Mehmet A. Orgun
John Thornton (Eds.)

LNAI 4830

AI 2007: Advances in Artificial Intelligence

20th Australian Joint Conference on Artificial Intelligence
Gold Coast, Australia, December 2007
Proceedings

Springer

Lecture Notes in Artificial Intelligence 4830

Edited by J. G. Carbonell and J. Siekmann

Subseries of Lecture Notes in Computer Science

Mehmet A. Orgun   John Thornton (Eds.)

AI 2007: Advances in Artificial Intelligence

20th Australian Joint Conference on Artificial Intelligence
Gold Coast, Australia, December 2-6, 2007
Proceedings

Springer

Series Editors

Jaime G. Carbonell, Carnegie Mellon University, Pittsburgh, PA, USA
Jörg Siekmann, University of Saarland, Saarbrücken, Germany

Volume Editors

Mehmet A. Orgun
Macquarie University
Department of Computing
Sydney, NSW 2109, Australia
E-mail: mehmet@ics.mq.edu.au

John Thornton
Griffith University
School of Information and Communication Technology
Gold Coast, Qld 4222, Australia
E-mail: j.thornton@griffith.edu.au

Library of Congress Control Number: 2007939893

CR Subject Classification (1998): I.2, F.4.1, H.3, H.2.8, F.1

LNCS Sublibrary: SL 7 – Artificial Intelligence

ISSN	0302-9743
ISBN-10	3-540-76926-9 Springer Berlin Heidelberg New York
ISBN-13	978-3-540-76926-2 Springer Berlin Heidelberg New York

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper   SPIN: 12195497   06/3180   5 4 3 2 1 0

Preface

This volume contains the papers presented at AI 2007: The 20th Australian Joint Conference on Artificial Intelligence held during December 2–6, 2007 on the Gold Coast, Queensland, Australia.

AI 2007 attracted 194 submissions (full papers) from 34 countries. The review process was held in two stages. In the first stage, the submissions were assessed for their relevance and readability by the Senior Program Committee members. Those submissions that passed the first stage were then reviewed by at least three Program Committee members and independent reviewers. After extensive discussions, the Committee decided to accept 60 regular papers (acceptance rate of 31%) and 44 short papers (acceptance rate of 22.7%). Two regular papers and four short papers were subsequently withdrawn and are not included in the proceedings.

AI 2007 featured invited talks from four internationally distinguished researchers, namely, Patrick Doherty, Norman Foo, Richard Hartley and Robert Hecht-Nielsen. They shared their insights and work with us and their contributions to AI 2007 were greatly appreciated. AI 2007 also featured workshops on integrating AI and data-mining, semantic biomedicine and ontology. The short papers were presented in an interactive poster session and contributed to a stimulating conference.

It was a great pleasure for us to serve as the Program Co-chairs of AI 2007. We would like to thank all the Senior Program Committee members for their extremely hard work in the two-stage review process and the Program Committee members and the reviewers for the timely return of their comprehensive reviews. Without their help and contributions, it would have been impossible to make decisions and produce such high-quality proceedings. We also would like to acknowledge the contributions of all the authors of 194 submissions. The EasyChair conference management system was used (www.easychair.org) in all stages of the review process and in the generation of the proceedings; it made our life much easier.

We would like to thank the Conference Co-chairs, Abdul Sattar and Vladimir Estivill-Castro of Griffith University for their guidance, and the local Organizing Co-chairs Michael Blumenstein and Guido Governatori for making sure that the conference ran smoothly. Special thanks go to Natalie Dunstan and Vicky Wheeler for supporting the Committees so effectively.

We also would like to thank the following organizations for their generous sponsorship of AI 2007: Griffith University, National ICT Australia, the University of Queensland, Queensland University of Technology, Bond University, the Australian Computer Society and the Gold Coast City Council.

December 2007

Mehmet A. Orgun
John Thornton

Organization

AI 2007 was hosted and organized by Griffith University, Gold Coast, Australia. The conference was held at the Holiday Inn Hotel, Surfers Paradise, December 2–6, 2007.

Conference Committee

Conference Co-chairs

Abdul Sattar (Griffith University)
Vladimir Estivill-Castro (Griffith University)

Program Co-chairs

Mehmet A. Orgun (Macquarie University)
John Thornton (Griffith University)

Local Organizing Co-chairs

Michael Blumenstein (Griffith University)
Guido Governatori (University of Queensland)

Organizing Committee

Udantha Abeyratne (University of Queensland)
Marcus Randall (Bond University)
Frederic Maire (Queensland University of Technology)

Senior Program Committee

Dan Corbett (DARPA, USA)
Stephen Cranefield (University of Otago, New Zealand)
Robert Dale (Macquarie University, Australia)
Adnan Darwiche (UCLA, USA)
James Delgrande (Simon Fraser University, Canada)
David Dowe (Monash University, Australia)
Vladimir Estvill-Castro (Griffith University, Australia)
Michael Fisher (University of Liverpool, UK)
Aditya Ghose (University of Wollongong, Australia)
Randy Goebel (University of Alberta, Canada)
Guido Governatori (University of Queensland, Australia)
Fikret Gürgen (Boḡaziçi University, Turkey)
Byeong Ho Kang (University of Tasmania, Australia)

Achim Hoffmann (University of New South Wales, Australia)
Kevin Korb (Monash University, Australia)
Jerome Lang (IRIT, France)
Jimmy Lee (Chinese University of Hong Kong, China)
Fangzhen Lin (HKUST, China)
John Lloyd (Australian National University, Australia)
Dickson Lukose (DL Informatique, Malaysia)
Michael Maher (National ICT Australia, Australia)
Abhaya Nayak (Macquarie University, Australia)
Ken Satoh (National Institute of Informatics, Japan)
Abdul Sattar (Griffith University, Australia)
Leon Sterling (University of Melbourne, Australia)
Markus Stumptner (University of South Australia, Australia)
Toby Walsh (National ICT Australia, Australia)
Geoff West (Curtin University, Australia)
Mary-Anne Williams (UTS, Australia)
Wayne Wobcke (University of New South Wales, Australia)
Xingdong Wu (University of Vermont, USA)
Roland Yap (National University of Singapore)
Chengqi Zhang (UTS, Australia)
Kang Zhang (University of Texas, USA)
Yan Zhang (UWS, Australia)

