador, S., Uhlig, K.: Content-based features in the composer identification problem, http://www.stanford.edu/class/cs229/projects2008.html
8. Pardo, B., Birmingham, W.P.: Algorithms for chordal analysis. Computer Music Journal 26(2), 27–49 (2002)

9. Beran, J.: Statistics in Musicology, 1st edn., July 2003. Chapman & Hall/CRC (2003)
10. Pollastri, G.S.E.: Classification of melodies by composer with hidden markov models (November 2001), http://www.computer.org/portal/web/csdl/doi/10.1109/WDM.2001.990162
11. Liu, Y.W.: Modeling music as markov chains: Composer identification, https://www-ccrma.stanford.edu/~jacobliu/254report.pdf
12. MIDI: The classical MIDI connection site map, http://www.classicalmidiconnection.com
13. Koepf, S., Haerpfer, B.: The MSQ project, http://www.aconnect.de/friends/msq2/msq.htm
14. Schell, D.: Optimality in musical melodies and harmonic progressions: The travelling musician. European Journal of Operational Research 140(2), 354–372 (2002)
15. Homei, M., Kazushi, N.: Extraction and analysis of Chroma-Profile from MIDI data. Joho Shori Gakkai Kenkyu Hokoku 2003(82(MUS-51)), 97–101 (2003)
16. Specht, D.F.: Probabilistic neural networks. Neural Networks 3(1), 109–118 (1990)
17. Hand, D.: Kernel Discriminant Analysis. John Wiley & Sons Ltd., Chichester (1982)
18. Georgiou, V.L., Pavlidis, N.G., Parsopoulos, K.E., Alevizos, P.D., Vrahatis, M.N.: New self-adaptive probabilistic neural networks in bioinformatic and medical tasks. International Journal on Artificial Intelligence Tools 15(3), 371–396 (2006)
19. Georgiou, V.L., Alevizos, P.D., Vrahatis, M.N.: Novel approaches to probabilistic neural networks through bagging and evolutionary estimating of prior probabilities. Neural Processing Letters 27(2), 153–162 (2008)
20. Haykin, S.S.: Neural networks and learning machines. Prentice Hall, Englewood Cliffs (2008)
21. Plagianakos, V.P., Vrahatis, M.N.: Parallel evolutionary training algorithms for 'hardware–friendly' neural networks. Natural Computing 1, 307–322 (2002)
22. Magoulas, G.D., Plagianakos, V.P., Vrahatis, M.N.: Neural network-based colonoscopic diagnosis using on-line learning and differential evolution. Applied Soft Computing 4, 369–379 (2004)
23. Plagianakos, V.P., Magoulas, G.D., Vrahatis, M.N.: Evolutionary training of hardware realizable multilayer perceptrons. Neural Computing and Application 15, 33–40 (2005)
24. Rumelhart, D.E., Hinton, G.E., Williams, R.J.: Learning internal representations by error propagation. In: Parallel distributed processing: explorations in the microstructure of cognition, foundations, vol. 1, pp. 318–362. MIT Press, Cambridge (1986)
25. Price, K., Storn, R.M., Lampinen, J.A.: Differential Evolution: A Practical Approach to Global Optimization. Natural Computing Series. Springer, New York (2005)
26. Marquardt, D.W.: An algorithm for Least-Squares estimation of nonlinear parameters. SIAM Journal on Applied Mathematics 11(2), 431–441 (1963)
27. Hagan, M.T., Menhaj, M.B.: Training feedforward networks with the Marquardt algorithm. IEEE Transactions on Neural Networks 5(6), 989–993 (1994)
28. Riedmiller, M., Braun, H.: A direct adaptive method for faster backpropagation learning: The Rprop algorithm. In: Proc. of the IEEE Intl. Conf. on Neural Networks, San Francisco, CA, pp. 586–591 (1993)
29. Nocedal, J., Wright, S.: Numerical Optimization. Springer, Heidelberg (1999)
30. Gill, P.E., Murray, W., Wright, M.H.: Practical Optimization. Academic Press, London (1982)
31. Demuth, H., Beale, M.: Neural Network Toolbox 6: For use with MATLAB: User's Guide. In: The Mathworks, Cochituate Place, 24 Prime Park Way, Natick, MA, USA (1992-2009)
32. Burges, C.J.C.: A tutorial on support vector machines for pattern recognition. Data Mining and Knowledge Discovery 2(2), 121–167 (1998)

Using an Evolutionary Algorithm to Discover Low CO_2 Tours within a Travelling Salesman Problem

Neil Urquhart, Cathy Scott, and Emma Hart

School Of Computing, Edinburgh Napier University, Edinburgh, UK
n.urquart@napier.ac.uk

Abstract. This paper examines the issues surrounding the effects of using vehicle emissions as the fitness criteria when solving routing problems using evolutionary techniques. The case-study examined is that of the Travelling Salesman Problem (TSP) based upon the road network within the City of Edinburgh, Scotland. A low cost path finding algorithm (A*) is used to build paths through the street network between delivery points. The EA is used to discover tours that utilise paths with low emissions characteristics. Two methods of estimating CO_2 emissions are examined; one that utilises a fuel consumption model and applies it to an estimated drive cycle and one that applies a simplistic CO_2 calculation model that focuses on average speeds over street sections. The results of these two metrics are compared with each other and with results obtained using a traditional distance metric.

1 Introduction and Motivation

Vehicle routing may be measured in terms of emissions produced as well as the more traditional raw distance metric. Vehicle emissions are dependant on the physical characteristics of the vehicle such as engine type and overall mass as well as factors related to the current driving activities such as acceleration and speed. Previous research in the area of optimisation has examined routing problems such as Travelling Salesman on the basis of a simplistic underlying geographical model, with a fitness function being based on the distance taken by a vehicle within a given solution. This paper will examine the effects of using emissions rather than distance covered as evaluation criterion. A more realistic underlying geographical model is required, rather than one that assumes a constant Euclidean distance between points.

In this paper we examine the Travelling Salesman Problem (TSP) based on real-world street data using CO_2 emissions as our evaluation criterion. The EA employed produces permutations of delivery points, the path taken through the street network between each point determined by the A* algorithm. It is feasible to use a CO_2 costing metric as the cost function of a search technique such as Dijkstra's algorithm, but the time taken to construct paths can be prohibitive. In this work we are using the EA to find paths created on the basis of distance that will contribute to a low CO_2 TSP tour.

C. Di Chio et al. (Eds.): EvoApplications 2010, Part II, LNCS 6025, pp. 421–430, 2010.

2 Previous Work

The TSP is a well known mathematical problem that involves construction a Hamiltonian Path within a graph. It is traditionally presented as finding the shortest route that allows a travelling salesman to visit each location. The TSP has been extensively investigated since the 1930s (for a brief history of TSP investigation see [1]). Because of its simple construction the TSP has been used for testing newly developed heuristics and algorithms.

The use of heuristic approaches to solve the TSP, such as Lin-Kernighan [2] are discussed in [3] [4] [5] [6]. The authors of [4] [5] [6] have carried out extensive investigations into the application of the Lin-Kernighan heuristic to very large instances of the TSP. The work of these authors has included the implementation of the Lin-Kernighan heuristic within a TSP solver entitled CONCORDE (Combinatorial Optimisation and Networked Combinatorial Optimisation Research and Development Environment). A set of benchmark TSP instances is maintained on the internet at the TSPLIB site [7]. Considerable research into solving the TSP using EAs has been undertaken [8] [9] [10] some investigation into the use of EAs to solve the Vehicle Routing Problem (VRP) has also been undertaken [11] [12] [13] [10].

3 Problem Description

3.1 The Geographical Data Source

In recent years sources of accurate geographical data have been made available online by vendors such as Google Maps, ViaMichelin and TeleAtlas. Such encapsulates the topology of streets and to a certain extent the layout and characteristics of road junctions. Within this paper the authors have used data from the Open Street Map (OSM) project (available from www.osm.org under the terms of the Creative Commons Attribution-Share Alike 2.0 licence). Data representing the street graph for the City of Edinburgh, Scotland has been downloaded and stored within a local MySQL database. For each road section the length and road class are stored. The authors have applied empirical knowledge and data presented in [14] to allocate a realistic average speed to each class of road (see table 1). At junctions the appropriate intersecting street details are stored along with attributes such as the existence of traffic signals or a roundabout. The model in use within this paper does not take into account waiting time at junctions or gradients.

The A* algorithm may be used to construct a path between any two locations within the OSM dataset, on the basis of shortest distance. A metric may subsequently be applied to calculate the cost associated with the route. In previous research this metric has often been based upon distance travelled. By examining the attributes of streets and junctions traversed within the route it is feasible to apply a CO_2 emissions metric to the route.

Table 1. Average speeds allocated to OSM link classes. For any link class not in the list, the default value (32kph) is used.

OSM Catagory	Speed (kph)
default	32
unclassified	32
secondary	36
residential	36
primary	38
tertiary	38
trunk	64
motorway-link	80
motorway	112

3.2 Estimating Vehicle Emissions

The emissions characteristics of a specific vehicle will depend on a range of factors, such as engine size, fuel type and vehicle mass. The actual driving activity and style also influences emissions through variables such as speed, acceleration and gradient. The calculation of estimated vehicle emissions is non-trivial exercise for which a number of approaches have been proposed. In this paper we examine two estimating techniques one based on [15] which attempts to calculate fuel consumption over each second of the journey and relate emissions produced to fuel used. The other (see section 3.4) is based on the work of [15] and assumes and average emissions rate over a road based on the probable speed of vehicles on that category of road.

3.3 Emissions Calculations Using a Fuel Consumption Model

A power based instantaneous fuel consumption model for road vehicles was proposed in [15]. The model may be described as follows:

$$dF = \alpha dt + \beta_1 R_T dx + [\beta_2 a R_1]_{a>0} \quad for R_T > 0 \tag{1}$$

$$= \alpha dt \qquad\qquad\qquad\qquad for R_T \leq 0 \tag{2}$$

where

dF = fuel (mL) consumed over distance dx (metres) during time dt(s)
α = idle fuel rate (mL/s)
β_1 = fuel consumption per unit of energy
β_2 = fuel consumption during positive acceleration
a = acceleration (m/s), negative when slowing down
R_t = total force required to drive the vehicle (kN) expressed as follows:

$$\begin{aligned} R_t &= R_D + R_l + R_G, \\ R_D &= b_1 + b_2 v^2, \\ R_l &= Ma/1000, \\ R_G &= 9.81M(G/100)/1000, \end{aligned} \tag{3}$$

where
v= speed (dx/dt) m/s
G = gradient (%) +ve or -ve
M = vehicle mass (kg)
b_1, b_2 = drag force function parameters

Appropriate values are provided in [15] to allow the model to be calibrated with respect to a Ford saloon car, these values were derived from observations made using an instrumented vehicle. This model is used by the authors of this paper to estimate fuel consumed within a given solution to the TSP. The litres fuel consumed are converted to Kg of co2 by multiplying by a conversion factor of 2.317 as specified in [16]. To estimate emissions from a route the model has to be utilised in conjunction with a drive cycle. A drive cycle being a series of data points representing the vehicles speed at given intervals (typically one second in many applications). The authors convert the OSM route into a series of interconnected drive cycles. Data for likely acceleration/deceleration curves is taken from [14]. The drive cycle is built as follows:

- for each street section establish a likely average speed based upon the OSM category using the values in table 1.
- create data points for that street within the drive cycle, speeds are not constant but deviate slightly in cycles derived from TRL data [14]
- evaluate the action required for each junction (change in speed, stop/restart etc) and plot appropriate data points (using acceleration/declaration curves from the TRL data [14])

The estimated changes in speed required at a junction are based upon the features present at the junction (traffic signals, roundabout etc) and the classification of the incoming and outgoing roads. It is assumed that at any junction having the attributes of a roundabout or traffic signals the vehicle will stop and restart.

3.4 Emissions Calculations Using a Simpler Model

The model described in 3.3 requires the construction of a drive cycle, which facilitates an attempt to model junctions, but is computationally expensive. As a contrast a simpler model based on the work of [17] is also tested. This is available as spreadsheet based model from the UK National Atmospheric Emissions Inventory [18].

The model proposed by [17] is an average speed model that is applied to each street section, with junctions being explicitly modelled. This negates the requirement to construct a drive cycle. The model may be described as follows:

$$em = (a + b.v + c.v^2 + d.v^e + f.ln(v) + g.v^3 + h/v + i/v^2 + j/v^3)$$

Where

em = the emissions produced as grams of CO2 per km
v = speed (kph)
a,b,c,d,e,f,g,h,i,h =coefficients that define the specific characteristics of the vehicle under consideration.

Values for a-j are provided within [17] for a range of vehicles. For the purposes of linking this model to the EA under consideration the spreadsheet model has been re-implemented as a set of Java classes. The model is applied to each street within the route, the length of street and class of street being available from the underlying OSM database. The average speed for that class of road is derived from observations within [14]. This is a computationally simpler model, but lacks the detail of that in section 3.3.

4 Experimental Method and Results

4.1 Problem Instances

Six problem instances were utilised, each requiring a visit to between 10 and 30 delivery points (addresses within the City of Edinburgh data set, see section 3.1) starting and ending at specific start point. Table 2 shows the respective distances between delivery points. The two data sets known as DS1 and DS2 were utilised their delivery points were chosen at random, two data sets referred to as Ring Road and City Centre were also utilised and two data sets big-20 and big-30 are used to explore longer runs. Ring Road uses delivery points located adjacent to the city's outer ring-road and City Centre used 15 delivery points located within the city centre area.

Table 2. A summary of the data sets used

	DS1	DS2	Ring Road	City Centre	Big-20	Big-30
Delivery points	10	10	10	15	20	30
Average dist (km)	8.29	12.9	18.52	1.45	11.65	9.181

4.2 The Evolutionary Algorithm Employed

The evolutionary algorithm uses a steady state population of 10 individuals. Each individual is a permutation of the points to be visited within the TSP instance. Within each generational cycle a parent is selected using tournament selection of size 2. A child is cloned from the parent and a mutation applied. The child is copied back into the population, by conducting another tournament and having the child replace the looser.

The fitness function builds a route through the street data visiting the points in the order they appear in the chromosome. To determine the path to be taken between each set of points the A* algorithm is utilised as described in section 3.1. The metric used to cost the route will be as set in section 4.3.

4.3 Experimental Method

The EA was executed on each of the four data sets with each of the three routing metrics were used to cost the TSP tours. The metrics used were raw distance and those outlined in sections 3.3 and 3.4. Within the results tables these metrics are referred to as dist, em and sEm respectively. Because of the non-deterministic nature of EAs all experiments were repeated 10 times and the results averaged.

Where results produced using dist and sEm as the evaluation criterion are being evaluated the em metric is applied to the final solution to produce a comparable emissions value.

4.4 Results

The results obtained by the methods outlined in section 4.3 are illustrated in table 5. The distances of tours do not increase significantly when the emissions metrics are utilised. Note that in two cases (City Centre and DS2) a slight decrease in the average distance is noted when optimising for emissions. The biggest increase in distance being the 10% increase noted on the Ring Road dataset. Table 3 shows the average emissions (kg/CO2) produced by solutions when optimising for distance and for emissions. When comparing the emissions produced by tours evaluated using the dist and em metrics all of the data sets show an improvement. Less of an improvement is noted when sEm is utilised, in the example of DS1, no improvement is noted. The improvements produced when using the larger datasets (Big-20 and Big-30) are less than the improvement with the smaller data sets.

Table 4 compares the emissions values produced by the individual results for each dataset and metric using t-tests. The T-test is used to establish where the difference in results is significant underlying changes. It suggests that the improvements noted in emissions between TSP tours evolved with the dist and em metrics are most significant in the smaller data sets. When optimising for low emissions some increase in distance may be expected as evidenced in table 5. The reduction in emissions is shown in table 3.

Table 3. The average emissions (kg/CO2) produced by solutions when optimising for distance and for emissions. Note that the final solution produced when using sEM or dist is recalculated using em to ensure that it is comparable with that produced using em.

metric	DS1	DS2	Ring Road	City Centre	big-20	big 30
dist	3.56	4.59	4.35	1.02	5.61	6.46
em	3.29	4.17	3.83	0.97	5.44	6.45
sEm	4.44	4.39	4.04	1.01	5.57	6.76

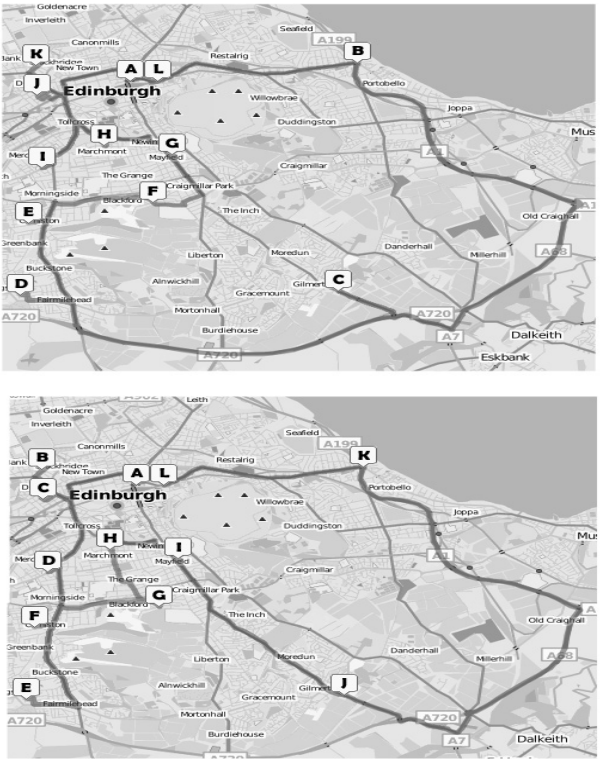


Fig. 1. Two typical tours produced with the DS1 problem instance, showing each delivery point (B-K) and the start/end point (A and L). The upper tour is optimised for minimum distance, the lower for minimum emissions. (Map ©2009 CloudMade - Map data CCBYSA 2009 OpenStreetMap.org contributors. *http* : *cloudmade.com/terms_conditions*)

Table 4. T-test results based on the raw emissions data presented in table 3 . The t-test compares the emissions values produced by the individual tours with the em and sEm metrics versus the dist metric.

metric	DS1	DS2	Ring Road	City Centre	big-20	big 30
em	0.0034	0.0001	0.0001	0.079	0.391	0.9637
sEm	0.445	0.1096	0.0079	0.7	0.1748	0.2944

Table 6 shows how the road categories used in solutions based on raw distance differs from those based on the emissions model. There do not appear to be any general trends, which suggest that a specific class of road is generally utilised more or less when costing the tours using a specific metric. But it should be noted that the Ring Road data set shows a significant switch to roads of the

Table 5. The average solution distance (km) when optimising for distance and for emissions

metric	DS1	DS2	Ring Road	City Centre	big-20	big 30
dist	60.2	84.9	70.6	13.1	107.6	114.43
em	60.8	88.33	79.8	13.3	113.3	125.08
sEm	60.7	83.9	77.8	12.8	110.9	117.15

Table 6. The effect of routing using the emissions metric by road class (expressed as a %)

	ds1			ds2			city centre		
	dist	em	sEm	dist	em	sEm	dist	em	sEm
primary	28.39	30.95	29.5	21	21.34	21.18	24.26	21.43	24.38
secondary	9.86	12.49	11.48	17.26	16.6	19.32	0.12	0.12	0.13
residential	29.38	29.85	26.34	27.52	24.49	27.93	16.03	16.89	15.99
unclassified	6.49	3.83	5.63	15.99	14.32	13.52	54.29	55.96	54.40
trunk	2.19	2.31	2.44	3.58	4.6	4.12	0.00	0.00	0.00
tertiary	23.69	20.57	24.62	14.65	18.67	13.92	5.30	5.59	5.10
	ring road			big-20			big-30		
	dist	em	sEm	dist	em	sEm	dist	em	sEm
primary	30.46	23.59	27.07	24.58	27.48	24.41	25.81	23.51	25.08
secondary	15.99	9.72	13.67	13.44	10.62	11.93	11.76	14.31	11.46
residential	13.73	10.72	11.12	28.74	26.51	28.29	26.81	27.26	28.47
unclassified	9.8	4.08	7.24	9.02	8.09	9.97	12.79	12.26	13.14
trunk	16.96	43.13	30.96	3.81	4.41	4.68	2.85	3.29	2.66
tertiary	13.06	8.76	9.95	20.41	22.88	20.72	19.97	19.37	19.20

trunk class when optimising for emissions and that all data sets show a slight reduction in their use of unclassified roads when optimising for emissions.

5 Conclusions and Future Work

The results presented suggest that within the context of the TSP there exist new challenges based upon the use of real-world data and emissions metrics. Of the metrics used sEm (see section 3.4) results are broadly similar to those obtained using raw distance, this might be expected given that both methods rely heavily on road weights. The em metric (see section 3.3) appears to reduce emissions further with relatively little increase in distance, the principle difference between the metrics being that em takes account of junctions. It should also be noticed that within the City-Centre data set, very little improvement in emissions is achieved. This data set has delivery points that are on average 1.45km apart, which provides few opportunities for alternative routes that provide a lower emissions factor. Overall it should be noted that the emissions reductions achieved by the EA have been based on the ordering of the points to be visited. The path

taken through the streets is determined by the A* algorithm. The improvement is due to the EA evolving tours that incorporate paths with a low emissions cost. There exists further potential for emissions reduction by applying the emissions metric at the path finding stage.

Both of the emissions metrics discussed in this paper have weaknesses, neither of them take into account all of the possible variables which may effect emissions. Further research to determine those variables that most affect emissions is required. Also required is to enhance the underlying map data to include factors such as congestion patterns and gradients. The inclusion of such additional factors (especially those with a dynamic aspect such as congestion) will further increase the complexity of the fitness function. A major challenge is to integrate such techniques into existing EA and heuristics. It is desirable to apply the techniques suggested in this paper to larger and more complex routing problems such as the Vehicle Routing Problem with Time Windows.

The successful use of A*, rather than a heuristic specially designed to build paths that meet low emission criterion suggests the potential to integrate with existing routing services and products. It is anticipated that demand for such services will increase as legislation forces communities and organisations to account for, and reduce emissions.

References

1. Applegate, D., Cook, W.: Concorde, combinatorial optimization and networked combinatorial optimization research and development environment (2002), http://www.math.princeton.edu/tsp/concorde.html
2. Lin, S., Kernighan, B.W.: An effective heuristic algorithm for the travelling salesman problem. Operations Research 21, 498–516 (1973)
3. Baraglia, R., Hidalgo, J.I., Perego, R.: A hybrid heuristic for the travelling salesman problem. IEEE Transactions on Evolutionary Computation 5, 613–622 (2001)
4. Cook, W., Applegate, D., Bixby, R., Chvatal, V.: Finding tours in the TSP. Technical Report 99885, Research Institute for Discrete Mathematics, University of Bonn (1999)
5. Cook, W.J., Cunningham, W.H., Pulleyblank, W.R., Schrijver, A.: The Travelling Salesman Problem. In: Combinatorial Optimization, pp. 241–271. John Wiley and Sons Ltd., Chichester (1998)
6. Applegate, D., Cook, W., Rohe, A.: Chained lin-kernighan for large travelling salesman problems (2000), http://www.citeseer.nj.nec.com/applegate99chained.html
7. Tsplib, a library of sample instances for the tsp (and related problems) from various sources and of various types (2002), http://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95/
8. Freisleben, B., Merz, P.: New genetic local search operators for the travelling salesman problem. In: Ebeling, W., Rechenberg, I., Voigt, H.-M., Schwefel, H.-P. (eds.) PPSN 1996. LNCS, vol. 1141, pp. 890–899. Springer, Heidelberg (1996)
9. Tamaki, H., Kita, H., Shimizu, N., Maekawa, K., Nishikawa, Y.: A comparison study of genetic codings for the travelling salesman problem. In: Proceedings of the 1st IEEE Conference on Evolutionary Computing (ICEC 1994). IEEE, Los Alamitos (1994)

10. Homaifar, A., Guan, S., Liepins, G.E.: A new approach on the travelling salesman problem by genetic algorithms. In: Forrest, S. (ed.) Proceedings of the 5th International Conference on Genetic Algorithms, pp. 460–466. Morgan Kaufmann, San Francisco (1993)
11. Thangiah, S.R.: A hybrid genetic algorithm, simulated annealing and tabu search heuristic for vehicle routing problems with time windows. Practical Handbook of Genetic Algorithms, Complex Coding Systems III, 347–381 (1999)
12. Blanton, J.L., Wainright, R.L.: Multiple vehicle routing with time and capacity constraints using genetic algorithms. In: Forrest, S. (ed.) Proceedings of the 5th International Conference on Genetic Algorithms, pp. 452–459. Morgan Kaufmann, San Francisco (1993)
13. Thangiah, S.R., Vinayaamoorthy, R., Gubbi, A.V.: Vehicle routing with time deadlines using genetic and local algorithms. In: Forrest, S. (ed.) Proceedings of the Fifth International Conference on Genetic Algorithms, pp. 506–515. Morgan Kaufmann, San Francisco (1993)
14. Green, J.M., Barlow, T.J.: Traffic management and air quality: realistic driving cycles for traffic management schemes. Technical Report TRL Report TRL596 (2004)
15. Akcelik, R., Biggs, D.C.: A discussion on the paper on fuel consumption modeling by post et al. Transportation Research Part B: Methodological 19, 529–533 (1985)
16. Act on co2 calculator: Data, methodology and assumptions paper. Department for Environment, Food and Rural Affairs (2008)
17. Barlow, T.J., Hickman, A.J., Boulter, P.: Exhaust emission factors 2001: database and emission factors. Technical Report PR/SE/230/00, Transport and Road Research Laboritory, Crowthrone, UK (2001)
18. Department for the Environment, Food and Rural Affairs. Results from the National Atmospheric Emissions Inventory (NAEI), http://www.naei.org.uk/

A Genetic Algorithm for the Traveling Salesman Problem with Pickup and Delivery Using Depot Removal and Insertion Moves

Volkan Çınar[1], Temel Öncan[2,⋆], and Haldun Süral[3]

[1] GEMPORT Port and Warehousing, Bursa, Turkey
cinarvolkan@gmail.com
[2] Department of Industrial Engineering,
Galatasaray University Ortaköy, İstanbul, 34357, Turkey
ytoncan@gsu.edu.tr
[3] Department of Industrial Engineering,
Middle East Technical University Ankara 06531, Turkey
sural@ie.metu.edu.tr

Abstract. In this work, we consider the Traveling Salesman Problem with Pickup and Delivery (TSPPD), which is an extension of the well-known NP-hard Traveling Salesman Problem. We propose a Genetic Algorithm (GA) based on a specially tailored tour improvement procedure for the TSPPD. Computational experiments are reported on the test instances taken from the literature. The experimental results suggest that the proposed GA yields a promising performance in terms of both accuracy and efficiency compared to existing algorithms in the literature.

Keywords: Genetic Algorithm, Traveling Salesman Problem, Pickup and Delivery.

1 Introduction

The Traveling Salesman Problem with Pickups and Deliveries (TSPPD) is defined on a complete graph $G = (V, E)$. The vertex set is defined as $V = \{0\} \cup V_d \cup V_p$, and it consists of a central depot 0, delivery customers V_d with requirement $d_i > 0$, and pickup customers V_p with requirement $p_i > 0$ for $i = 1, \ldots, n$. The edge (arc) set $E = V \times V$ denotes the edges connecting all pairs of vertices. Let c_{ij} denote the length of the arc (i, j) between customers i and j. Delivery customers receive goods from the central depot and pickup customers send goods to the depot. A single vehicle of a given capacity Q serves all customers without exceeding its own capacity. The vehicle starts and finishes its tour at the depot. The objective is to minimize the overall tour length.

When the capacity Q is less than either $\sum_{i=1}^{n} d_i$ or $\sum_{i=1}^{n} p_i$, then there is no feasible solution. The capacity constraints are redundant when $d_i = p_i = 0$ for $i = 1, \ldots, n$

⋆ Corresponding author.

C. Di Chio et al. (Eds.): EvoApplications 2010, Part II, LNCS 6025, pp. 431–440, 2010.

or Q is sufficiently large. For instance, when $Q > \sum_{i=1}^{n} d_i + \sum_{i=1}^{n} p_i$ holds the TSPPD reduces to the classical Traveling Salesman Problem (TSP) (see Gutin and Punnen [5]). The TSPPD is a generalization of the NP-hard TSP because the TSPPD considers the additional capacity constraints such that the vehicle load should remain less than Q along the tour. TSPPD has many real-life applications such as the distribution system of beverage industry where full bottles must be delivered and empty ones must be collected.

In this work we deal with the standard TSPPD in which the total pickup and delivery quantities are equal to the vehicle capacity, namely $\sum_{i=1}^{n} d_i = \sum_{i=1}^{n} p_i = Q$ holds. In other words, the vehicle starts the tour at the depot fully loaded with the total demand, performs all deliveries and pickups, and returns back to the depot again fully loaded with total pickup quantities. Any TSPPD instance can be transformed into the standard form by adding artificial customers located to the depot that makes the total pickup and delivery quantities equal to the vehicle capacity.

The TSPPD assumes that each customer is visited exactly once. Furthermore, the delivery and pickup quantities are served simultaneously for each location. If both a delivery and a pickup are located at the same location, the overall effect of their service on the vehicle's load is $|d_i - p_i|$. That is to say, the vehicle should first unload its delivery then should load its pickup. We can assume that the customer is a delivery (pickup) customer if its delivery (pickup) size is greater than or equal to its pickup (delivery) size. We also assume that for a customer i with equal pickup and delivery quantities, i.e. $d_i = p_i$, it is not possible to skip the visit of customer i, because the customer cannot use its own stock to meet the demand. Therefore, this customer must be visited although the visit does not change the vehicle's load. Note that if delivery and pickup quantities are equal for each location then the problem reduces to the TSP.

Let x_{ij} denote whether the vehicle immediately travels from customer i to customer j, y_i stand for the total load carried immediately after delivery to customer i, and z_i denote the total load carried immediately before pickup from customer i. According to the above mentioned assumptions the TSPPD can be formulated as follows (Süral and Bookbinder [12]).

$$\min \sum_{i=0}^{n} \sum_{j=0}^{n} c_{ij} x_{ij} \tag{1}$$

$$\text{subject to} \quad \sum_{i=0}^{n} x_{ij} = 1 \qquad j = 1, \ldots, n \tag{2}$$

$$\sum_{j=0}^{n} x_{ij} = 1 \qquad i = 1, \ldots, n \tag{3}$$

$$y_j - y_i + Q x_{ij} \leq Q - d_j \qquad i, j = 1, \ldots, n; i \neq j \tag{4}$$

$$z_i - z_j + Q x_{ij} \leq Q - p_i \qquad i, j = 1, \ldots, n; i \neq j \tag{5}$$

$$y_i + z_i \leq Q \qquad i = 1, \ldots, n \tag{6}$$

$$x_{ij} \in \{0, 1\}, y_i \geq 0, z_i \geq 0 \qquad i, j = 1, \ldots, n; i \neq j \tag{7}$$

The objective function is to minimize overall tour length. Constraints (2) and (3) are the assignment constraints. Constraints (4) and (5) are the subtour elimination constraints adapted from Miller et al. [8]. These two constraints ensure that the vehicle load is kept less than or equal to its capacity at every customer. Constraints (6) ensure that the delivery and pickup operations are performed without exceeding the vehicle capacity Q. Finally, nonnegativity and binary restrictions are given in (7).