Program Committee

Murali Agastya, Australia
Rodrigo Agerri, UK
Damminda Alahakoon, Australia
Songül Albayrak, Turkey
Grigoris Antoniou, Greece
Mike Bain, Australia
Florence Bannay, France
Nick Bassiliades, Greece
Nicolas Beldiceanu, France
David Benn, Australia
Ghassan Beydoun, Australia
Alan Blair, Australia
Mikael Boden, Australia
Guido Boella, Italy
Alexander Bolotov, UK
Richard Booth, Thailand
Rafael Bordini, UK
Adi Botea, Australia
Pierre Boulos, Canada
Sebastian Brand, Australia
Lawrence Cavedon, Australia
Hei Chan, USA
Laiwan Chan, China
Chee-Fon Chang, Australia
Xiaoping Chen, China
Yin Chen, China
Yann Chevaleyre, France
Arthur Choi, USA
Aniruddha Dasgupta, Australia
John Debenham, Australia
Da Deng, New Zealand
Louise Dennis, UK
Grant Dick, New Zealand
Clare Dixon, UK
Atilla Elçi, Turkey
R. Cenk Erdur, Turkey
Valnir Ferreira Jr., Australia
Cesar Ferri, Spain
Lee Flax, Australia
Tim French, Australia

Sanjing Li, China
Xue Li, Australia
Alexei Lisitsa, UK
Wei Liu, Australia
Paolo Terenziani, Italy
Michael Thielscher, Germany
Peter Tischer, Australia
Paolo Torroni, Italy
Andre Trudel, Canada
Mirek Truszczynski, USA
Suzan Üsküdarlı, Turkey
Hans van Ditmarsch, New Zealand
Kirsten Venable, Italy
Murlikrishna Viswanathan, USA
Peter Wallis, UK
Kewen Wang, Australia
Lipo Wang, Singapore
Yisong Wang, Australia
Dianhui Wang, Australia
Peter Whigham, New Zealand
Insu Song, Australia
Olivier Spanjaard, France
Nicola Stokes, Australia
Ahmed Tawfik, Canada
William Wilson, Australia
Brendon Woodford, New Zealand
Dan Wu, Canada
Yang Xiang, Canada
Wai-Kiang Yeap, New Zealand
Bruno Zanuttini, France
Dongmo Zhang, Australia
Mengjie Zhang, New Zealand
Minjie Zhang, Australia
Shichao Zhang, Australia
Yuan-Lin Zhang, USA
Zili Zhang, Australia
Yi Zhou, Australia
Zhi-Hua Zhou, China
Sandra Zilles, Germany

Additional Reviewers

David Allen
Josep Argelich
Ching-man Au Yeung
Quan Bai
Omar Baqueiro
Antonis Bikakis
Gökhan Bilgin
Katarina Britz
Peter Busch
Hadrien Cambazard
Rapeeporn Chamchong
Mu-chen Chen
Federico Chesani
Andrew Dowell
Lan Du
Thibaut Feydy
Masabumi Furuhata
Ranadhir Ghosh
Roberto Grenna
Hakan Haberdar
Jose Hernandez-Orallo
Dennis Hooijmaijers
Tertia Horne
Eric Hsu
Jiehui Jiang
Huidong Jin
Yi Jin
Ioannis Katakis
Matthew Kitching
Kyriakos Kritikos
Sid Kulkarni
Richard Leibbrandt
Trent Lewis
Jiuyong Li
Chavalit Likitvivatanavong
Jing Liu
Andreas Lommatzsch
Daniela Marconi
Wolfgang Mayer
Georgios Meditskos
Nina Narodytska
Conor Nugent
Mehrdad Oveisi-Fordoei
F. Canan Pembe

Franz Pernkopf
Jordi Planes
Jantima Polpinij
Jakob Puchinger
Saba Sajjadian
Hasim Sak
Horst Samulowitz
Anoop Sarkar
Stefan Schlobach
Jochen Schmidt
Murat Semerci
Qi Shan
Chenxi Shao
Jonathan Shapiro
Sajjad Siddiqi
Eric Sinzinger
Elankayer Sithirasenan
Zhiwei Song
Sergio Tessaris
Dmitry Tishkovsky
Rodney Topor
Grigorios Tsoumakas
Aydın Ulaş
Zhe Wang
Wilson Wong
Xiaoqun Zhang
Yuanlin Zhang
Zhongwei Zhang

Table of Contents

A UAV Search and Rescue Scenario with Human Body Detection and Geolocalization*

Patrick Doherty and Piotr Rudol

Department of Computer and Information Science
Linköping University, SE-58183 Linköping, Sweden
{patdo,pioru}@ida.liu.se

Abstract. The use of Unmanned Aerial Vehicles (UAVs) which can operate autonomously in dynamic and complex operational environments is becoming increasingly more common. The UAVTech Lab [1], is pursuing a long term research endeavour related to the development of future aviation systems which try and push the envelope in terms of using and integrating high-level deliberative or AI functionality with traditional reactive and control components in autonomous UAV systems. In order to carry on such research, one requires challenging mission scenarios which force such integration and development. In this paper, one of these challenging emergency services mission scenarios is presented. It involves search and rescue for injured civilians by UAVs. In leg I of the mission, UAVs scan designated areas and try to identify injured civilians. In leg II of the mission, an attempt is made to deliver medical and other supplies to identified victims. We show how far we have come in implementing and executing such a challenging mission in realistic urban scenarios.

1 Introduction

The use of Unmanned Aerial Vehicles (UAVs) which can operate autonomously in dynamic and complex operational environments is becoming increasingly more common. While the application domains in which they are currently used are still predominantly military in nature, we can expect to see widespread usage in the civil and commercial sectors in the future as guidelines and regulations are developed by aeronautics authorities for insertion of UAVs in civil airspace.

One particularly important application domain where UAVs could be of great help in the future is in the area of catastrophe assistance. Such scenarios include natural disasters such as earthquakes or tsunamis or man-made disasters caused by terrorist activity. In such cases, civil authorities often require a means of acquiring an awareness of any situation at hand in real-time and the ability to monitor the progression of events in catastrophe situations. Unmanned aerial

* This work is supported in part by the National Aeronautics Research Program NFFP04 S4202 and the Swedish Foundation for Strategic Research (SSF) Strategic Research Center MOVIII and a Swedish Research Council grant.

[1] Autonomous Unmanned Aerial Vehicle Technologies Lab, Linköping University, Sweden, http://www.ida.liu.se/~patdo/auttek/

vehicles offer an ideal platform for acquiring the necessary situation awareness to proceed with rescue and relief in many such situations. It is also often the case that there is no alternative in acquiring the necessary information because one would like to avoid placing emergency services personal in the line of danger as much as possible.

For a number of years, The Autonomous Unmanned Aerial Vehicle Technologies Lab (UAVTech Lab) at Linköping University, Sweden, has pursued a long term research endeavour related to the development of future aviation systems in the form of autonomous unmanned aerial vehicles [1,2]. The focus has been on both high autonomy (AI related functionality), low level autonomy (traditional control and avionics systems), and their integration in distributed software architectural frameworks [3] which support robust autonomous operation in complex operational environments such as those one would face in catastrophe situations.

More recently, our research has moved from single platform scenarios to multi-platform scenarios where a combination of UAV platforms with different capabilities are used together with human operators in a mixed-initiative context with adjustable platform autonomy. The application domain we have chosen to pursue is emergency services assistance. Such scenarios require a great deal of cooperation among the UAV platforms and between the UAV platforms and human operators.

The paper is structured in the following manner. In section 2, we introduce the emergency services scenario. In section 3, we describe the UAV platforms used in the scenario. In section 4, we consider the body identification and geo-location phase of the mission in more detail and in section 5, we consider the supply delivery phase of the mission in more detail.

2 An Emergency Service Scenario

On December 26, 2004, a devastating earthquake of high magnitude occured off the west coast off Sumatra. This resulted in a tsunami which hit the coasts of India, Sri Lanka, Thailand, Indonesia and many other islands. Both the earthquake and the tsunami caused great devastation. During the initial stages of the catastrophe, there was a great deal of confusion and chaos in setting into motion rescue operations in such wide geographic areas. The problem was exacerbated by shortage of manpower, supplies and machinery. Highest priorities in the initial stages of the disaster were search for survivors in many isolated areas where road systems had become inaccessible and providing relief in the form of delivery of food, water and medical supplies.

Let's assume for a particular geographic area, one had a shortage of trained helicopter and fixed-wing pilots and/or a shortage of helicopters and other aircraft. Let's also assume that one did have access to a fleet of autonomous unmanned helicopter systems with ground operation facilities. How could such a resource be used in the real-life scenario described?

A pre-requisite for the successful operation of this fleet would be the existence of a multi-agent (UAV platforms, ground operators, etc.) software infrastructure for assisting emergency services in such a catastrophe situation. At the very least, one would require the system to allow mixed initiative interaction with multiple platforms and ground operators in a robust, safe and dependable manner. As far as the individual platforms are concerned, one would require a number of different capabilities, not necessarily shared by each individual platform, but by the fleet in total. These capabilities would include:

- the ability to scan and search for salient entities such as injured humans, building structures or vehicles;
- the ability to monitor or surveil these salient points of interest and continually collect and communicate information back to ground operators and other platforms to keep them situationally aware of current conditions;
- the ability to deliver supplies or resources to these salient points of interest if required. For example, identified injured persons should immediately receive a relief package containing food, medical and water supplies.

Although quite an ambitious set of capabilities, several of them have already been achieved to some extent using our experimental helicopter platforms, although one has a long way to go in terms of an integrated, safe and robust system of systems.

To be more specific in terms of the scenario, we can assume there are two separate legs or parts to the emergency relief scenario in the context sketched previously.

Leg I. In the first part of the scenario, it is essential that for specific geographic areas, the UAV platforms should cooperatively scan large regions in an attempt to identify injured persons. The result of such a cooperative scan would be a saliency map pinpointing potential victims, their geographical coordinates and sensory output such as high resolution photos and thermal images of potential victims. The resulting saliency map would be generated as the output of such a cooperative UAV mission and could be used directly by emergency services or passed on to other UAVs as a basis for additional tasks.

Leg II. In the second part of the scenario, the saliency map generated in Leg I would be used as a basis for generating a logistics plan for several of the UAVS with the appropriate capabilities to deliver food, water and medical supplies to the injured identified in Leg I. This of course would also be done in a cooperative manner among the platforms.

3 Hardware Platform

The UAVTech UAV platform [1] is a slightly modified Yamaha RMAX helicopter (Fig. 1). It has a total length of 3.6 m (including main rotor) and is powered by a 21hp two-stroke engine with a maximum takeoff weight of 95 kg. The on-board

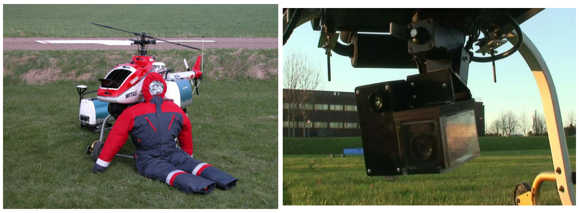


Fig. 1. The UAVTech UAV and the on-board camera system mounted on a pan-tilt unit

system contains three PC104 embedded computers. The primary flight control (PFC) system includes a Pentium III 700Mhz, a wireless Ethernet bridge, a GPS receiver, and several additional sensors including a barometric altitude sensor. The PFC is connected to the RMAX helicopter through the Yamaha Attitude Sensor (YAS) and Yamaha Attitude Control System (YACS), an image processing computer and a computer responsible for deliberative capabilities. The deliberative/reactive system (DRC) runs on the second PC104 embedded computer (Pentium-M 1.4GHz) and executes all high-end autonomous functionalities such as mission or path planning. Network communication between computers is physically realized with serial lines RS232C and Ethernet.

The image processing system (IPC) runs on the third PC104 embedded Pentium III 700MHz computer. The camera platform suspended under the UAV fuselage is vibration isolated by a system of springs. The platform consists of a Sony CCD block camera FCB-780P and a ThermalEye-3600AS miniature infrared camera mounted rigidly on a Pan-Tilt Unit (PTU) as presented in Fig. 1. The video footage from both cameras is recorded at a full frame rate by two miniDV recorders to allow processing after a flight.

4 Mission Leg I: Body Identification

The task of the 1st leg of the mission is to scan a large region with one or more UAVs, identify injured civilians and output a saliency map which can be used by emergency services or other UAVs. The technique presented uses two video sources (thermal and color) and allows for high rate human detection at larger distances then in the case of using the video sources separately with standard techniques. The high processing rate is essential in case of video collected onboard a UAV in order not to miss potential objects as a UAV flies over it. A thermal image is analyzed first to find human body sized silhouettes. Corresponding regions in a color image are subjected to a human body classifier which is configured to allow weak classifications. This focus of attention allows for maintaining a body classification at a rate up to 25Hz. This high processing rate allows for collecting statistics about classified humans and to prune false classifications of the "weak" human body classifier. Detected human bodies are geolocalized on a map which can be used to plan supply delivery. The technique presented has been tested on-board the UAVTech helicopter platform and is an important component in our research with autonomous search and rescue missions.

4.1 Image Processing

Video footage collected by a UAV differs substantially from images acquired on the ground and the use of standard techniques is not straight forward. For instance, both maximum and minimum speeds are determined by an aircraft's properties. Nevertheless, high flight speed is preferred in case of search and rescue applications. Therefore it is essential for the image processing algorithm to perform close to the full frame rate to process all frames of the video.