The earliest work on the TSPPD is performed by Mosheiov [11] who formulated the problem and proposed the first TSP based heuristic algorithms. Anily and Mosheiov [1] proposed an efficient $O(n^2)$ heuristic with worst case performance of 2 based on the computation of shortest spanning trees on delivery and pickup customers. Gendreau et al. [4] have developed two different heuristics: One is based on transformation of a TSP tour into a feasible tour to the TSPPD and the other one is a Tabu Search (TS) approach employing a 2-exchange neighborhood (Lin [7]). Baldacci et al. [2] have proposed a two commodity flow formulation of the TSPPD and devised a branch and cut algorithm while Süral and Bookbinder [12] have solved the formulation given above by using a standard integer programming solver. Hernández-Perez and Salazar-González [6] have proposed two heuristics for the one-commodity pickup-and-delivery traveling salesman problem. These algorithms are also used to solve the TSPPD. The authors have put forward a greedy heuristic with a k-optimality criterion and an incomplete branch-and-cut algorithm. Recently, Zhao et al. [13] have developed a hybrid Genetic Algorithm (HGA) for the TSPPD. The authors have used a new pheromone based crossover operator which uses both local and global information to construct offspring. To the best of our knowledge there is no other work published on the TSPPD.

In this work, we propose a Genetic Algorithm (GA) for the TSPPD. It runs with a specially tailored depot removal-insertion based tour improvement procedure. We have observed that the GA with the tour improvement procedure yields promising results. The rest of this work is organized as follows. In Section 2 we present the GA and the depot removal-insertion based tour improvement procedure. The next section is where we present computational experiments. Finally, we conclude with Section 4.

2 An Evolutionary Approach for the TSPPD

2.1 A Genetic Algorithm

In our GA we have randomly generated an initial population of solutions. The initial Hamiltonian tours are generated considering all vertices except the depot. Then we try to insert the depot to the first feasible available location. Finally, 2-exchange moves are applied to improve the initial instances. The length of a

Hamiltonian tour has been considered as its fitness function. The path representation has been used as a chromosome representation of a Hamiltonian tour. In this representation, the Hamiltonian tour is represented by an ordered sequence of vertices.

The population size is carefully determined based on some preliminary experiments. We have observed that when the size of the population is smaller; the convergence of the GA becomes faster. Moreover, with the increasing population size one may encounter convergence problems. Considering the trade-off between the population size and the convergence of the GA, we have decided to employ a population size of 30 solutions. We have employed the Nearest Neighbor Crossover (NNX) operator to generate an offspring. NNX randomly selects a vertex as the starting point on the parent graph containing the edges from the parents' tours. A single offspring is generated by visiting the nearest unvisited vertex using only the edges from the parents. Hence, NNX concentrates on preserving edges from the parents. If a feasible tour cannot be found using the edges from the parents, NNX considers all arcs in the original data to find the next unvisited vertex. During the evolution process, we form a mating pool from the current population by replicating each chromosome twice. Then we select random pairs of the parents without replacement and we generate one offspring from each pair by using the NNX operator. As a result of this operation the population is doubled. We sort the parents and offspring according to their fitness values and we choose the best half of these chromosomes to the next generation. After these steps, we apply mutation operators to the new population. For that purpose, we perform a 3-exchange move to randomly selected 3 edges. Then, among all possible combinations including the original Hamiltonian tour, we choose the one with the lowest fitness function. The mutation operator is performed for each offspring included in the new population. The population has evolved through a number of generations until a stopping criterion is satisfied. After a careful experimentation and fine-tuning process, *number of generations* $:= 2 \times n$ is found to be a suitable choice for the algorithm. We stop the algorithm, when the average fitness is the same in two consecutive generations. For the details of EAs, see Michalewicz and David [10] and Michalewicz [9].

2.2 Tour Improvement Procedure

In his early paper Mosheiov [11] has shown that given a Hamiltonian tour $(i_1, i_2, \ldots, i_k, \ldots, i_n)$ covering all the pickup and delivery points but the depot, there exists at least one starting point i_k on this tour such that when the depot is inserted between i_k and i_{k+1} the resulting tour, $(i_1, i_2, \ldots, i_k, 0, i_{k+1}, \ldots, i_n)$ is feasible for the TSPPD. Using this result Mosheiov [11] has proposed a two stage Depot Insertion (DI) heuristic. In this heuristic first a Hamiltonian tour consisting of pickup and delivery points is found. Then a starting point i_k on this tour is found such that the depot is feasibly inserted right after it. Mosheiov [11] has noted that since the starting point that we will insert the depot is not necessarily unique, among all possible starting points, the one which yields the minimum tour length should be chosen.

One possible direction to improve the DI heuristic is the use of a local neighborhood search scheme. For instance, after the second stage of the DI heuristic one may employ the (feasible) arc exchange neighborhood scheme proposed by Gendreau et al. [4]. Another enhancement direction of the DI heuristic is to employ local search strategies in the first phase. After improving the Hamiltonian tour in the first phase, one can find at least one feasible starting point where the depot can feasibly be inserted in the new improved tour. The idea is illustrated as follows. Consider a feasible TSPPD tour $(0, 1, 2, 3, 4, 0)$ for a five vertex problem instance. When we apply 2-exchange operations on this TSPPD, assuming that we do not harm the feasibility, we obtain the following five solutions:

(i) $(0, 3, 2, 1, 4, 0)$ **(ii)** $(0, 1, 3, 2, 4, 0)$ **(iii)** $(0, 1, 2, 4, 3, 0)$
(iv) $(0, 2, 1, 3, 4, 0)$ **(v)** $(0, 1, 4, 3, 2, 0)$

As an alternative operation, first we propose to disconnect the depot from the original TSPPD tour $(0, 1, 2, 3, 4, 0)$ and we obtain the Hamiltonian tour $(1, 2, 3, 4, 1)$. Then when we apply 2-exchange on this tour, we obtain the following neighbors:

(vi) $(1, 2, 4, 3, 1)$ **(vii)** $(1, 3, 2, 4, 1)$

For each of these tours, including $(1, 2, 3, 4, 1)$, we have four alternative locations to reinsert the depot:

(viii) $(0, 1, 3, 4, 2, 0)$ **(xii)** $(0, 1, 4, 2, 3, 0)$ **(xvi)** $(0, 1, 4, 3, 2, 0)$
(ix) $(0, 1, 2, 4, 3, 0)$ **(xiii)** $(0, 1, 3, 2, 4, 0)$ **(xvii)** $(0, 2, 1, 4, 3, 0)$
(x) $(0, 2, 1, 3, 4, 0)$ **(xiv)** $(0, 2, 4, 1, 3, 0)$ **(xviii)** $(0, 3, 2, 1, 4, 0)$
(xi) $(0, 3, 1, 2, 4, 0)$ **(xv)** $(0, 2, 3, 1, 4, 0)$ **(xix)** $(0, 1, 2, 3, 4, 0)$

In total, we obtain 12 solutions. Observe that (i) and (xviii), (ii) and (xiii), (iii) and (ix), (iv) and (x), and (v) and (xvi) are the same tours. Henceforth, we can say that the depot removal, 2-exchange, and depot reinsertion operations include (at least) all the neighbor solutions obtained with 2-exchange operation applied to the original tour $(0, 1, 2, 3, 4, 0)$.

Fig.1 illustrates the depot removal-insertion based tour improvement operation. Black (white) vertices represent the deliveries (pickups). There are eight customers indicated by numbers on vertices and their demands are given in parentheses. Positive (negative) demands correspond to pickup (delivery) requests. The vehicle capacity Q is equal to 10. In the left hand side of Fig.1, the solution is $(0, 3, 2, 1, 4, 5, 6, 7, 8, 0)$ with the tour length 34.05 while in the right hand side the improved solution is $(0, 6, 5, 4, 3, 2, 1, 8, 7, 0)$ with the tour length 23.28. Observe that we have deleted edges $(3, 0), (0, 8), (7, 6), (1, 4)$ and added edges $(1, 8), (7, 0), (0, 6), (3, 4)$.

The proposed depot removal-insertion based tour improvement procedure is performed at the end of our GA for each Hamiltonian tour in the population. Then the best solution is reported as the final output of the GA algorithm.

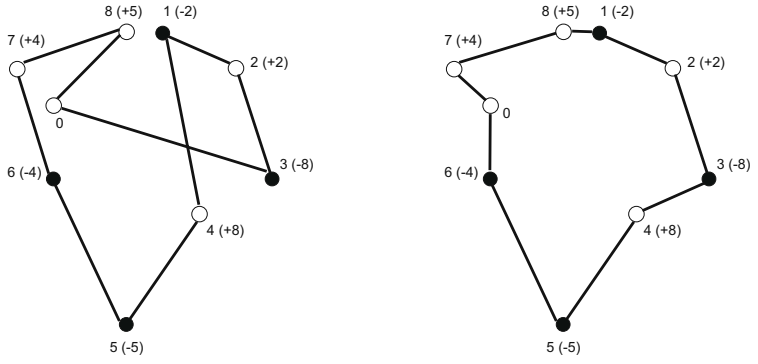


Fig. 1. Depot Removal-Insertion Based Tour Improvement Operation

3 Computational Experiments

The proposed GA is tested on standard instances taken from the literature. Our test bed contains two classes of instances generated by Gendreau et al. [4]. Gendreau et al. [4] have used a β parameter to indicate the percentage of demand allocated to pickups. More specifically, given the demand quantity d_i of each vertex i of a vehicle routing problem (VRP) test instance, the delivery quantity of that vertex is set to d_i and the pickup quantity p_i is determined according to the following rule:

$$p_i = \begin{cases} \lfloor (1-\beta)d_i \rfloor & \text{if } i \text{ is even} \\ \lfloor (1+\beta)d_i \rfloor & \text{if } i \text{ is odd} \end{cases} \qquad i = 1, \ldots, n$$

For each instance size, we set $\beta = 0.00, 0.05, 0.10, 0.20, \infty$. For $\beta = 0$ we have a TSP instance, and for $\beta = \infty$, the d_i and p_i values are uncorrelated.

The first class of instances generated by Gendreau et al. [4] consists of 26 test problems with customer sizes varying from 6 to 261. Test problems are derived from the symmetric VRP instances from the literature. The second class consists of randomly generated instances with $n = 25, 50, 75, 100, 150, 200$. d_i values are randomly chosen within the interval $[1, 100]$. For each pair of n and β values, 10 instances are generated. Note that for instances in the second class with $\beta = \infty$, d_i and p_i values are uncorrelated and generated uniformly random in $[1, 100]$.

Table 1. Best bounds obtained on the first class of instances

β	**TS**	**TBB**	**HGA**	**GA**
0.00	100.51	100.05	100.05	**100.046**
0.05	102.45	100.61	100.04	**100.028**
0.10	104.34	100.72	100.08	**100.038**
0.20	106.16	100.90	100.06	**100.058**
Average	103.37	100.57	100.06	**100.04**

Tables 1 and 2 summarize the results of the first class of test instances. Each cell of these tables stands for the average value obtained with 26 instances in the first class. In Tables 3 and 4 we present the results obtained with the instances in the second class. Each cell of Tables 3 and 4, indicates the average results of 10 instances for each pair of n and β values. In Tables 1 and 3, we present the results obtained with several upper bounding approaches. The values reported

Table 2. Average CPU times spent on the first class of instances

β	**TS**	**TBB**	**HGA**	**GA**
0.00	3.10	**0.93**	1.41	1.69
0.05	2.18	**0.87**	1.41	1.78
0.10	2.31	**1.07**	1.42	1.78
0.20	2.26	**1.02**	1.43	1.58
Average	2.46	**0.97**	1.42	1.71

Table 3. Best bounds obtained on the second class of instances

β	**n**	**TS**	**TBB**	**HGA**	**GA**
0.00	25	**100.00**	**100.00**	**100.00**	**100.00**
	50	100.20	**100.00**	**100.00**	**100.00**
	75	100.78	**100.00**	**100.00**	**100.00**
	100	101.27	100.10	100.10	**100.01**
	150	102.37	100.34	100.33	**100.10**
	200	103.13	100.35	100.36	**100.41**
0.05	25	107.36	102.07	**100.00**	**100.00**
	50	103.95	100.17	**100.00**	**100.00**
	75	110.13	100.83	100.12	**100.04**
	100	107.23	100.39	100.13	**100.02**
	150	108.72	100.35	100.26	**100.11**
	200	108.57	100.69	100.57	**100.39**
0.10	25	108.21	101.18	**100.00**	**100.00**
	50	106.34	100.47	**100.00**	**100.00**
	75	113.28	101.32	**100.15**	**100.15**
	100	111.53	100.50	**100.12**	100.15
	150	110.90	100.68	100.46	**100.12**
	200	111.31	100.76	100.72	**100.44**
0.20	25	107.28	102.59	**100.00**	100.05
	50	106.53	100.79	**100.00**	100.05
	75	114.25	101.77	100.12	**100.06**
	100	113.09	100.96	100.20	**100.12**
	150	111.69	101.02	100.60	**100.17**
	200	113.28	100.99	100.85	**100.46**
∞	25	105.64	101.32	**100.00**	**100.00**
	50	110.86	102.47	**100.00**	100.03
	75	111.86	100.91	100.13	**100.10**
	100	110.34	100.87	100.15	**100.13**
	150	112.85	100.63	100.39	**100.29**
	200	113.03	100.83	100.71	**100.46**
Average		108.20	100.85	100.22	**100.13**

Table 4. Average CPU times spent on the second class of instances

β	n	TS	TBB	HGA	GA
0.00	25	0.16	1.10	0.25	**0.04**
	50	0.75	1.20	0.51	**0.25**
	75	1.82	1.50	0.90	**0.39**
	100	3.24	1.60	2.25	**1.00**
	150	8.35	**2.70**	4.25	4.15
	200	15.27	**3.40**	6.93	7.32
0.05	25	0.18	1.00	0.24	**0.05**
	50	0.60	1.20	0.50	**0.25**
	75	1.32	1.10	0.89	**0.40**
	100	2.37	1.90	2.28	**0.90**
	150	5.46	**2.80**	4.15	3.49
	200	11.15	**3.00**	6.71	7.87
0.10	25	0.17	1.10	0.24	**0.05**
	50	0.63	1.10	0.51	**0.27**
	75	1.42	1.40	0.95	**0.39**
	100	2.60	2.30	2.25	**0.93**
	150	6.19	**3.00**	4.18	3.28
	200	11.93	**3.10**	6.92	7.98
0.20	25	0.20	1.10	0.24	**0.06**
	50	0.72	1.10	0.50	**0.28**
	75	1.44	1.50	0.88	**0.42**
	100	2.61	2.00	2.26	**0.91**
	150	6.01	**2.50**	4.30	3.27
	200	11.85	**3.50**	6.94	7.12
∞	25	0.18	1.00	0.25	**0.07**
	50	0.73	1.20	0.51	**0.30**
	75	1.42	1.20	0.92	**0.44**
	100	2.59	1.70	2.29	**0.94**
	150	5.78	**2.40**	4.22	3.62
	200	11.21	**3.60**	6.98	7.65
Average		3.95	**1.91**	2.51	2.14

Table 5. Scaled CPU times

Instance Set	TS	TBB	HGA	GA
First Class	2.46	0.97	**0.77**	1.05
Second Class	3.95	1.91	1.36	**1.32**
Average	3.21	1.44	**1.07**	1.19

are computed as $100 \times z_{UB}/z_{TSP}$, where $z_{UB}(z_{TSP})$ is the upper bound obtained with the corresponding algorithm (the optimum TSP solution value). In Tables 2 and 4, we give the average CPU times. In Tables 1-4, the first columns include β parameters. TS columns indicate the results obtained with the TS algorithm by Gendreau et al. [4]. TBB columns include the results reported with the truncated branch and bound algorithm by Hernández-Perez and Salazar-González [6]. HGA columns indicate the results of the HGA devised by Zhao et al. [13]. The last columns are for the results obtained with the GA. The last rows of the tables stand for the averages of the corresponding columns.

In order to compare the CPU time requirements of the algorithms we consider the performance evaluation and benchmarking approach proposed by Dongarra [3]. The author proposes to measure the power of a computer by its floating-point rate of execution in Mflops. Both the TS and TBB are run on an AMD 1.333 GHz PC with 649 Mflops. The HGA is tested on a Pentium 1.33 GHz PC with 352 Mflops. Our experiments were performed on an Intel Pentium 2.2 GHz PC with 400 Mflops. The estimated powers of these computers, in terms of Mflops, are taken from Dongarra [3]. We scale the average CPU times to the slowest computer namely, the computer on which the TS and TBB are run. Considering the CPU times reported in Table 5, we can say that the HGA is slightly better than the GA. However, for the instances in the second class, the GA is slightly better. Therefore we can conclude that the proposed GA yields a comparable efficiency to the HGA with a better accuracy.

Recall that the GA runs with the tour improvement procedure at the final step for each member in the population and outputs the best solution. For the sake of clarity, we should report that when the GA runs without the final tour improvement procedure we have obtained average bounds of 100.29 (100.68) for the first (second) class of instances. However, the bounds obtained with using the tour improvement procedure are 100.04 (100.13) for the first (second) class of instances. This shows us the power of our improvement that is specially tailored for the TSPPD.

4 Conclusion

In this work, we consider the TSPPD which is an extension of the well-known TSP. We have proposed a GA which includes specially tailored tour improvement procedure for the TSPPD. Computational experiments are reported on the standard test instances from the literature. According to the experimental results, we can say that the proposed GA performs at least as good as the recently proposed HGA in terms of both accuracy and efficiency.

Acknowledgments. The authors thank two anonymous referees for their valuable comments and suggestions. The second author acknowledges the support by Galatasaray University Research Foundation Grant 09.402.20.

References

1. Anily, S.E., Mosheiov, G.: Traveling Salesman Problem with Delivery and Backhauls. Operations Research Letters 16, 11–18 (1994)
2. Baldacci, R., Hadjiconstantinou, E., Mingozzi, A.: An Exact Algorithm for the Traveling Salesman Problem with Deliveries and Collections. Networks 42, 26–41 (2004)
3. Dongarra, J.J.: Performance of Various Computers Using Standard Linear Equations Software, Technical Report CS-89-85, University of Tennessee CS Dept. (2009)

4. Gendreau, M., Laporte, G., Vigo, D.: Heuristics for the Traveling Salesman Problem with Pickup and Delivery. Computers and Operations Research 26, 699–714 (1999)
5. Gutin, G., Punnen, A.P.: The Traveling Salesman Problem and Its Variations. Kluwer Academic Publishers, Dordrecht (2002)
6. Hernández-Perez, H., Salazar-González, J.-J.: Heuristics for the One-commodity Pickup and Delivery Traveling Salesman Problem. Transportation Science 38(2), 245–255 (2004)
7. Lin, S.: Computer Solutions of the Traveling Salesman Problem. Bell System Technical Journal 44, 2245–2269 (1965)
8. Miller, C., Tucker, A., Zemlin, R.: Integer Programming Formulation of Traveling Salesman Problem Formulations. Journal of the Association for Computing Machinery 7, 326–329 (1960)
9. Michalewicz, Z.: Genetic Algorithms + Data structures = Evolutions Programs, pp. 215–219. Springer, Heidelberg (1999)
10. Michalewicz, Z., David, F.: How To Solve It: Modern Heuristics, pp. 189–224. Springer, Heidelberg (2000)
11. Mosheiov, G.: Traveling Salesman Problem with Pickup and Delivery. European Journal of Operational Research 79, 299–310 (1994)
12. Süral, H., Bookbinder, J.: The Single-Vehicle Routing Problem with Unrestricted Backhauls. Networks 41(3), 127–136 (2003)
13. Zhao, F.-G., Sun, J.S., Li, S.J., Liu, W.-M.: A Hybrid Genetic Algorithm for the Traveling Salesman Problem with Pickup and Delivery. International Journal of Automation and Computing 6(1), 97–102 (2009)

Fast Approximation Heuristics for Multi-Objective Vehicle Routing Problems

Martin Josef Geiger

Helmut Schmidt University, University of the Federal Armed Forces Hamburg,
Logistics Management Department, Holstenhofweg 85, 22043 Hamburg, Germany
m.j.geiger@hsu.hh.de
http://logistik.hsu-hh.de/geiger

Abstract. The article describes an investigation of the use of fast approximation heuristics for multi-objective vehicle routing problems (MO-VRP). We first present a constructive heuristic based on the savings approach, which we generalize to fit the particular multi-objective nature of the problem. Then, an iterative phase based on local search improves the solutions towards the Pareto-front. Experimental investigations on benchmark instances taken from the literature show that the required computational effort for approximating such problems heavily depends on the underlying structures of the data sets. The insights gained in our study are particularly valuable when giving recommendations on how to solve a particular MO-VRP or even a particular MO-VRP instance, e. g. by means of a posteriori or interactive optimization approaches.

1 Introduction

Since the early development of operations research techniques for problems found in logistics, a considerable progress can be observed with solving challenging optimization models from that domain. Nowadays, (semi-)automated planning systems are available for a number of logistical applications. A prominent example can be found in vehicle routing, where a given set of transportation orders has to be served using a fleet of vehicles. The solution to such problems generally involves the assignment of orders to resources (vehicles), as well as the routing of the vehicles, and a common optimality criterion is found in the minimization of the total cost of transportation. Besides, complex side constraints have to be obeyed, such as time windows defining the availability of resources or expressing preferences of the customers. Often, such problems are referred to as 'rich' vehicle routing problems.

Practical vehicle routing problems not only involve numerous constraints, but also other criteria besides the cost minimization objective, e. g. the maximization of the provided service. Consequently, multi-objective problem formulations are proposed in the literature. The simultaneous consideration of conflicting objectives has some effects on the solution of vehicle routing problem formulations. Since not a single optimal solution exists that optimizes all criteria at once, a Pareto-set P containing equally Pareto-optimal alternatives must be found, and

C. Di Chio et al. (Eds.): EvoApplications 2010, Part II, LNCS 6025, pp. 441–450, 2010.

the choice of a most-preferred solution $x^* \in P$ must be made involving a human decision-maker.

Solution strategies for multi-objective vehicle routing problems can be organized in three ways.

1. *A priori* approaches first reduce the multi-objective formulation into a single-objective one by introducing a utility function or some other approach with which the preference of the decision maker can be expressed and measured.
2. *A posteriori* approaches on the other hand identify the Pareto-set P in an offline optimization phase, and then allow the decision maker to identify some most-preferred element $x^* \in P$.
3. *Interactive* approaches aim to combine the above mentioned concepts by allowing a gradual articulation of preferences, alternating between optimization and decision making phases until some satisfying alternative is found.

Obviously, both the optimization and the decision making phase are time-consuming processes, and time is, especially in operative planning problems, a limited resource. Therefore, any practical implementation of a solution concept must find the correct balance of each part. From that perspective, no general recommendation of which of the three concepts should be used can be given.

(Meta-)heuristics have become increasingly popular for solving vehicle routing problems [1,2,3,4]. More recently, multi-objective adaptations of such techniques have been proposed and applied to the VRP [5,6,7]. Interactive approaches are however still rather new and need to be studied further [8].

In the light of the explanations above, it becomes clear that more thorough investigations are needed with respect to two issues. (i) First, fast approximation approaches are needed, that allow the (approximation) solution of hard combinatorial optimization problems within little time. Such ideas generally contribute to the development of interactive approaches, that have to rely on decision support systems providing fast responses. (ii) Second, the cost of approximating multi-objective vehicle routing problems using heuristic search algorithms must be further studied, with 'cost' measuring the required computations effort for finding (local) optima. The results obtained from the investigation may then be used when trying to give recommendations on whether to opt for a priori, a posteriori, or interactive approaches in the particular problem domain. As we cannot expect to observe an algorithmic behavior independent from the underlying characteristics of the particular benchmark instances, structurally different data sets are used for the experiments.

2 Problem Statement

The vehicle routing problem under multiple objectives as tackled in this article is defined by a given set of customers, each of which must be served with a given number of goods from a depot by means of a vehicle. It therefore falls into the class of deterministic problems, an assumption which appears on a

short planning horizon to be feasible in many practical cases. A formal problem representation is possible using a graph $G = (V, E)$, consisting of vertices $v_i \in V$ representing customers i at given locations, and an interconnecting edge set E representing the route network between the nodes, and traveling along the edges of the graph results in a given travel time. For each customer i, time windows $[t_i^e, t_i^l]$ are given, which define the desired time of service. With respect to our multi-objective formulation of the problem, the earliest possible time of service t_i^e is interpreted as a constraint. The lasted desired time t_i^l however may be violated, but is penalized by means of an additional objective function as described below [9].

Additional side constraints of the problem are capacity constraints of the vehicles and their maximum total travel time $t^{\max}$, which easily can be represented by a (hard) time window at the depot node v_0 by $[t_0^e = 0, t_0^l = t^{\max}]$. We further require that all customers are serviced by exactly one vehicle, therefore avoiding split-deliveries.

A feasible solution to the problem defines routes and thus delivery schedules for each customer, respecting all given side constraints of the problem. Alternatives are evaluated with respect to two criteria: The minimization of the resulting cost and the maximization of the quality of the provided of service. While the first criterion is translated into the minimization of the total traveled distances, the second is represented by the total tardiness, thus providing a measure for the punctuality of the deliveries.

Besides the scope of this investigation, other criteria might be of relevance also, e. g. the minimization of the number of vehicles in use. On a short planning horizon however, the available fleet for transportation is often constant, and therefore is left aside here.

A considerable conflict among the two objective functions can be expected. A simple argument for this is that a direct delivery to each customer will yield into a minimum achievable tardiness, but will inflict high cost. Any practical solution consequently must find a balance between the two criteria, and a human decision maker will have to state his/her preferences with respect to the two issues in order to identify a preferred compromise solution. From a decision support and optimization point of view, being able to find all Pareto-optimal solutions, or at least approximations to them, is therefore vital.

3 Solution Approach

3.1 Encoding of Alternatives

Alternatives to the above introduced problem are represented by a set of routes $\mathcal{R} = \{R_1, \ldots, R_m\}$, where each route R_j is driven by a particular vehicle. A route R_j defines an ordered set of customers v_i, which are visited in the sequence in which they appear in the route R_j. Overall, it is thus possible to describe the alternative as a permutation of customers, with additional partitions within the permutation defining the routes.

3.2 Constructive Phase: A Multi-objective Savings Heuristic

The construction of initial solutions is based on the savings heuristic [10]. In this procedure, each customer is first assigned to a distinct vehicle. Then, routes are, if feasible, combined, yielding a 'savings' in terms of the driven distances. For example, the routes $R_1 = \{v_1\}$ and $R_2 = \{v_2\}$ may be combined into $R_{1'} = \{v_1, v_2\}$, reducing the total driven distance by $\overline{v_0v_1} + \overline{v_0v_2} - \overline{v_1v_2}$, with v_0 denoting the depot. Obviously, driving direction decisions play a role when considering time windows. Therefore, $R_{1'} = \{v_1, v_2\}$ but also $R_{2'} = \{v_2, v_1\}$ must be examined in this more general case.

It is possible to further generalize this approach for the above described two-objective case, i. e. the case in which time window violations are generally allowed but minimized by an objective function. What is needed is a mechanism controlling the degree to which time window violations are permitted. This is possible by introducing an parameter $\texttt{TARDY}_{\max}$ that acts as a constraint when executing the constructive savings procedure. While a minimum of $\texttt{TARDY}_{\max} = 0$ is possible, no trivial statements can be made about its' maximum value. Therefore, a first setting of the control parameter would be to assume $\texttt{TARDY}_{\max} = \infty$.

Algorithm 1 describes the logic behind the proposed procedure. A natural termination criterion is found when obtaining a value of $\texttt{TARDY}_{\max} < 0$, which cannot be used an a constraint in a further loop. As a result, the algorithm returns a first approximation of the Pareto-set, denoted with P^{approx}.

Algorithm 1. Multi-objective savings heuristic

1: Set $P^{approx} = \emptyset$
2: Set $\texttt{TARDY}_{\max} = \infty$
3: **repeat**
4: Create new alternative using the Clarke and Wright savings heuristic [10]: In this procedure, allow a maximum tardiness per customer of $\texttt{TARDY}_{\max}$
5: Update P^{approx} with the new alternative
6: Obtain the maximum tardiness $\texttt{TARDY}'_{\max}$ from the constructed alternative, set $\texttt{TARDY}_{\max} := \texttt{TARDY}'_{\max}$
7: Set $\texttt{TARDY}_{\max} := \texttt{TARDY}_{\max} - 1$
8: **until** $\texttt{TARDY}_{\max} < 0$
9: **return** P^{approx}

3.3 Iterative Phase: Population-Based Multi-operator Search

The elements of the initial approximation P^{approx} are further improved by means of local search. In contrast to the single-objective case, which commonly concentrates on the minimization of the traveled distances (or a similar function), some modifications must be made to heuristic search procedures in order to treat the multi-objective nature of the problem.

One the one hand, a set of solutions must be kept throughout the iterative phase. Naturally, evolutionary approaches possess this property, and therefore are a good basis for the considered case. On the other hand, the chosen neighborhood operators must be able to improve the solutions not only with respect to the classical minimization of the traveled distances, but also with respect to other criteria.

In the light of the discussion above, a population-based local search strategy making use of several neighborhood operators at once has been used to solve the problem at hand. We propose to use inversion, exchange, and move operators, each modifying the sequence of customers within the routes. An inversion is defined by the reversal of a subsequence of customers, e. g. transferring $R = \{v_1, v_2, v_3, v_4, v_5, v_6\}$ with the given positions $p_1 = 2$ and $p_2 = 5$ into $R' = \{v_1, v_5, v_4, v_3, v_2, v_6\}$. An exchange swaps the positions of two jobs, i. e. for the example of $R = \{v_1, v_2, v_3, v_4, v_5, v_6\}$ and the given positions $p_1 = 2$ and $p_2 = 5$ a resulting $R' = \{v_1, v_5, v_3, v_4, v_2, v_6\}$. Finally, a move-operator moves a customer from some position p_1 to p_2, keeping the rest of the sequence untouched.

As the neighborhoods are not disjunct, some calculations can be omitted when starting with a particular neighborhood and then continuing with another, obtaining a computational speedup by avoiding unnecessary (duplicate) moves.

Algorithm 2 describes the steps of the procedure. Again, a natural termination criterion can be mentioned, which is the identification of an approximation set P^{approx} containing local optima only.

Overall, the procedure can be characterized as a multi-point multi-neighborhood search algorithm. The concept incorporates ideas from Pareto Local Search [11] and Variable Neighborhood Search (VNS) [12]. Contrary to VNS however, all neighborhood operators are applied when improving the alternatives, thus searching the neighborhoods considerably more thoroughly.

Algorithm 2. Multi-objective local search

Require: $P^{approx}, P^{approx} \neq \emptyset$
Require: Neighborhood definitions $\mathcal{NH} = \{NH_1, \ldots, NH_k\}$
1: **repeat**
2: Select some element x from P^{approx} that is not marked as 'investigated'
3: Compute all feasible neighbors $\mathcal{NH}(x) = \bigcap_{i=1}^{k} NH_i(x)$
4: Evaluate the neighbors
5: Update P^{approx} with $\mathcal{NH}(x)$
6: **if** $x \in P^{approx}$ **then**
7: Mark x as 'investigated' w. r. t. the local search, thus excluding it in further loops from the neighborhood search procedure
8: **end if**
9: **until** All elements of P^{approx} are locally optimal w. r. t. $\mathcal{NH}$
10: **return** P^{approx}

Remark. Step 6 is required, simply because the previous line 5 may lead to a removal of x from P^{approx}.