The algorithm we use takes as input two images (camera planes are assumed to be close to parallel to the earth plane) and the processing starts by analyzing the thermal image. The image is first thresholded to find regions of human body temperature. The shape of the regions is analyzed and those which do not resemble a human body (i.e. wrong ratio of minor and major axes of the fitted ellipse and incorrect area) are rejected. Additionally, regions which lie on the image border are rejected as they may belong to a bigger warm object. Once human body candidates are found in the thermal image, corresponding regions in the color image are calculated.

Computation of the corresponding region in the color image could be achieved by performing image registration or feature matching in both images. The former technique is too time consuming and the latter is infeasible because of mostly different appearance of features in color and thermal images. Here, a closed form solution is used which takes into account information about the UAV's state.

Computation of the corresponding region in the color image starts with calculating coordinates of a point T ($\tilde{v_T}$) whose projection is the pixel in the thermal image $\tilde{u_t}$ i.e.

$$\tilde{u_t} = P_t \tilde{v_T} \quad \tilde{u_t} \in \mathcal{P}^2 \quad \tilde{v_T} \in \mathcal{P}^3 \tag{1}$$

where P_t represents extrinsic and intrinsic parameters of the thermal camera. The general scheme of the problem is shown in Figure 2. A line equation with the direction vector $\tilde{v}_{cam}$ which goes through camera center through pixel $\tilde{u_t}$ and intersects the ground plane in point T is:

$$\tilde{v}_T - \tilde{v}_C = t \cdot \tilde{v}_{cam} \quad t \in \mathbb{R} \tag{2}$$

The ground plane is defined by the point G($\tilde{v}_G$) and the normal vector $\tilde{n}$ which is the down component of the NED (North, East, Down) frame:

$$(\tilde{v}_T - \tilde{v}_G) \cdot \tilde{n} = 0 \tag{3}$$

Finally, the vector $\tilde{v}_T$ which describes the point of intersection of a ray of light going through the camera center and the pixel of the target can be calculated according to:

$$\tilde{v}_T = \tilde{v}_C + \frac{(\tilde{v}_G - \tilde{v}_C) \cdot \tilde{n}}{\tilde{v}_{cam} \cdot \tilde{n}} \cdot \tilde{v}_{cam} \tag{4}$$

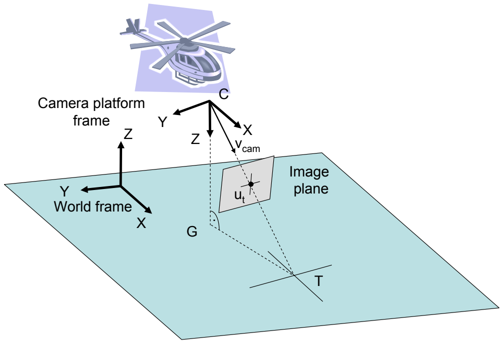


Fig. 2. Calculation of a target coordinates

In order to calculate $\tilde{v}_{cam}$ the vector along the X axis of the camera frame must be expressed in the world coordinate frame. This transformation can be expressed as:

$$^{w}\tilde{v}_{cam} = P_{heli} P_{ptu} P_{p} \begin{pmatrix} 1 & 0 & 0 \end{pmatrix}^{T} \tag{5}$$

where P_p describes the transformation depending on the undistorted pixel position $\tilde{u_t}$. Matrix P_{ptu} is built to represent a transformation introduced by the pan-tilt unit. P_{heli} represents the attitude of the UAV and is built up from roll, pitch and yaw angles delivered by the YAS system.

The method presented can be extended to relax the flat world assumption. The point T can be found by performing ray-tracing along the line described by equation Eq. 2 to find the intersection with the ground elevation map.

Calculated world position can additionally be checked against the on-board geographic information database to verify whether the calculated point is valid. Depending on the situation, certain positions can be excluded from the map. If the world position is accepted, its projection is calculated for the color camera using the following formula:

$$\tilde{u}_c = P_c \tilde{v}_T \quad \tilde{u}_c \in \mathcal{P}^2 \quad \tilde{v} \in \mathcal{P}^3 \tag{6}$$

where P_c constitutes the matrix encoding intrinsic and extrinsic parameters of the color camera.

4.2 The Classifier

Once the corresponding pixel in the color image is identified, a sub-window with the pixel $\tilde{u}_c$ in the center is selected and it is subjected to an object detector first suggested by [11]. The work was a basis for several improvements, one of which was presented in [9]. One of these included extending the original feature set which is presented in Fig. 3.

The classifier which is in fact a cascade of boosted classifiers working with Haar-like features requires training with a few hundred positive and negative examples. During learning the structure of a classifier is learned using boosting.

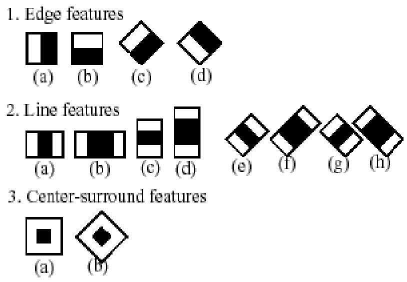


Fig. 3. Leinhart's extended set of available features

The use of a cascade of classifiers allows for dramatic speed up of computations by skipping negative instances and only computing features with high probability for positive classification. The speed up comes from the fact that the classifier, as it slides a window at all scales, works in stages and is applied to a region of interest until at some stage the candidate is rejected or all the stages are passed. This way, the classifier quickly rejects subregions which most probably do not include features needed for positive classification (i.e. background processing is quickly terminated). The classifier works with features which can be quickly extracted using intermediate image representations - integral images. The reason for working with features instead of pixel intensities is that features encode knowledge about the domain, which is difficult to learn from raw input data. The features encode the existence of oriented contrasts between regions of an image. The Haar-like features used here can be calculated at any position and any scale in constant time using only eight look-ups in the integral image.

The classifier used in this work is a part of the Open Source Computer Vision Library [10] and the trained classifier for upper-, lower- and full human body is a result of [8]. The trained classifier is best suited for pedestrian detection in frontal and backside views which is exactly the type of views a UAV has when flying above the bodies lying on the ground.

Since the body classifier is configured to be "relaxed" it delivers sporadic false positive classifications. To counter for most of them the following method is used to prune the results. Every salient point in the map has two parameters which are used to calculate certainty of a location being a human body: T_{frame} which describes the amount of time a certain location was in the camera view and T_{body} which describes the amount of time a certain location was classified as a human body. The certainty factor is calculated as follows:

$$p_{body}(loc_i) = \frac{T_{body}}{T_{frame}} \tag{7}$$

A location is considered a body if $p_{body}(loc_i)$ is larger than a certain threshold (e.g. 0.5 during the flight tests) and T_{frame} is larger than a desired minimal observation time. Locations are considered equal if geographical distance between them is smaller then a certain threshold (depending on the geolocation accuracy) and the final value of a geolocalized position is an average of the observations (c.f. Section 4.4).