4 Experimental Investigation

Experiments have been conducted with the initially sketched aim of investigating the computational effort that comes with solving certain data sets. Recalling, that in an interactive setting with alternating phases of search and decision making, the time allocated for the optimization runs may be limited, such insights come useful when giving recommendations for how to tackle a particular problem.

4.1 Benchmark Data

We chose to use established benchmark data from the literature for our experiments. Firstly, this data is openly available, and thus can easily be cross-checked by our peers. Secondly, different problem characteristics are present in the data sets, allowing us to compare the obtained results for instances with differing attributes.

First investigations have been carried out using the famous Solomon VRPTW instances [13]. The data sets contain 100 customers, distributed in an euclidian space, following certain distribution patterns. Such patterns comprise 'clustered', 'random', and a mixture of both, 'random-clustered' instances. Besides, variants with 'wide' and 'narrow' time windows exist, simply meaning that average values of $t_i^l - t_i^e$ are considerable smaller in the latter case.

Figure 1 plots the geographical data of the instances. Interestingly, the clustered data sets show a different geographical distribution of the customers depending on the properties of the time windows. The random and random-clustered instances are however geographically identical, independent from time window data.

4.2 Experiments and Results

Both the constructive and the iterative phase of the local search algorithm have been tested on the described benchmark data sets. A total of 56 instances has been used, as reported in more detail in Table 1.

Table 1. Solomon data sets

Graph	Time windows	No of instances
Clustered	Narrow	9
Clustered	Wide	8
Random	Narrow	12
Random	Wide	11
Random-clustered	Narrow	8
Random-clustered	Wide	8

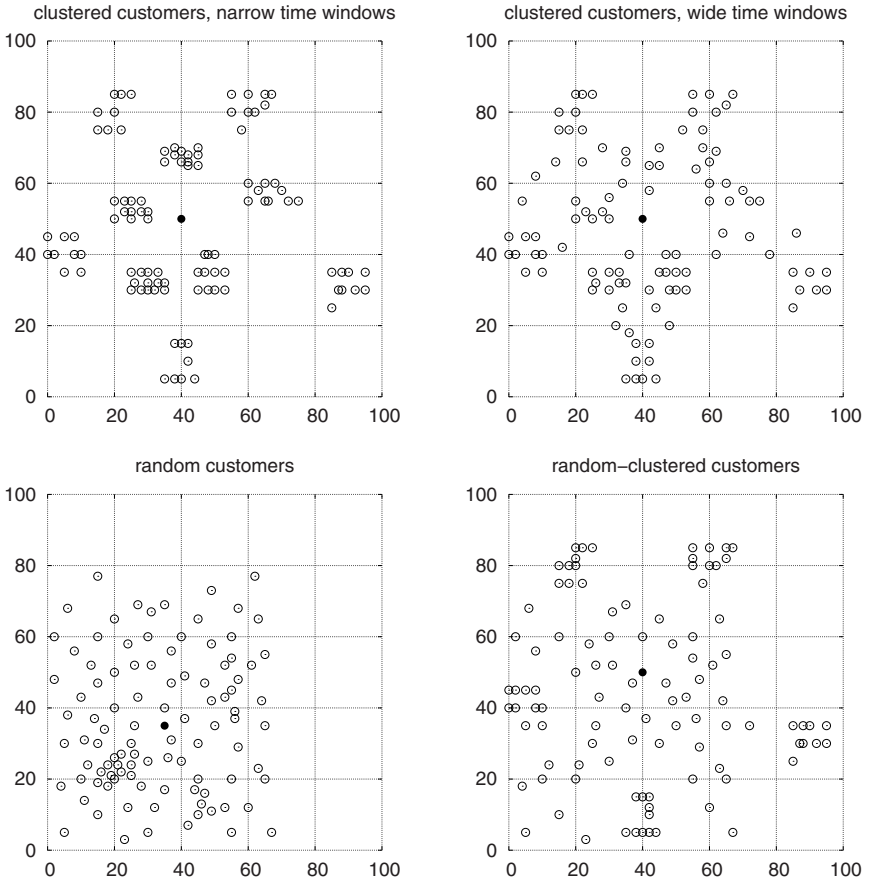


Fig. 1. Solomon instances: Geographical distribution of the depot and the customers

Constructive phase. Table 2 gives the data obtained from the constructive phases of the algorithm. Obviously, the average computational effort is heavily influenced by the underlying structures of the problem, especially when comparing the classes of clustered versus random/random-clustered data sets. Clustered data sets are in average more than ten times faster approximated than instances of the other categories.

Besides the sheer difference in terms of the amount of computations, the average number of elements in the first approximation set P^{approx} differ depending on the structural properties of the data sets. Few solutions are found for clustered instances, and significantly more for the ones of the other classes.

The two findings of this phase of the algorithm go hand in hand. It can be expected that an approximation run yielding a set of larger cardinality is more costly. Then however, this does not entirely explain the apparent difference between the classes of instances. We can suspect that the clustered instances

Table 2. Results of the constructive phase

Graph	Time windows	Average eval.	Average of $\lvert P^{approx}\rvert$	Average hypervolume
Clustered	Narrow	2,983	6	0.5775
Clustered	Wide	4,276	4	0.4724
Random	Narrow	92,702	29	0.7417
Random	Wide	44,151	31	0.6891
Random-clustered	Narrow	54,336	31	0.7378
Random-clustered	Wide	56,469	38	0.6981

favor solutions in which vehicles do not travel in between clusters, canceling out a rather big number of potential solutions. As the other, random instances do not posses this property, more savings are feasible here. In combination with the recursive computation of alternatives as given in Algorithm 1, this leads to an approximation set of larger cardinality.

The rightmost column of Table 2 states the average hypervolume [14] of the obtained approximations. By normalizing the objective function values, the hypervolume assumes a value between 0 and 1, 1 being the best-possible outcome. The reference point has been chosen such that each element of the approximation sets contributes to the hypervolume, also including the later following results of the iterative phase.

It can be seen, that the average hypervolume obtained for instances with narrow time windows is better than the one for instances with wide time windows. Besides, instances with random and random-clustered customers are better approximated when considering the resulting hypervolume indicator.

Iterative phase. The results of the succeeding improvement phase are given in Table 3.

Again, a considerable difference of the computational effort for finding local optima can be, depending on the structural properties of the instances, found. While the algorithm converges rather fast for clustered data sets, significantly more evaluations are required for random, and even more for random-clustered

Table 3. Results of the iterative phase

Graph	Time windows	Average eval.	Average of $\lvert P^{approx}\rvert$	Average hypervolume	Hypervolume improvement
Clustered	Narrow	52,473	6	0.8715	63%
Clustered	Wide	186,456	5	0.9569	178%
Random	Narrow	1,379,383	270	0.8312	13%
Random	Wide	12,425,077	418	0.8115	20%
Random-clustered	Narrow	2,032,069	320	0.8443	15%
Random-clustered	Wide	24,412,509	557	0.8497	22%

instances. Then, the influence of the time windows becomes apparent. Relatively wider time windows lead to less constrained problems, resulting in more potential candidate solutions that have to be examined before reaching a set of local optima.

The difference of the average cardinality of the approximation sets P^{approx} is rather big. This is especially the case between the clustered instances and the random/random-clustered ones with large time windows. It is interesting obtaining this result, especially as it implies that a decision making phase involving a human decision maker will have to work with candidate sets of rather different cardinality. Consequently, different Multiple Criteria Decision Making/Aiding techniques may be used, depending on whether they support decision aiding for many efficient alternatives or not.

Clearly, the iterative phase further contributes to the quality of the obtained results. In case of each single instance, significantly improved solutions have been found. Table 3 illustrates this by reporting the average hypervolume and the relative improvement upon the results of the constructive phase. Relatively higher improvements have been achieved in cases where the constructive phase first led to weaker results. This implies that the succeeding iterative phase successfully balances out the comparably weaker first approximation. Especially for clustered data sets, this is the case.

5 Conclusions

Several insights in the structures of the investigated problem and several conclusions arise from our work.

First, a fast approximation heuristic based on the savings construction procedure has been presented. A first and computationally cheap approximation has been possible by use of the proposed concept. Considerable differences of the computational effort can be found depending on the structures of the data sets. Especially for instances with geographically clustered customers, comparably few computations are required before the algorithm terminates. Nevertheless, a range of alternatives is obtained, allowing the presentation of some first variety of solutions to the decision maker.

Second, population-based local search using multiple neighborhood structures has been tested on the benchmark instances. Again, the computational effort required for finding local optima is significantly influenced by the geographical distribution of customers, and the tightness of the time windows. Maps with randomly distributed customers and wide time windows require significantly more computations than clustered data sets with narrow time windows. With respect to the overall quality of the results, the local search significantly improves the initial approximation.

With respect to our aim of providing decision support in multi-objective vehicle routing, we may conclude that clustered data sets with narrow time windows appear better suited for interactive approaches than instances with randomly distributed customers and wide time windows. In the latter case, a posteriori

approaches should be given preference, assuming that the time allocated for search in between decision making phases is scarce.

References

1. Gendreau, M., Bräysy, O.: Metaheuristic approaches for the vehicle routing problem with time windows: A survey. In: MIC 2003: Proceedings of the Fifth Metaheuristics International Conference, Kyoto, Japan, August 2003, pp. 1–10 (2003)
2. Potvin, J.Y., Kervahut, T., Garcia, B.L., Rousseau, J.M.: The vehicle routing problem with time windows. Part I: Tabu search. INFORMS Journal on Computing 8(2), 158–164 (Spring 1996)
3. Potvin, J.Y., Bengio, S.: The vehicle routing problem with time windows. Part II: Genetic search. INFORMS Journal on Computing 8(2), 165–172 (Spring 1996)
4. Cordeau, J.F., Laporte, G., Mercier, A.: A unified tabu search heuristic for vehicle routing problems with time windows. Journal of the Operational Research Society (52), 928–936 (2001)
5. Jozefowiez, N., Semet, F., Talbi, E.G.: Multi-objective vehicle routing problems. European Journal of Operational Research 189(2), 293–309 (2008)
6. Park, Y.B., Koelling, C.P.: An interactive computerized algorithm for multicriteria vehicle routing problems. Computers & Industrial Engineering 16(4), 477–490 (1989)
7. Pacheco, J., Marti, R.: Tabu search for a multi-objective routing problem. Journal of the Operational Research Society (57), 29–37 (2006)
8. Phelps, S.P., Köksalan, M.: An interactive evolutionary metaheuristic for multiobjective combinatorial optimization. Management Science 49, 1726–1738 (2003)
9. Taillard, É., Badeau, P., Gendreau, M., Guertin, F., Potvin, J.Y.: A tabu search heuristic for the vehicle routing problem with soft time windows. Transportation Science 31(2), 170–186 (1997)
10. Clarke, G., Wright, J.W.: Scheduling of vehicles from a central depot to a number of delivery points. Operations Research 12, 568–581 (1964)
11. Paquete, L., Chiarandini, M., Stützle, T.: Pareto local optimum sets in the biobjective traveling salesman problem: An experimental study. In: Gandibleux, X., Sevaux, M., Sörensen, K., T'kindt, V. (eds.) Metaheuristics for Multiobjective Optimisation. Lecture Notes in Economics and Mathematical Systems, vol. 535. Springer, Heidelberg (2004)
12. Hansen, P., Mladenović, N.: Variable neighborhood search. In: Glover, F., Kochenberger, G.A. (eds.) Handbook of Metaheuristics. International Series in Operations Research & Management Science, vol. 57, pp. 145–184. Kluwer Academic Publishers, Dordrecht (2003)
13. Solomon, M.M.: Algorithms for the vehicle routing and scheduling problems with time windows constraints. Operations Research 35(2), 254–265 (1987)
14. Auger, A., Bader, J., Brockhoff, D., Zitzler, E.: Theory of the hypervolume indicator: Optimal μ-distributions and the choice of the reference point. In: Proceedings of the tenth ACM SIGEVO workshop on Foundations of Genetic Algorithms, Orlando, Florida, January 2009, pp. 87–102 (2009)

Particle Swarm Optimization and an Agent-Based Algorithm for a Problem of Staff Scheduling

Maik Günther and Volker Nissen

Ilmenau University of Technology, Information Systems in Services, D-98684 Ilmenau
maik.guenther@gmx.de, volker.nissen@tu-ilmenau.de

Abstract. Eight problems of a practical staff scheduling application from logistics are used to compare the effectiveness and efficiency of two fundamentally different solution approaches. One can be called *centralized* and is based on *search* in the solution space with an adapted metaheuristic, namely particle swarm optimization (PSO). The second approach is *decentralized*. Artificial agents negotiate to *construct* a staff schedule. Both approaches significantly outperform todays manual planning. PSO delivers the best overall results in terms of solution quality and is the method of choice, when CPU-time is not limited. The agent approach is vastly quicker in finding solutions of almost the same quality as PSO. The results suggest that agents could be an interesting method for real-time scheduling or re-scheduling tasks.

Keywords: staff scheduling, sub-daily planning, particle swarm optimization, multi-agent system, combinatorial optimization.

1 Introduction

Employees spend up to 36% of their working time unproductively, depending on the branch [14]. Major reasons include a lack of planning and controlling. Staff scheduling assigns employees to workstations subject to constraints. In practice, planning often takes place based on prior experience or with the aid of spreadsheets [1]. It is obvious that demand-oriented staff scheduling cannot be realised with these planning tools. Even with popular staff planning software employees are regularly scheduled for one workstation per day. However, in many branches, such as logistics and trade, the one-employee-one-station concept does not correspond to the actual requirements and sacrifices potential resources. Intra-day variations in demand require more flexible changes of employees among workstations. Therefore, *sub-daily* planning should be an integral component of demand-oriented staff scheduling.

Staff scheduling is a hard optimization problem. Garey and Johnson [6] demonstrate, that even simple versions of staff scheduling problems are NP-complete. Kragelund and Kabel [10] show the NP-hardness of the general employee timetabling problem. According to Puppe et al. [15], centralized scheduling approaches are difficult to employ successfully. In this paper we investigate whether

C. Di Chio et al. (Eds.): EvoApplications 2010, Part II, LNCS 6025, pp. 451–461, 2010.

this is actually true for sub-daily staff scheduling problems, which are typical for industries such as trade and logistics as well as call centers. A centralized metaheuristic approach that utilizes Particle Swarm Optimization (PSO) is compared to a decentralized multi-agent approach. PSO was chosen because in previous tests it outperformed other heuristics, namely evolution strategies [12] and Local Search [16] on the given test problems. Multiple artificial agents, on the other hand, are a promising way to achieve decentralized problem solving.

In the following section, we describe the practical planning scenario from logistics before we discuss work related to our own research in Section 3. Section 4 presents the two solution methods based on PSO and artificial agents. The experimental setup and empirical results are presented and discussed in Section 5. The paper concludes with a short summary.

2 A Real-World Problem from Logistics

Due to the given page limit the problem can not decribed in detail here. For a compehensive description and a mathematical representation see [12]. And for real-world data sets and benchmarks see [16]. The present problem originates from a German logistics service provider where the task is to find a staff schedule that respects certain hard constraints and minimizes the violation of soft contraints. An example of a hard constraint is that any one employee must only be assigned to one workstation at a time. An example of a soft constraint would be the avoidance of understaffing. The violation of soft constraints is penalized with error points that reflect the companys requirements. The objective is to minimize the total count of error points. Ernst et al. offer a summary of papers related to the issue of staff scheduling [5]. They identify certain categories of problems, such as *flexible demand*. Our application can be classified as this category and additionally as *task assignment*.

The tasks of employees concern logistic services e.g. loading and unloading or short distance transportation. The employees are quite flexible in terms of their working hours, which results in 13 different working-time models. There are strict regulations with regard to qualifications, because the assignment of unqualified employees might lead to significant damage. Many employees can work at several different workstations. Currently, monthly staff scheduling is carried out manually within MS EXCEL^{TM}. Employees are assigned a working-time model and a fixed workstation each day. Several considerations are included, such as absence, timesheet balances, qualifications and resting times etc. The personnel demand for each workstation is subject to large variations during the day, causing large phases of over- and understaffing. This lowers the quality of service and the motivation of employees and leads to unnecessary personnel costs as well as downtime.

We investigate a total of 8 problem instances associated with this practical case: planning for the full week as well as planning each of the seven days individually. The full week problem covers seven days (20 hours each), divided into 15-minute intervals. It includes 65 employees and, thus, an uncompressed total

of 36,400 dimensions for the optimization problem to be solved. The general availability of the employees (based on working-time models) is known for each interval from the previous full-day planning and must not be changed. Nine different workstations need to be filled. Planning the individual days is less complex with 80 time slots and between 38 and 46 employees to be considered. The demand schemes vary significantly for different days so that a representative spectrum of the real-world situation is given in the problem instances.

A solution is represented as a two-dimensional matrix of employees and time periods, where the cells are filled with workstation assignments (Table 1). To mark times, in which an employee is not present due to his work-time model, workstation 0 is used. For example, employee two is absent in the first period and then is assigned to workstation 2. Assignment changes can only be made to cells of available employees.

Table 1. Assignment of workstations in a matrix

employee	**period**						
	1	2	3	4	5	6	...
1	1	1	1	1	1	1	
2	0	2	2	2	2	2	
3	0	1	1	2	2	2	
...							

3 Related Work

Poli analysed the IEEE Xplore database for the thematic grouping of PSO applications in 2007 [13]. Of approximately 1,100 publications only one work is focused specifically on timetabling [3], which is related to our own application problem. Chu et al. adjust PSO to the combinatorial domain [3]. No longer is the position of a particle determined by its speed, but rather by using permutation operators. Brodersen uses this concept for a related problem of university timetabling [2].

In a different paper, we compare heuristics based on PSO and evolution strategies (ES) for the current problem set. PSO outperforms ES on a statistically significant level [12]. Moreover, PSO also shows better performance than local search [16]. It is this PSO-implementation (with a repair heuristic added) that we compare to a multi-agent approach here. In [7] various neighbourhood topologies are tested for the same problems with a gBest topology, where each particle is a neighbour of every other particle, performing best.

Puppe et al. [15] present two concepts for artificial agents on scheduling in hospitals. In the resource-oriented view each resource or the associated organizational unit is represented as an agent. This concept is more applicable, when the problem is static. In the patient-oriented view, an agent is created for every patient examination, which is more adapted to dynamical problems.

Krempels [11] also creates a staff schedule by using agents. The agent approach is divided in several phases. Initially a planner agent creates a plan ignoring staff preferences. Thereafter, the planner tries to improve the plan by incorporating preferences. A knowledge tank stores all relevant aspects of the resources. In case of a conflict, an agent is created for each staff member, followed by a negotiation phase.

De Causemaecker et al. [4] make comments on negotiation schemes for course timetabling. Only necessary information should be exchanged among agents. Moreover, a negotiation process should not take exceedingly long.

4 PSO and Artificial Agent Approach

4.1 PSO for This Application

The basic principles of PSO were developed by Kennedy and Eberhart among others [8], [9]. Swarm members are assumed to be massless, collision-free particles, that search for optima with the aid of a fitness function within a solution space. In this process, each single particle together with its position embodies a solution to the problem. While looking for the optimum, a particle does not simply orient itself using its own experience, but also using the experience of its neighbours. The particles exchange information, which can then positively influence the development of the population in the social system as a whole. The following pseudocode presents an overview of the implemented PSO. Here, pBest represents the best position found so far by the particle while gBest corresponds to the best position of all particles globally.

```
01: initialise the swarm
02: evaluate the particles of the swarm
03: determine pBest for each particle and gBest
04: loop
05:    for i = 1 to number of particles
06:       calculate new position // use the 4 alternative actions
07:       repair the particle
08:       evaluate the particle
09:       if f(new position)<f(pBest) then pBest=new position // new pBest
10:       if f(pBest)<f(gBest) then gBest=pBest // new gBest
11:    next i
12: until termination
```

The initialisation of the particle position creates valid assignments w.r.t. the hard constraints by using information from the company's current full-day staff schedule. Valuable prior knowledge is not wasted. Based on this plan, improved solutions can now be determined that include plausible workstation changes.

In each iteration the new particle position is determined by traversing all dimensions and executing one of the following actions with predefined probability. The probability distribution was heuristically determined in prior tests. The behaviour of the PSO-heuristic is relatively insensitive to changes of p_1, p_3, and p_4. The optimal value for p_2 depends on the problem size.

- No change (p_1=9.7%): The workstation already assigned remains.
- Random workstation (p_2=0.3%): A workstation is randomly determined and assigned. Only those assignments are made, for which the employee is qualified. The probability function is uniformly distributed.
- pBest workstation (p_3=30%): The corresponding workstation is assigned to the particle dimension from pBest. Through this, the individual PSO component is taken into account.
- gBest workstation (p_4=60%): The corresponding workstation is assigned to the particle dimension from gBest. gBest was found to work best as a neighbourhood topology for this type of application in [7]. By considering the best position of all particles, the swarm's experience is included in the position calculation.

PSO employs a repair heuristic to reduce the total error points of a solution before it undergoes evaluation. This repair heuristic corrects constraint violations in the following order, based on error point size:

- qualification: employees, not qualified for the currently assigned workstation, are given an appropriate assignment, whilst ignoring under- or overstaffing.
- no demand: employees, currently assigned to a workstation with zero demand, are given a different assignment (if possible), whilst simultaneously considering their qualification.
- understaffing: if workstations are understaffed, employees are reassigned from other workstations with overstaffing (if possible) also considering their qualification. Simultaneously the problem of overstaffing is reduced.

The characteristics of PSO have not been changed with these modifications. There are merely changes in the way to determine a new particle position, so that the calculation of the velocity is not needed. The current form of position determination makes it unnecessary to deal with dimension overruns. All other peculiarities of PSO, regarding social or individual behaviour, remain. In our implementation, PSO terminates after 400,000 inspected solutions. Alternatively, convergence-based termination criteria could be employed.

4.2 Artificial Agents for This Application

Following the suggestion of Puppe et al. [15], resource-oriented agents are used for this static staff scheduling application. In our problems, constraints and preferences come from two directions. On one side is the employer who aims at reduced overall costs, a high service level, the consideration of qualifications in the schedule etc. On the other side there are the employees, that try to enforce their rights, such as legal regulations and the minimization of workstation rotations during the day. Consequently, following Krempels [11], our approach is structured in two phases associated with employer and employees.

Fig. 1 shows a schematic representation of our multi-agent approach, which also respects the recommendations in [4]. The individual steps, that finally construct a staff schedule, can be described as follows:

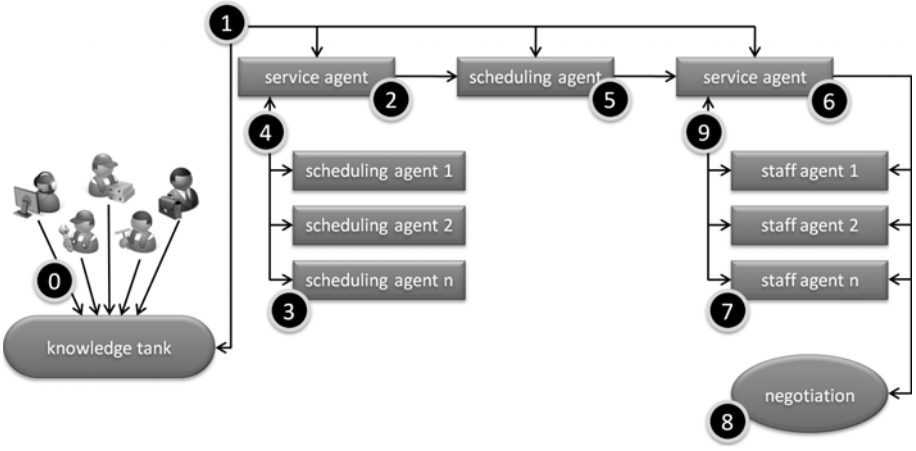


Fig. 1. Representation of the agent approach for the logistics problem

- First, the properties of existing resources, current demands and conditions of the problem space are stored in the knowledge tank (0). This information includes absence of employees, required qualifications, error-point values for violations of restrictions, personnel requirements of each interval etc.
- The information in the knowledge tank is supplied (1) to three agents (2), (5) and (6).
- Before starting to plan, service agent (2) initialises the schedule by assigning all employees to a dummy workstation. This indicates, that these employees are not currently assigned to an actual workstation.
- Following that, service agent (2) ranks the nine workstations, with the highest priority going to workstations for which the least number of employees are qualified. Should the number of qualified staff for two or more workstations be identical, then the priorities are ordered at random.
- Scheduling agents (3) are sequentially initialised by the service agent (2), according to priority. Each scheduling agent (3) represents one of the nine workstations. Only one scheduling agent (3) exists at any time. The scheduling agent, for which the fewest employees are qualified, begins. He schedules qualified employees, who are present and have not yet been assigned. Over- and understaffing should be minimised as much as possible. The planning result of the first scheduling agent is passed (4) to service agent (2), which in turn gives feedback regarding the schedule to the knowledge tank (1). Then, service agent (2) initiates the next scheduling agent (3), which also attempts to cover its personnel demand as best as possible. During this process, previously assigned employees may not be deployed to subsequent workstations. Service agent (2) sequentially initiates scheduling agents (3) until all nine workstations have been processed.
- After an assignment plan was created, there could still be employees in some timeslots, who have not yet received an assignment. Switching employees to

other workstations could result in better coverage of demand. The service agent (2) calls a scheduling agent (5), also connected (1) to the knowledge tank. Scheduling agent (5) finalises the schedule by deploying all workers, who are still unassigned, necessarily accepting overstaffing. Possible switches are again checked as to whether they would lead to better demand coverage and those that would are carried out.

- Assignment planning was done up to now from the point of view of the company. This occurred while neglecting employee needs – the reduction of the number of workstation rotations. For this reason, scheduling agent (5) initiates another service agent (6) in order to consider employee preferences. This service agent (6) is also connected to the knowledge tank (1).
- Service agent (6) examines each timeslot in the schedule and checks whether a workstation rotation occurs. If this is the case, all workers are identified for whom a negotiation could occur for this timeslot. They must be present in the timeslot and qualified for the switch. Service agent (6) simultaneously generates a staff agent (7) for each relevant employee. In contrast to the scheduling agents (3), more than one staff agent exists at the same time.
- Two staff agents (7) negotiate a workstation assignment switch (8) in the following way: The staff agent where the service agent (6) identified a workstation rotation sequentially asks the other staff agents for a swap. Each staff agent knows its current workstation assignment at times t, t-1 und t+1. The two communicating staff agents exchange only information about their assignments at time t. Without re-calculating the whole fitness function they can now decide, if a swap would reduce the overall error count of the schedule. If this is the case, they agree to swap and communicate (9) this to the service agent (6). Then, the swap is executed and all staff agents are deleted. If a swap would not reduce the error count, the process continues by asking the next staff agent in the queue. As a result, a swap may or may not occur for each staff agent where a workstation rotation was identified, depending on the availability of a swap option that improves the overall error count of the schedule.
- In addition to the negotiation (8) between staff agents (7), a negotiation is also carried out between the service agent (6) and the staff agent, for which the workstation rotation was identified. The goal of this negotiation is not to execute a switch with another staff agent, but rather to carry out a switch at time t for the workstation at which the employee is working at times t-1 or t+1. This also helps reduce the number of workstation rotations. Service agent (6) only agrees to the switch, if the overall quality of staff assignments does not deteriorate. The result of the negotiation is either the assignment to a different workstation at time t (and thus the reduction of workstation rotations) or keeping the assignment as is. This decision is reported to service agent (6) and carried out.
- Service agent (6) repeats the last three steps up to the point where no further improvements occur.

Table 2. Comparison (error points) of the approaches, based on 30 independent runs each. Best results are bold and underlined. For PSO, the swarm size is given in brackets.

heuristic	error			number of job-changes	wrong skills in minutes	under-staffing in minutes	overstaffing in minutes	
	mean	min	standard deviation				demand >0	demand =0
manual plan	411330	411330	-	0.0	1545	20130.0	14610.0	33795.0
PSO (10)	**51752**	**51736**	9.2	1502.2	0	7365.0	28395.0	7245.0
PSO (20)	51781	51763	9.3	1531.4	0	7365.0	28395.0	7245.0
PSO (100)	51826	51811	9.0	1575.8	0	7365.0	28395.0	7245.0
Agents	51829	51801	15,4	1579,0	0	7365.0	28395.0	7245.0

Table 3. t-test results for comparison of multi-agent approach and best PSO

H_1	T	df	significance H_0 (l-tailed)	mean difference	95% confidence intervall of differences	
					lower	upper
PSO(10) < Agents	-23.57	47.17	< 0.001	-77.27	-83.86	-70.67

5 Results and Discussion

The results of the various scheduling approaches for the problem of the whole week are shown in Table 2. All test runs were conducted on a PC with an Intel 4 x 2.67 GHz processor and 4 GB of RAM. Thirty independent runs were conducted each time for each of the experiments to allow for statistical testing. An individual run with the multi-agent approach takes approx. 1 sec of CPU-time. The runtime requirements for the PSO approach are much higher and in the order of 50 minutes per run (PSO terminates after 400,000 inspected solutions). This effort, however, is acceptable as there is sufficient time available for creating the schedule.

Starting with the complex entire week problem, the full-day manual staff schedule with MS EXCEL^{TM} results in 411,330 error points. All heuristics for sub-daily staff scheduling significantly outperform the manual full-day schedule in terms of total error points. This demonstrates the value of sub-daily scheduling as compared to today's standard staff scheduling approach. The problems of understaffing and overstaffing for periods without demand are greatly reduced. On the other hand, all approaches lead to more overstaffing in periods with $demand > 0$, as compared to the initial plan. This, however, is sensible because employees can still support each other instead of being idle when $demand = 0$.

PSO provides the best results with a rather small swarm size of 10 particles, but also larger swarm sizes produce good results. Many steps are required to arrive at a good schedule. Thus, it seems preferable to track changes for more iterations as compared to a higher diversity through larger population or swarm size with the current termination criterion.

The results from the multi-agent system are quite close to the schedules created by PSO. With 30 independent runs for each heuristic it is possible to test the performance difference of the best parameterisation of PSO (swarm size 10) and the multi-agent approach for statistical significance with a t-test (see Table 3). A Levene-test revealed the heterogeneity of variances (test level 5%) between both groups ($F = 6.585, p = 0.013$). The corresponding t-test with a 95% confidence interval confirms the better performance of PSO (10) with a very high statistical significance ($p < 0.001$ for H_0).

An advantage of the multi-agent system, alongside the low CPU-requirements, is the relative simplicity of its scheduling strategy. While it is hard for a staff planner to grasp what is really happening during optimization with PSO, the acceptance for the agent-derived solution is likely to be far higher, since the individual steps of the planning and negotiation procedure are relatively straightforward and familiar for staff managers. The importance of this comprehensibility for the acceptance of the resulting schedule should not be underestimated.

The agent approach does not violate qualification constraints and over- as well as understaffing are reduced to the possible minimum as found by PSO. It is only the number of sub-daily workstation rotations that is greater in the solutions produced by the multi-agent system. To achieve an improved solution quality, an extended re-scheduling and swapping of assignments would have been required. It must consider more than two staff members in parallel as well as large parts of the planning horizon. This is beyond what is possible through one-to-one negotiation of a staff agent with the service agent or other staff agents. It can only be achieved with the aid of a central planning instance, that partly ignores the individual preferences of agents for a better overall result of the entire schedule. Such a central planning instance, however, is not in line with the distributed negotiation and decision scheme that is generally associated with multi-agent systems.