4.3 Experimental Setup

A series of flight tests were performed in southern Sweden at an emergency services training center used by the Swedish Rescue Services Agency to train fire, police and medical personnel. Flight tests were performed over varied terrain such as asphalt and gravel roads, grass, trees, water and building roof tops which resulted in a variety of textures in the images. Two UAVs were used over a search area of 290x185 meters. A total of eleven bodies (both human and dummies with close to human temperature) were placed in the area. The goal of the mission was to generate a saliency map. The general mission plan is shown in Fig. 4. Before take-off, one of the UAVs was given an area to scan (dashed

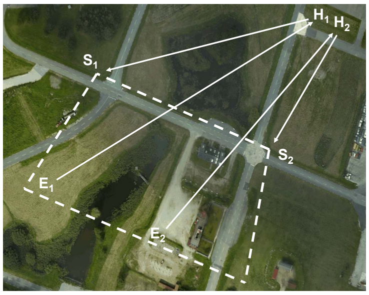


Fig. 4. Mission overview

line polygon). It then delegated part of the scanning task to another platform, generating sub-plans for itself and the other platform. The mission started with a simultaneous autonomous take-off at positions H_1 and H_2 and the UAVs flew to starting positions S_1 and S_2 for scanning. Throughout the flights, saliency maps were incrementally constructed until the UAVs reached their ending positions E_1 and E_2. The UAVs then returned to their respective take-off positions for a simultaneous landing. The mission took approximately ten minutes to complete and each UAV traveled a distance of around 1km.

4.4 Experimental Results

The algorithm found all eleven bodies placed in the area. The saliency map generated by one of the helicopters is shown in Fig. 5. The images of identified objects are presented in Fig. 6. Several positions were rejected as they were not observed long enough (i.e. 5 seconds). Images 7, 9, and 14 present three falsely identified objects.

The accuracy of the body geolocation calculation was performed by measuring GPS (without differential correction) positions of bodies after an experimental

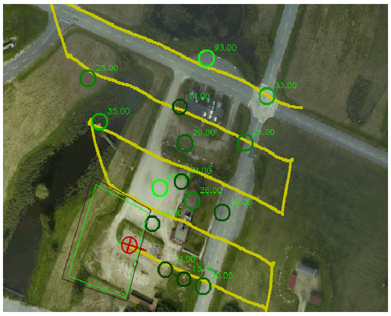


Fig. 5. Flight path and geolocated body positions

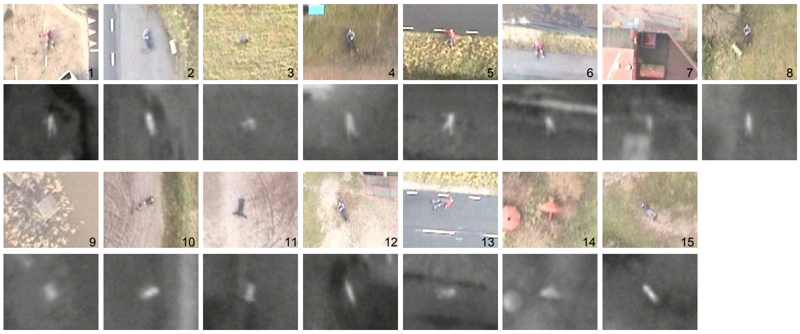


Fig. 6. Images of classified bodies. Corresponding thermal images are placed under color images.

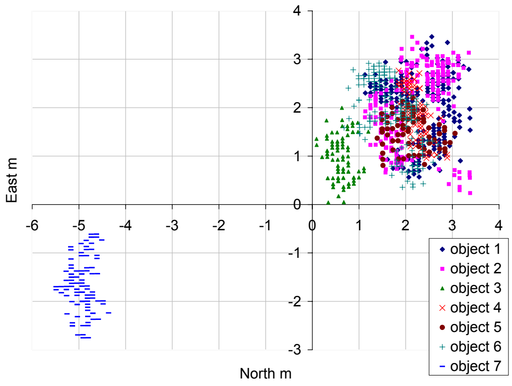


Fig. 7. Geolocation error for multiple objects

flight. The accuracy of the system is sufficient for the application of delivering supplies to the detected humans. Figure 7 presents the error measurement for seven geolocated objects. The measurement has a bias of approximately two meters in both east and north directions. It is a sum of errors in GPS measurement, accuracy of the camera platform mounting, PTU measurement, and camera calibration inaccuracies. The spread of measurement samples of approximately 2.5 meters in both east and north directions is caused by the error of attitude measurement, the system of springs in the camera platform, the flat ground assumption, and time differences between UAV state estimate, PTU angle measurement and image processing result acquisition. A large geolocation error of object 7 is caused by erroneous GPS measurement. Object 7 was located on a metal foot-bridge and the GPS antenna during static measurement was additionally partially occluded by metal railings. The noise on the measurement however is consistent with the rest of the objects.

5 Mission Leg II: Package Delivery

After successful completion of leg I of the mission scenario, we can assume that a saliency map has been generated with geo-located positions of the injured civilians. In the next phase of the mission, the goal is to deliver configurations of medical, food and water supplies to the injured. In order to achieve this leg of the mission, one would require a task planner to plan for logistics, a motion planner to get one or more UAVS to supply and delivery points and an execution monitor to monitor the execution of highly complex plan operators. Each of these functionalities would also have to be tightly integrated in the system. These components are described in section 5.1

Currently, we have developed this mission leg primarily in simulation with hardware-in-the-loop. Our avionics boxes are coupled directly to a simulator and execute all functionalities necessary for completion of the mission in the actual hardware we fly missions with. A physical winch system for picking up and putting down packages is currently under development.

For these logistics missions, we assume the use of one or more UAVs with diverse roles and capabilities. Initially, we assume there are [illegible] injured body locations, several supply depots and several supply carrier depots (see figure 8).

5.1 Planning, Execution and Monitoring

Figure 9 shows part of our UAV system architecture, with an emphasis on those components that are the most relevant for planning, execution, and execution monitoring.