The multi-agent and PSO approaches were also tested on the smaller problem sets, representing the individual days of the week. Table 4 shows the respective mean errors (based again on 30 runs) for each day. The relative performance is similar to the more complex week problem discussed before, supporting our previous conclusions.

Two forms of hybridization of the agent approach with PSO were tested, but both could not improve upon the results for the "pure" approaches presented above. First, our additional experiments indicated, that initializing the PSO start solutions with the help of the multi-agent system leads to premature con-

Table 4. Mean results for individual days of the week problem (30 runs each). Best results are bold and underlined.

	Mo	Tu	We	Th	Fr	Sa	So
PSO (10)	**7712**	**5900**	**8161**	**8248**	**5500**	**8838**	7330
PSO (20)	7726	5910	8171	8257	5508	8846	**7325**
PSO (100)	7725	5909	8170	8255	5508	8844	**7325**
Agents	7727	5917	8183	8272	5528	8861	7337

vergence. The swarm is not sufficiently diversified from the beginning and the particles are already in a very good local optimum, such that further progress is extremely difficult. Second, trying to use PSO as an individual optimization strategy for staff agents was also not successful, since it is relatively straightforward how individual agents can improve their situation through negotiation. (If an employee wants to remove his workstation rotation at time t, then his staff agent negotiates with other staff agents whether he can work at the workstation he occupies at t-1 or t+1.) The same applies to the strategy of the scheduling agents. Thus, so far the experimental results indicate, that hybridization of PSO and the agent-based scheduling approach is not a fruitful strategy in our domain.

6 Conclusion and Future Work

Using complex, high-dimensional and highly constrained planning scenarios, it was demonstrated, that PSO (as a rather centralized approach) and artificial agents (a constructive and decentralized approach) produce far better results than today's spreadsheet-based full day scheduling. Thus, sub-daily scheduling significantly increases the value contributions of individual staff members.

Because PSO in its traditional form is not suitable for the planning problems at hand, the method was adapted to the combinatorial domain without sacrificing the basic PSO mechanism. Based purely on solution quality, the PSO approach has a slight advantage over the agent approach and should, thus, be favored when runtime is not a seriously limiting factor for optimization. In our practical applications this is the case. This success of PSO contradicts Puppe et al. [15] who suggest that centralized scheduling methods are likely to fail due to the many constraints and complexity of the task.

The multi-agent approach is vastly quicker in finding solutions of almost the same quality as PSO. Multi-agent systems have rarely been shown to be competitive with modern metaheuristics. Moreover, the results suggest, that artificial agents could be useful for real-time scheduling or re-scheduling tasks where runtime for the optimization is usually very limited.

Investigations of seven easier versions of the same application problem were undertaken, in which similar results were achieved. In order to base the conclusions of this work on a wider foundation, the investigations done here are currently extended to a practical problem from the trade domain, which is even more extensive with respect to dimensions and constraints.

References

1. ATOSS Software AG, FH Heidelberg (eds.): Standort Deutschland 2006. Zukunftssicherung durch intelligentes Personalmanagement. München (2006)
2. Brodersen, O.: Eignung schwarmintelligenter Verfahren für die betriebliche Entscheidungsunterstützung. Cuvillier, Göttingen (2008)
3. Chu, S.C., Chen, Y.T., Ho, J.H.: Timetable Scheduling Using Particle Swarm Optimization. In: Proc. of ICICIC 2006, vol. 3, pp. 324–327 (2006)

4. De Causemaecker, P., Ouelhadj, D., Vanden Berghe, G.: Agents in Timetabling Problems. In: Proc. of MISTA 2003, pp. 67–71 (2003)
5. Ernst, A.T., et al.: An Annotated Bibliography of Personnel Scheduling and Rostering. In: Annals of OR, vol. 127, pp. 21–144 (2002)
6. Garey, M.R., Johnson, D.S.: Computers and Intractability. In: A Guide to the Theory of NP-Completeness. Freeman, New York (1979)
7. Günther, M., Nissen, V.: A Comparison of Neighbourhood Topologies for Staff Scheduling With Particle Swarm Optimisation. In: Mertsching, B., Hund, M., Aziz, Z. (eds.) KI 2009. LNCS (LNAI), vol. 5803, pp. 185–192. Springer, Heidelberg (2009)
8. Kennedy, J., Eberhart, R.C.: Particle Swarm Optimization. In: Proc. of IEEE Int. Conf. on Neural Networks,, pp. 1942–1948 (1995)
9. Kennedy, J., Eberhart, R.C., Shi, Y.: Swarm Intelligence. Kaufmann, San Francisco (2001)
10. Kragelund, L., Kabel, T.: Employee Timetabling. An Empirical Study. Master's Thesis, Dep. of Computer Science, Univ. of Aarhus (1998)
11. Krempels, K.H.: Lösen von Scheduling-Konflikten durch Verhandlungen zwischen Agenten. In: Sauer, J. (ed.) Proc. of PuK 2002, pp. 86–89 (2002)
12. Nissen, V., Günther, M.: Staff Scheduling with Particle Swarm Optimization and Evolution Strategies. In: Cotta, C., Cowling, P. (eds.) EvoCOP 2009. LNCS, vol. 5482, pp. 228–239. Springer, Heidelberg (2009)
13. Poli, R.: An Analysis of Publications on Particle Swarm Optimization. Report CSM-469, Dep. of Computer Science, Univ. of Essex (2007)
14. Proudfoot Consulting: Produktivitätsbericht 2007. Company Report (2007)
15. Puppe, F., Klügl, F., Herrler, R., Kirn, S., Heine, C.: Konzeption einer flexiblen Agentenkomponente für Schedulingaufgaben im Krankenhausumfeld. In: Proc. of 2. Kolloquium Intelligente Softwareagenten und betriebswirtschaftliche Anwendungsszenarien (2000)
16. Sub-Daily Staff Scheduling Data Sets and Benchmarks, http://www.tu-ilmenau.de/fakww/2608+M54099f70862.0.html

A Math-Heuristic for the Multi-Level Capacitated Lot Sizing Problem with Carryover

Marco Caserta[1], Adriana Ramirez[2], and Stefan Voß[1]

[1] Institute of Information Systems, University of Hamburg,
Von-Melle-Park 5, 20146 Hamburg, Germany
[2] Telecommunications and Systems Engineering Department,
Universitat Autonoma de Barcelona, 08193 Bellaterra, Barcelona, Spain

Abstract. We present a math-heuristic algorithm for the lot sizing problem with carryover. The proposed algorithm uses mathematical programming techniques in a metaheuristic fashion to iteratively solve smaller portions of the original problem. More specifically, we draw ideas from the corridor method to design and impose exogenous constraints on the original problem and, subsequently, we solve to optimality the constrained problem using a MIP solver. The algorithm iteratively builds new corridors around the best solution found within each corridor and, therefore, explores adjacent portions of the search space. In the attempt of fostering diversification while exploring the original search space, we generate a pool of incumbent solutions for the corridor method and, therefore, we reapply the corridor method using different starting points. The algorithm has been tested on instances of a standard benchmark library and the reported results show the robustness and effectiveness of the proposed scheme.

1 Introduction

The Multi-Level Capacitated Lot Sizing Problem (MLCLSP) finds application in production systems where setup times are significant (see, *e.g.*, [2,14]). An example of such a production system is the job-shop system, *i.e.*, a production system where operations require specialized manufacturing processes such as costumer adapted orders or small batch jobs. The MLCLSP is an extension of the Capacitated Lot Sizing Problem (CLSP) [9,13]. In the problem, multiple items (products) must be produced following a known Bill of Material (BOM). The objective is to find an optimal production plan that minimizes production, setup, and inventory costs, and delivers optimal lot sizes and production periods for each product. In the problem, the external demands (volume) are given for predefined periods. When an item is produced, machine (resource) capacity is consumed, which is limited. When there is a change of production from one item to another, it is also necessary to account for setup costs as well as setup times. Incurring in setup times means reducing the available machine capacity. Lastly, whenever there is an excess of production over the current demand in a given period for a specific item, inventory is built up and, consequently, inventory holding costs must be paid.

C. Di Chio et al. (Eds.): EvoApplications 2010, Part II, LNCS 6025, pp. 462–471, 2010.

This work deals with the MLCLSP with setup carry-over (MLCLSP-CO), also called MLCLSP with linked lot sizes [2,14,10,3]. The MLCLSP-CO introduces some scheduling information into the classical lot sizing problem. Whenever an item is produced over two consecutive periods on the same machine, its setup in the second period can be discounted, under the assumption that its setup state is "carried over" the two periods. This implies that a partial scheduling is introduced, *i.e.*, which item is produced as the last one in the first period and the first one in the second period is established [9]. Under the classical "small bucket vs. big bucket" classification, MLCLSP-CO is a big-bucket model [10,12], as it is an extension of the CLSP. Thus, the MLCLSP-CO, being an extension of the standard CLSP, is NP-hard [8].

In [14] a lagrangean heuristic is presented. First, inventory balance constraints, capacity constraints, and carryover constraints are relaxed in a lagrangean fashion. A Wagner-Within algorithm is used to compute a lower bound. Next, a pool of repair schemes is used to find a feasible solution.

Similarly, in [3] a corridor method algorithm paired with dynamic programming and a lagrangian based heuristic is used to generate a (possible infeasible) solution. Subsequently, three different repair mechanisms are implemented to reach a feasible solution.

In this paper we present a math-heuristic algorithm for the MLCLSP-CO. The proposed algorithm hybridizes mathematical programming techniques with a metaheuristic to iteratively solve smaller portions of the original problem. More specifically, we utilize the corridor method together with a mathematical programming formulation which is solved with a MIP solver.

The paper is structured as follows. First, in the next section, we provide a mathematical programming formulaton of the problem. Then, the basic idea of the algorithm is outlined in Section 3. A simple scheme for the generation of feasible incumbent solutions is presented in Section 4. Section 5 presents computational results on a set of benchmark instances and, finally, Section 6 offers a few final remarks.

2 A Formal Model for the MLCLSP-CO

In this section, we present a mixed integer formulation for the problem. The MLCLSP-CO model presented in this work is based on some models from the literature, particularly [2], [14], [10], [3], and [12].The following notation is used to formulate the MLCLSP-CO.

Indices and index sets:
j is the items index, with $j = 1, 2, ..., J$;
m is the machine index, with $m = 1, 2, ..., M$;
t is the period index, with $t = 1, 2, ..., T$;
$\Gamma(m)$ is the set of items produced on machine m;
$\Lambda(j)$ is the set of predecessors of item j;
$\Sigma(j)$ is the set of successors of item j.

Parameters:
f_{jt} is the setup cost of item j in period t;
c_{jt} is the unitary production cost of item j in period t;
h_{jt} is the unitary holding cost of item j in period t;
tp_{jt} is the unitary resource usage of item j in period t;
ts_{jt} is the setup time of item j in period t;
b_{mt} is the capacity of resource m in period t;
d_{jt} is the external demand of item j in period t;
γ_{ij} is the number of units of item i required to produce one unit of item j;
Z big number;
$\hat{s}_j$ is the physical inventory of item j at the beginning of the planning horizon.

Decision variables:
x_{jt} is the binary setup variable for item j in period t;
ω_{jt} is the binary setup carry-over variable for item j at the beginning of period t;
y_{jt} is the production quantity (lot size) of item j in period t;
s_{jt} is the inventory of item j at the end of period t.

A MIP formulation for the problem is given below (we assume a lead time of 1 period). It captures the following features: Constraints (1)–(3) are balance constraints; Constraint (4) is a capacity constraint, taking into account both variable production times and setup times; Constraint (5) establishes the relation between production and setup variables; Constraint (6) ensures that only one carryover per machine is fixed; Constraints (7) and (8) account for the relation between setup and carryover variables; Constraint (9) ensures that consecutive carryover for the same item are allowed only when no other items are setup on that machine; and Constraint (10) makes sure that no carryover is established in the first period.

3 General Idea and Algorithm

The general idea of the proposed algorithm relies on the combination of mathematical programming techniques with metaheuristic features. More precisely, we use mathematical programming techniques in a metaheuristic fashion, following the paradigm of "math-heuristic" algorithms.

Small-size instances of the MLCLSP-CO can be solved to optimality using modern MIP solvers. However, when the size of the instances grows, such solvers fail to deliver optimal solutions in a timely fashion. Nevertheless, in many cases, the same MIP solvers are able to quickly provide good quality solutions in a short computational time when used on *constrained* versions of the original problem. The key idea of the proposed approach is to iteratively define smaller versions of the MLCLSP-CO problem by imposing exogenous constraints on the original problem. Thus, the MIP solver is used to solve, perhaps to optimality, such smaller problems. The mechanism guiding the generation of subproblems is, in turn, guided by a metaheuristic.

$$\min \quad \sum_{t=1}^{T}\sum_{j=1}^{J}\left[c_{jt}y_{jt} + f_{jt}(x_{jt} - \omega_{jt}) + s_{jt}h_{jt}\right]$$

s.t.

$$s_{jt-1} + y_{jt} = s_{jt} + d_{jt} + \sum_{k\in\Sigma(j)} \gamma_{jk}x_{kt+1}, \qquad \begin{array}{l} j = 1,\ldots,J, \\ t = 1,\ldots,T-1, \end{array} \tag{1}$$

$$s_{jT-1} + y_{jT} = s_{jT} + d_{jT}, \qquad j = 1,\ldots,J, \tag{2}$$

$$s_{j0} = \hat{s}_j - \sum_{k\in\Sigma(j)} y_{k1}, \qquad j = 1,\ldots,J, \tag{3}$$

$$\sum_{j\in\Gamma(m)} tp_{jt}y_{jt} + ts_{jt}(x_{jt} - \omega_{jt}) \leq b_{mt}, \qquad \begin{array}{l} m = 1,\ldots,M, \\ t\ \ = 1,\ldots,T, \end{array} \tag{4}$$

$$y_{jt} \leq Zx_{jt}, \qquad \begin{array}{l} j = 1,\ldots,J, \\ t = 1,\ldots,T, \end{array} \tag{5}$$

$$\sum_{j\in\Gamma(m)} \omega_{jt} \leq 1, \qquad \begin{array}{l} m = 1,\ldots,M, \\ t\ \ = 2,\ldots,T, \end{array} \tag{6}$$

$$\omega_{jt} \leq x_{jt}, \qquad \begin{array}{l} j = 1,\ldots,J, \\ t = 2,\ldots,T, \end{array} \tag{7}$$

$$\omega_{jt} \leq x_{jt-1}, \qquad \begin{array}{l} j = 1,\ldots,J, \\ t = 2,\ldots,T, \end{array} \tag{8}$$

$$Z\left(2 - \omega_{jt} - \omega_{jt+1}\right) + 1 \geq \sum_{i\in\Gamma(m)} x_{it}, \qquad \begin{array}{l} t\ \ = 2,\ldots,T, \\ m = 1,\ldots,M, \\ j\ \in\Gamma(m), \end{array} \tag{9}$$

$$\omega_{j1} = 0, \qquad j = 1,\ldots,J, \tag{10}$$

$$y_{jt}, s_{jt} \geq 0, \qquad \begin{array}{l} j = 1,\ldots,J, \\ t = 1,\ldots,T, \end{array} \tag{11}$$

$$x_{jt}, \omega_{jt} \in \{0,1\}, \qquad \begin{array}{l} j = 1,\ldots,J, \\ t = 1,\ldots,T. \end{array} \tag{12}$$

The *corridor method* (CM) is a hybrid metaheuristic proposed by [11]. While some successful applications of the method have already been reported (*e.g.*, [5,6]), its full exploitation to solve complex combinatorial optimization problems still needs to be advanced. The central idea of the CM is the use of a (possibly exact) method to solve smaller "versions" of the original optimization problem, *i.e.*, to find the optimal solution on a limited portion of the solution space.

Given a problem P belonging to the class of $\mathcal{NP}$-hard problems, a very large solution space $\mathcal{X}$, and an exact method $\mathcal{M}$ that could be used to solve problem P to optimality if $\mathcal{X}$ were not too large, the method receives as input an incumbent solution $\mathbf{x} \in \mathcal{X}$. The CM defines a "corridor" around the incumbent solution by imposing exogenous constraints on P and, therefore, by cutting out portions of the solution space $\mathcal{X}$. The nature of the exogenous constraints is problem and method specific; however, ideally, the resulting neighborhood built around $\mathbf{x}$, *i.e.*, $\mathcal{N}(\mathbf{x})$, should be such that it could be explored in (pseudo) polynomial time using method $\mathcal{M}$.

Let us now consider how the CM can be applied to the MLCLSP-CO. In the following, let us indicate with $\mathcal{X}$ the feasible space of the problem , while we indicate with $\mathcal{M}$ a branch and bound algorithm implemented within a MIP solver. Let us suppose that we are given an incumbent solution $(\mathbf{x}, \mathbf{y}, \mathbf{s}, \omega)^I$. We build a neighborhood around the incumbent solution by imposing the following constraint:

$$\sum_{j=1}^{J} \sum_{t=1}^{T} x_{jt}^I x_{jt} \geq \delta n_i \tag{13}$$

where $\delta \in [0, 1]$ is a parameter used to define the *corridor width*, and $n_i = \sum_{j=1}^{J} \sum_{t=1}^{T} x_{jt}^I$ accounts for the number of setups in the incumbent solution. Therefore, the *corridor constraint* establishes a lower bound in the number of setups of a possible solution. It is worth noting, though, that such constraint is "flexibly" fixing some of the setup variables to 1. Constraint (13) cuts out of the solution space $\mathcal{X}$ all those (originally feasible) solutions that do not fix to 1 at least δ% of the setup variables currently set to 1 in the incumbent solution. The difference between arbitrarily fixing to 1 δ% of the variables in MLCLSP-CO and the introduction of the *corridor constraint* lies in the fact that, with the *corridor constraint*, we let the MIP solver the flexibility to choose which variables have to be fixed to 1.

It is easy to see that, by increasing (decreasing) the value of δ, we reduce (enhance) the corridor width and, consequently, the size of the search space. Constraint (13) can also be seen as a type of distance, or diversity, measure and, therefore, the addition of such constraint to MLCLSP-CO generates a neighborhood around the incumbent solution $(\mathbf{x}, \mathbf{y}, \mathbf{s}, \omega)^I$, defined as:

$$\mathcal{N}_\delta(\mathbf{x}^I) = \left\{ (\mathbf{x}, \mathbf{y}, \mathbf{s}, \omega) \in \mathcal{X} : \sum_{j=1}^{J} \sum_{t=1}^{T} x_{jt}^I x_{jt} \geq \delta n_i \right\} \tag{14}$$

In Figure 1 we present an outline of the overall algorithm. The algorithm is iteratively repeated using different incumbent solutions for every iteration. We create and store each incumbent solution in Ω. Such set is used to check that no incumbent solution is fed to the corridor method more than once.

Each iteration of the corridor method (step S2) stops when one of the two stopping criteria is met, *i.e.*, either a maximum running time for the exploration of the neighborhood is reached, or a maximum number of new feasible solutions have been collected. It is worth noting that the incumbent solution is used as starting point by the MIP solver and its objective function value as a cut. Therefore, every new feasible solution found by the MIP solver within the corridor has an objective function value better than the incumbent itself.

The overall algorithm terminates once a predefined stopping criterion is reached, *i.e.*, (i) a maximum running time; or (ii) a maximum number of incumbent solutions in Ω.

S1. Initialization

- Generation of the incumbent solution. A simple scheme (Section 4) is used to generate a new feasible incumbent $(\mathbf{x}, \mathbf{y}, \mathbf{s}, \omega)^I$, *i.e.*, a feasible solution such that $(\mathbf{x}, \mathbf{y}, \mathbf{s}, \omega)^I \notin \Omega$.
- Add $(\mathbf{x}, \mathbf{y}, \mathbf{s}, \omega)^I$ to Ω.

S2. Corridor Method $\left((\mathbf{x}, \mathbf{y}, \mathbf{s}, \omega)^I, \delta, \delta^m\right)$

- Add the *corridor constraint* (13) and solve the resulting MIP problem.
- Stopping criteria of the MIP solver:
  - maximum running time
  - maximum number feasible solutions
- $(\mathbf{x}, \mathbf{y}, \mathbf{s}, \omega)^c$ is the best solution found in $\mathcal{N}_\delta(\mathbf{x}^I)$

S3. Update $\left((\mathbf{x}, \mathbf{y}, \mathbf{s}, \omega)^I, \delta, \delta^m\right)$

- If $(\mathbf{x}, \mathbf{y}, \mathbf{s}, \omega)^c$ is better than $(\mathbf{x}, \mathbf{y}, \mathbf{s}, \omega)^I$, then set $(\mathbf{x}, \mathbf{y}, \mathbf{s}, \omega)^c$ as incumbent and go to **S2.**
- If no solution better than the incumbent has been found:
  - If $0.9\delta \geq \delta^m$, set $\delta \leftarrow 0.9\delta$ (enhance the corridor) and go to **S2.**
  - If a minimum δ value has been reached (δ^m), stop the corridor method and go to **S1.**

Fig. 1. Outline of the proposed algorithm

4 Incumbent Solution Generation: A Metaheuristic Scheme

As presented in Section 3, one of the main ingredients of the proposed algorithm is the generation of an incumbent solution, around which a corridor is going to be build. In this section, we illustrate how a metaheuristic method can be used to generate a set of incumbent solutions for the corridor method.

The *cross entropy method* (CE) has been presented by [7] and its application to the CLSP has been proposed by [4]. The CE can be used to generate a population of N binary matrices $X_1, \ldots, X_N$, to identify the periods in which each item is setup. Given a binary matrix X_i, we fix the corresponding x_{jt} variables accordingly, hence fixing periods in which a setup is scheduled. The problem obtained after fixing the setup variables is composed of continuous variables ($\mathbf{s}$ and $\mathbf{y}$) and binary variables (ω) strongly connected to the value of the setup variables, as established by constraints (6)– (8). Such restricted problem can quickly be solved to (near) optimality using a standard MIP solver.

If the restricted problem obtained after fixing the setup variables has a feasible solution, we associate the objective function value of the best solution found to the corresponding stochastic matrix X_i. Otherwise, if the derived problem is infeasible, we simply discard the stochastic matrix X_i and we generate a new stochastic matrix.

Let us define a probability matrix $P = \{p_{jt}\}$, where p_{jt} is the probability that a setup is scheduled for item j in period t. In order to exploit the stochastic nature of the proposed approach, we generate a *population* of X_i of size N, all drawn under the same initial probability matrix P. Inspired by the spirit of CE, we then use the "Maximum Likelihood Estimator" method to generate a new probability matrix P^1 that better describes the best individuals of the current population. Therefore, we update the set of probabilities p_{jt} in order to reflect how likely it is that, in a high-quality solution, a setup of item j in period t is scheduled. Once a new probability matrix P^1 is obtained, a new population of size N can be drawn under such matrix. Hopefully, such matrix better describes high quality solutions obtained in the previous generation and, therefore, the chances of obtaining high quality individuals under the new matrix are higher.

This process of "probability matrix update" and "population generation" can be iterated until either the P matrix converges to a binary matrix (therefore, the process converges to a unique solution) or a pre-specified maximum number of iterations has been reached.

The "Maximum Likelihood Estimator" method is used to modify probabilities p_{jt} in such a way that the new stochastic matrix better reflects the chances of obtaining high quality solutions. Let us assume that, based upon the current stochastic matrix P^k, we have generated a population of size N, *i.e.*, $X_1, \ldots, X_N$. Let us now find, within the current population, the objective function value of the $(1-\rho)\%$ quantile, *i.e.*, the value γ for which $\rho\%$ of the population have a better objective function value and $(1-\rho)\%$ of the population have a worse objective function value.

We modify the probability matrix using the following updating rule:

$$\hat{p}_{jt} = \frac{\sum_{i=1}^{N} x_{jt}^{i} \times I_{\{f(X_i) \leq \gamma\}}}{\rho N} \tag{15}$$

where x_{jt}^{i} indicates component (j,t) of matrix X_i, $f(X_i)$ is the objective function value of the i^{th} solution $(X_i, \omega_i, \mathbf{y}_i, \mathbf{s}_i)$, and $I_{\{f(X_i) \leq \gamma\}}$ is the indicator function, whose value is 1 if $f(X_i) \leq \gamma$ and 0 otherwise.

Remark. As pointed out by [7], in order to prevent the CE from converging too fast to a suboptimal solution, a *smoothing factor* α (typically $0.7 \leq \alpha \leq 0.9$) could be used in the updating rule. Therefore, to foster a more thorough exploration of the solution space, at each iteration k we use the following updating rule:

$$p_{jt}^{k+1} = \alpha \hat{p}_{jt} + (1-\alpha) p_{jt}^{k}. \tag{16}$$

5 Computational Results

In order to evaluate the quality of the proposed approach, we ran the algorithm on benchmark instances from the literature. More specifically, we solved some of the instances of Tempelmeier et al. [14]. In this section, we present results on the first and fourth classes of such instances. The two classes had been chosen firstly, to see for a simpler class to which extent the approach could provide meaningful results, and secondly, to see whether those results are also confirmed when considering the hardest instances among those literature data. Class one covers one general and one assembly product structure, 10 products, 3 resources, and 4 periods. A total of 480 instances are contained in this class. Class four is characterized by 20 products, 6 resources, and 16 periods and composed of 240 instances. These instances have been downloaded from the author's web page, and recreated following the instructions provided by the authors themselves, as indicated in [14].

All computational experiments have been carried out on a 2.0 GHz Pentium 4 Workstation with 2Gb of RAM running Linux. The algorithm has been coded in C++ and compiled with the GNU g++4.3 compiler.

The values of the algorithmic parameters N, ρ, and α have been determined using the response surface methodology [1], as presented in [4]. The MIP solver used is Cplex 11. As reported by [14], many of these instances could not be solved to optimality using a MIP solver within a time limit of 3600 seconds.

The stopping criteria for both the overall algorithm and the corridor method are (These values have been chosen such as to find a balance between solution quality and CPU time.):

- Maximum running time of the overall algorithm: 300 seconds.
- Maximum number of incumbent solutions in Ω: 10.
- Maximum running time of the constrained MIP (corridor method): 10 seconds.
- Maximum number of new solutions visited within a corridor: 10.
- Initial value of δ : 0.1.
- Maximum value of δ : 0.2.

In Table 1 we summarize the computational results on these 720 benchmark instances. The first three columns describe the instance, in terms of number of items, number of periods and number of resources (*e.g.*, machines), respectively. Column four reports the results of the algorithm presented in [14]. The computational time of these authors is provided as an average over all the instances, and reported as being 0.15 seconds. Since many of the benchmark instances could not be solved to optimality within the established time limit, we measure the solution quality computing the dual gap, as in [14]. Dual gap and computational time are used to measure the algorithmic performance of the proposed algorithm. The lower bound values used to compute the distance from optimality have been obtained solving the MLCLSP-CO with Cplex 11 for 3600 seconds. Therefore, for each instance of the class, we computed the optimality gap as

$$\gamma = \frac{z^H - lb}{lb} \times 100,$$

Table 1. Results on first and fourth classes of benchmark instances. Time measured in CPU seconds and averaged over each instance class.

Instance Details			Tempelmeier et al.	Proposed Algorithm	
No. Items	No. Periods	No. Machines	Gap	Gap	Time
10	4	3	1.39	0.10	2.7
20	6	16	21.90	8.56	59.3

where z^H is the objective function value of the heuristic solution found by the proposed algorithm, and lb is the best lower bound obtained by Cplex.

From the table we can confirm the robustness and the effectiveness of the proposed scheme in dealing with these classes of instances of the MLCLSP-CO. If we compare the running time of the proposed algorithm with that of [14], we can observe that the average time of the proposed algorithm is of 31 seconds as opposed to 0.15 seconds of [14]. However, such an increase in computational time allows to sensibly reduce the gap and, therefore, to find better quality solutions.

6 Conclusions

We have presented a math-heuristic algorithm for the multi-item multi-period capacitated lot sizing problem with carryover, which is a variation of the standard lot sizing problem where some scheduling information is included in the final solution. More specifically, given any two consecutive periods, it is possible to introduce information about the last item scheduled in one period and the first item scheduled in the subsequent period. Therefore, by extending a setup of an item over two consecutive periods, a setup in the second period can be saved, hence leading to more realistic solutions.

We have proposed a three-step algorithm, where (i) an incumbent solution is generated through the use of the cross entropy method; (ii) an iterative corridor method algorithm is applied starting from the incumbent solution and moving in the direction dictated by the best solution found in each corridor; and (iii) an updating phase, in which either the corridor is enhanced or the method is reapplied starting from a different portion of the original search space.

The method has been tested on a limited set of benchmark instances, more precisely, on "extreme" classes one and four of a larger set of benchmark instances from [14]. Our results show that the algorithm is effective, both from the point of view of the solution quality as well as from the perspective of computational time.

References

1. Box, G., Wilson, K.: On the Experimental Attainment of Optimum Conditions. Journal of the Royal Statistical Society B-13, 1–45 (1951)

2. Buschkühl, L., Sahling, F., Helber, S., Tempelmeier, H.: Dynamic capacitated lot-sizing problems: a clasification and review of solution approaches. OR Spectrum (2008), doi:10.1007/s00291-008-0150-7
3. Caserta, M., Ramirez, A., Voß, S., Moreno, R.: A hybrid algorithm for the multi level capacitated lot sizing problem with setup carryover. In: Voß, S., Pahl, J., Schwarze, S. (eds.) Logistik Management, pp. 123–138. Physica, Heidelberg (2009)
4. Caserta, M., Quiñonez, E.: A Cross Entropy-Lagrangean Hybrid Algorithm for the Multi-Item Capacitated Lot-Sizing Problem with Setups. Computers and Operations Research 36(2), 530–548 (2009)
5. Caserta, M., Voß, S., Sniedovich, M.: Applying the corridor method to a block relocation problem. OR Spectrum (2009), doi:10.1007/s00291-009-0176-5
6. Caserta, M., Voß, S.: A corridor method-based algorithm for the pre-marshalling problem. In: Giacobini, M., et al. (eds.) EvoWorkshops 2009. LNCS, vol. 5484, pp. 788–797. Springer, Heidelberg (2009)
7. De Boer, P., Kroese, D.P., Mannor, S., Rubinstein, R.Y.: A tutorial on the cross-entropy method. Annals of Operations Research 134(1), 19–67 (2005)
8. Maes, J., McClain, J., Van Wassenhove, L.: Multilevel capacitated lotsizing complexity and LP-based heuristics. European Journal of Operational Research 53, 131–148 (1991)
9. Quadt, D., Kuhn, H.: Capacitated lot-sizing with extensions: a review. 4OR: A Quarterly Journal of Operations Research 6(1), 61–83 (2008)
10. Sahling, F., Buschkühl, L., Tempelmeier, H., Helber, S.: Solving a multi-level capacitated lot sizing problem with multi-period setup carry-over via a fix-and-optimize heuristic. Computers and Operations Research 36(9), 2546–2553 (2009)
11. Sniedovich, M., Voß, S.: The Corridor Method: a Dynamic Programming Inspired Metaheuristic. Control and Cybernetics 35(3), 551–578 (2006)
12. Suerie, C., Stadtler, H.: The capacitated lot-sizing problem with linked lot sizes. Management Science 49(8), 1039–1054 (2003)
13. Tempelmeier, H., Destroff, M.: A lagrangian-based heuristic for dynamic multilevel multiitem constrained lotsizing with setup times. Management Science 42(5), 738–757 (1996)
14. Tempelmeier, H., Buschkühl, L.: A heuristic for the dynamic multi-level capacitated lotsizing problem with linked lotsizes for general product structures. OR Spectrum 31, 385–404 (2009)

Author Index