At the top of the center column is the [illegible] which given a mission request, calls [illegible] [6,7], a knowledge processing middleware, to acquire essential information about the current contextual state of the world or the UAV's own internal states. Together with a domain specification and a goal specification related to the logistics scenario, this information is fed to [illegible] [4,5], a logic-based task planner which outputs a plan that will achieve the designated goals, under the assumption that all actions succeed and no failures occur. Such a plan can also be automatically annotated with global and/or operator-specific

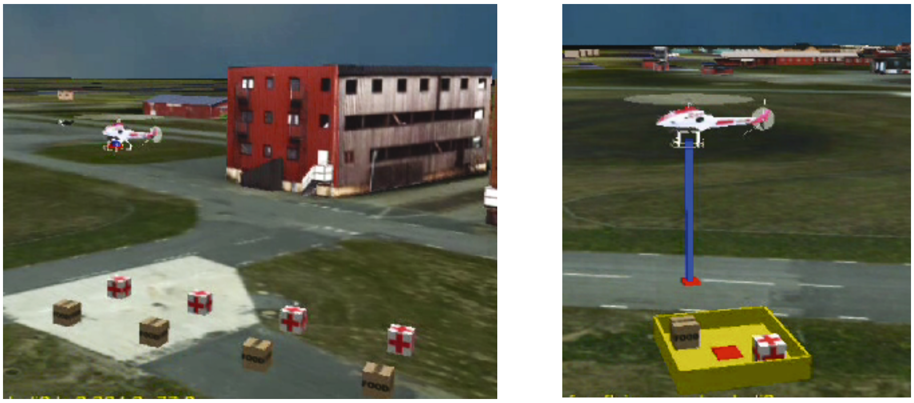


Fig. 8. A Supply Depot (left) and a Carrier Depot (right)

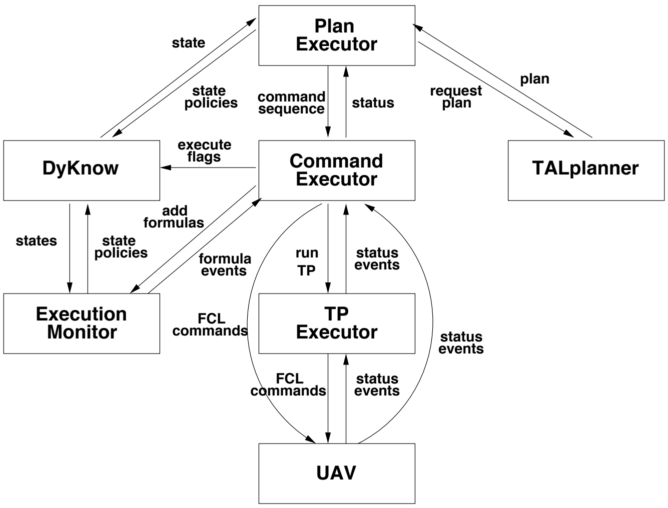


Fig. 9. System Architecture Overview

conditions to be monitored during execution of the plan by an execution monitor in order to relax the assumption that no failures can occur. Such conditions are expressed as temporal logical formulas and evaluated on-line using formula progression techniques. This execution monitor notifies the plan executor when actions do not achieve their desired results and one can then move into a plan repair phase.

The plan executor translates operators in the high-level plan returned by TALplanner into lower level command sequences which are given to the [illegible] [illegible]. The command executor is responsible for controlling the UAV, either by directly calling the functionality exposed by its lowest level Flight Command

Language (FCL) interface or by using so called [illegible] (TPs) through the [illegible] subsystem. The TP Executor is part of the Modular Task Architecture (MTA) [3], which is a reactive system designed in the procedure-based paradigm and developed for loosely coupled heterogeneous systems. A task is a behavior intended to achieve a goal in a limited set of circumstances. A task procedure is the computational mechanism that achieves this behavior. The TPs have the ability to use deliberative services, such as the task planner described above or motion planners [12,13], in a reactive or contingent manner and to use traditional control services in a reactive or contingent manner and thereby integrate deliberation and reaction.

During plan execution, the command executor adds formulas to be monitored to the [illegible]. DyKnow continuously sends information about the development of the world in terms of state sequences to the monitor, which uses a progression algorithm to partially evaluate monitor formulas. If a violation is detected, this is immediately signaled as an event to the command executor, which can suspend the execution of the current plan, invoke an emergency brake command, optionally execute an initial recovery action, and finally signal new status to the plan executor. The plan executor is then responsible for completing the recovery procedure.

The fully integrated system is implemented on our UAVs and can be used onboard for different configurations of the logistics mission described in Leg II of the larger mission. The simulated environments used are in urban areas and quite complex. Plans are generated in the millisecond to seconds range using TALplanner and empirical testing shows that this approache is promising in terms of integrating high-level deliberative capability with lower-level reactive and control functionality.

6 Conclusions

We have described a realistic emergency services scenario and shown how far we have come in the deployment of autonomous UAV systems which require the use of deliberative, reactive and control capabilities in a highly integrated and time-constrained context. Currently, we are developing a winch system for the RMAX which will be used to deliver supplies of the type described in leg II of the scenario. We are also refining the body identification algorithms and developing a framework for cooperation based on the use of delegation of goals and action sequences.

References

1. Doherty, P.: Advanced research with autonomous unmanned aerial vehicles. In: Proceedings on the 9th International Conference on Principles of Knowledge Representation and Reasoning (2004)
2. Doherty, P.: Knowledge representation and unmanned aerial vehicles. In: IAT 2005. Proceedings of the IEEE Conference on Intelligent Agent Technolology (2005)

3. Doherty, P., Haslum, P., Heintz, F., Merz, T., Persson, T., Wingman, B.: A Distributed Architecture for Autonomous Unmanned Aerial Vehicle Experimentation. In: Proceedings of the International Symposium on Distributed Autonomous Robotic Systems, pp. 221–230 (2004)
4. Doherty, P., Kvarnström, J.: TALplanner: A temporal logic based planner. Artificial Intelligence Magazine (Fall Issue, 2001)
5. Doherty, P., Kvarnström, J.: Temporal action logics. In: Lifschitz, V., van Harmelen, F., Porter, F. (eds.) The Handbook of Knowledge Representation, vol. ch.18, Elsevier, Amsterdam (to appear, 2007)
6. Heintz, F., Doherty, P.: Dyknow: An approach to middleware for knowledge processing. Journal of Intelligent and Fuzzy Systems 15(1), 3–13 (2004)
7. Heintz, F., Doherty, P.: Dyknow: A knowledge processing middleware framework and its relation to the JDL fusion model. Journal of Intelligent and Fuzzy Systems 17(4) (2006)
8. Kruppa, H., Castrillon-Santana, M., Schiele, B.: Fast and robust face finding via local context. In: Joint IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance (October 2003)
9. Lienhart, R., Maydt, J.: An Extended Set of Haar-like Features for Rapid Object Detection. In: Proc. of International Conference on Image Processing, pp. 900–903 (2002)
10. OSCVL. Open Source Computer Vision Library, http://www.intel.com/technology/computing/opencv/
11. Viola, P., Jones, M.J.: Rapid Object Detection using a Boosted Cascade of Simple Features. In: Proc. of Conference on Computer Vision and Pattern Recognition (2001)
12. Wzorek, M., Conte, G., Rudol, P., Merz, T., Duranti, S., Doherty, P.: From Motion Planning to Control - A Navigation Framework for an Autonomous Unmanned Aerial Vehicle. In: Proc. of the 21th Bristol International UAV Systems Conference (2006)
13. Wzorek, M., Doherty, P.: A framework for reconfigurable path planning for autonomous unmanned aerial vehicles. Journal of Applied Artificial Intelligence (2007)

Social Agents

Norman Foo

The School of Computer Science and Engineering
University of New South Wales, Sydney NSW 2052, Australia
and
National ICT Australia
Neville Roach Labs, Kensington
UNSW Sydney NSW 2052, Australia

Abstract. Social agents are computational cognitive agents that have epistemic abilities besides the traditional ones of action, communication and other forms of interaction. There is increasing interest in the modelling of such agents. This talk will first outline selected aspects of social agents and potential applications. Then we proceed to an example of how a couple of aspects may be investigated, viz., agent trust based on experience.

Extended Summary

Traditionally agents in artificial intelligence are imbued with abilities to act on environments, interacting with other agents from time to time. Multiagent systems are configured within such a framework. These agents use reasoning to enhance reactive scripts triggered by sensing. The reasoning is typically confined that about distributing tasks, failure planning, etc. Beliefs are often part of an agent's attributes, and an essential component of any epistemic ability. The nature of these beliefs determines the cognitive power of the agent. In certain applications it may be necessary for an agent A to maintain beliefs not only about another agent B but also about B's beliefs about A, and B's beliefs about A's beliefs about B, etc. Social agents are those that ideally have such complex beliefs. However, as infinite nesting is not feasible there is the question about how any working system (as in the "real world") can get away with less. Some suggestions will be reviewed.

In social agent interactions within a stable society many conventions are taken for granted. These are the social norms. For social norms to prevail agents must trust one another. The second part of the talk will outline recent work [Foo and Renz 07-1] on how trust can be modelled as "condensed experience", revealing the kind of mathematical structures needed and some logic questions that naturally arise.

Reference

[Foo and Renz 07-1] Foo, N., Renz, J.: Experience and Trust: A Systems-Theoretic Approach, Technical Report UNSW-CSE-TR-0717, Computer Science and Engineering, University of New South Wales (August 2007)

Globally Optimal Solutions to Vision Using Convex and Quasi-convex Optimization

Richard Hartley

Department of Systems Engineering
RSISE, Building 135
Australian National University
ACT 0200, Australia
Richard.Hartley@anu.edu.au

Abstract. Vision Geometry is the are of Computer Vision that deals with computing the geometry of the 3D world from sequences of images. It grew out of Photogrammetry, a field that goes back at least to the start of the 20th century. In the 1990s this field was transformed by the application of methods of Projective Geometry, leading to many new algorithms and deployment of the new methods in a wide variety of applications.

The algorithmic basis for Vision Geometry still ultimately relied on a technique called "bundle adjustment", involving iterative refinement of initial solutions by Newton or Gauss-Newton methods. These had the disadvantage of often finding local rather than global minima.

Recent work has focussed on applying different optimization techniques, particularly Convex Optimization techniques to attempt to find guaranteed global solutions to these problems. I will talk about progress in this area, through the use of methods such as Second Order Cone Programming, branch-and-bound fractional programming and semi-definite programming.

Efficient Learning of Bayesian Network Classifiers

An Extension to the TAN Classifier

Alexandra M. Carvalho[1], Arlindo L. Oliveira[1], and Marie-France Sagot[2]

[1] IST, TULisbon/INESC-ID, Lisboa, Portugal
{asmc,aml}@inesc-id.pt
[2] Inria Rhône-Alpes, Université Claude Bernard, Lyon I, France
Marie-France.Sagot@inria.fr

Abstract. We introduce a Bayesian network classifier less restrictive than Naive Bayes (NB) and Tree Augmented Naive Bayes (TAN) classifiers. Considering that learning an unrestricted network is unfeasible the proposed classifier is confined to be consistent with the breadth-first search order of an optimal TAN. We propose an efficient algorithm to learn such classifiers for any score that decompose over the network structure, including the well known scores based on information theory and Bayesian scoring functions. We show that the induced classifier always scores better than or the same as the NB and TAN classifiers. Experiments on modeling transcription factor binding sites show that, in many cases, the improved scores translate into increased classification accuracy.

1 Introduction

Learning Bayesian networks has been a hot and fruitful research topic [23, 15, 3, 17]. The goal of learning a Bayesian network is to find both the structure and the parameters of the network that best fit the data, according to a given scoring function. The inference of a general Bayesian network has been shown to be an NP-hard problem [5], even for approximate solutions [7]. As a consequence, heuristic algorithms became the standard methodology for addressing this problem [23]. A common approach is to impose restrictions over the network structure. In this context, two results set the border between efficient and non-efficient structure learning. In one hand, Chow and Liu showed that trees can be learned in polynomial time [4], on the other hand, Dasgupta proved that learning 2-polytrees is NP-hard [8].

Bayesian networks have been widely used in the context of classification [21, 16, 14]. The simplicity and high accuracy of the *Naive Bayes* (NB) classifier [11] have led to its extensive use, and to several attempts to extends it. In this line of research Friedman et al [14] proposed the *Tree Augmented Naive Bayes* (TAN) classifier in order to overcome the strong independence assumptions imposed by the NB network. The TAN is an extension of NB which allows additional edges between the attributes of the network in order to capture correlations among them. Such correlations are however restricted to a tree structure. Friedman et al showed that TAN was indeed more accurate than NB on benchmark datasets.

Herein, we introduce a Bayesian network classifier less restrictive than NB and TAN classifiers. Considering that learning an unrestricted network is unfeasible, the underlying graph of the proposed classifier is confined to be consistent with the *breadth-first search* (BFS) order of an optimal TAN and to have a bounded in-degree, henceforward called *BFS-consistent k-graph* (BCkG). We show that learning BCkG's can be done efficiently and for any scoring functions that decomposes over the network structure. Well known scores with this property are those based on information theory, such as *log likelihood* (LL), *Akaike information criterion* (AIC), *Bayesian information criterion* (BIC), *minimum description length* (MDL); and Bayesian scoring function such as K2, *Bayesian Dirichlet* (BD) and its variants (BDe, BDeu), and *mutual information test* (MIT). We show that the classifiers induced from BCkG's score always better than or the same as the NB and TAN classifiers, and moreover that the search space of the learning algorithm is exponentially larger than the TAN learning algorithm. We check the quality of our approach with biological data. Experiments show that, in many cases, the improved scores translate into increased classification accuracy.

The paper is organized as follows. In Section 2, we briefly revise Bayesian networks, Bayesian network classifiers and their learning algorithms. In Section 3, we introduce the main contribution of this paper, the learning algorithm for BCkG classifiers. In Section 4, we apply our approach in the realm of computational biology, namely to model transcription factor binding sites, and present some experimental results. Finally, in Section 5 we draw some conclusions and discuss future work.

2 Basic Concepts and Results

In this section we introduce some notation, while recalling relevant concepts and results concerning Bayesian networks which are directly related with the contribution of this paper.

2.1 Bayesian Networks

A *Bayesian network* is a triple $B = (\mathbf{X}, G, \Theta)$. The first component $\mathbf{X} = (X_1, \ldots, X_n)$ is a finite random vector where each random variable X_i ranges over a finite domain D_i. We denote the joint domain $\mathbf{D} = \Pi_{i=1}^{n} D_i$. The second component $G = (N, E)$ is a directed acyclic graph with nodes $N = \{X_1, \ldots, X_n\}$ and edges E representing direct dependencies between the variables. The third component Θ encodes the parameters $\{\theta_{x_i|\Pi_{x_i}}\}_{\mathbf{x}\in\mathbf{D}}$ of the network, where $\theta_{x_i|\Pi_{x_i}} = P_B(x_i|\Pi_{x_i})$ for each possible value x_i of X_i, and Π_{x_i} of Π_{X_i}, where Π_{X_i} denotes the set of parents of X_i in G. A Bayesian network defines a unique joint probability distribution over $\mathbf{X}$ given by

$$P_B(X_1, \ldots, X_n) = \prod_{i=1}^{n} \theta_{X_i|\Pi_{X_i}}. \tag{1}$$

We denote the set of all Bayesian networks with n variables by $\mathcal{B}_n$.

Informally, a Bayesian network encodes the independence assumptions over the component random variables of $\mathbf{X}$. An edge (i, j) in E represents a direct dependency of X_j to X_i. Moreover X_i is independent of its non descendants given its parents Π_{X_i} in G.

The problem of learning a Bayesian network given data T consists on finding the Bayesian network that best fits the data T. In order to quantify the fitting of a Bayesian network a *scoring function* $\phi : \mathcal{B}_n \times \mathbf{D}^m \to \mathbb{R}$ is considered. In this context, the problem of learning a Bayesian network can be recasted to the following optimization problem. Given a dataset $T = \{\mathbf{x}_1, \ldots, \mathbf{x}_m\}$ and a scoring function ϕ, the *problem of learning a Bayesian network* is to find a Bayesian network $B \in \mathcal{B}_n$ that maximizes the value ϕ for T.

Several scoring functions have been proposed in the literature [6,17,18,22]. The discussion of the advantages and disadvantages of each of these functions is outside the scope of this paper.

2.2 Bayesian Network Classifiers

A *Bayesian network classifier* is a Bayesian network where $\mathbf{X} = (X_1, \ldots, X_n, C)$. The variables $X_1, \ldots, X_n$ are called *attributes* and C is called the *class variable.* Moreover, the graph structure G is such that the class variable has no parents, that is, $\Pi_C = \emptyset$, and all attributes have at least the class variable as parent, that is, $C \in \Pi_{X_i}$. The corresponding classifier is defined as

$$\arg\max_C P_B(C|X_1, \ldots, X_n).$$

We therefore reformulate the model to make it more tractable. Using the definition of conditional probability and Equation (1) leads to the following classifier:

$$\arg\max_C P_B(C) \prod_{i=1}^{n} \theta_{X_i|\Pi_{X_i}}.$$

Informally, the problem of learning a Bayesian network classifier can be recasted as the problem of learning a Bayesian network where all attributes have the class variable as parent.

Naive Bayesian Network Classifier. A *naive Bayesian network* (NB) [11] is a Bayesian network classifier where each attribute has the class variable as its unique parent, that is, $\Pi_{X_i} = \{C\}$ for all $1 \leq i \leq n$. Since the NB has a fixed graph structure, learning the network reduces to computing the empirical distribution.

The NB classifier is one of the most effective classifiers, in the sense that, in many cases, its predictive performance is competitive with state-of-the-art classifiers [10,9]. In fact, the NB classifier is computationally undemanding and shows an unexpected accuracy in many applications. However, the independence assumption is too strict and relaxing this assumption may lead to more accurate classification.

Tree Augmented Naive Bayesian Network Classifier. A *tree augmented naive Bayesian network* (TAN) [14] is a Bayesian network classifier where there exists an $r \in \{1, \ldots, n\}$ such that $\Pi_{X_r} = \{C\}$ and $\Pi_{X_i} = \{C, X_j\}$ for all $1 \leq i \leq n$ with $i \neq r$. The TAN was first proposed by Friedman et al [14] to overcome the strong independence assumptions imposed by the NB network. In fact, the TAN is an extension of NB which allows additional edges between the attributes of the network in order to capture correlations among them. Such correlations are however restricted to a tree structure.

In [14] an algorithm to find an optimal TAN that maximizes the LL is given. The main idea is to consider a complete weighted undirected graph, where each edge between X_i and X_j is weighted with the conditional mutual information between X_i and X_j given the class variable C. Given this, the problem reduces to determining a maximal weighted spanning tree, using the algorithm by Chow and Liu [4]. After computing such spanning tree, a direction has to be assigned to each edge of the tree. This is done by choosing an arbitrary attribute as the tree root and then setting the direction of all edges to be outward from it.

Extending TAN Classifier to Deal with Decomposable Scores. The TAN was originally proposed for maximizing the LL score but it can be easily adapted to deal with any scoring function that is *decomposable* and *score equivalent*. We recall that a scoring function ϕ is *decomposable* if it can be written as

$$\phi(B, T) = \sum_{i=1}^{n} \phi_i(\Pi_{X_i}, T). \tag{2}$$

Moreover, a scoring function is said to be *score equivalent* if it assigns the same value to all directed acyclic graphs that are represented by the same essential graph. All interesting scoring functions in the literature are decomposable, since it is unfeasible to learn undecomposable scores. LL, AIC, BIC and MDL are decomposable and score equivalent, whereas K2, BD, BDe, BDeu and MIT are decomposable but not score equivalent.

According to Heckerman et al [17], finding an optimal TAN classifier for decomposable and score equivalent scoring functions reduces to weighting each undirected edge between X_i and X_j by $\phi_j(\{X_i, C\}, T) - \phi_j(\{C\}, T)$, which is equal to $\phi_i(\{X_j, C\}, T) - \phi_i(\{C\}, T)$ by score equivalence of ϕ, and to find a maximal weighted (undirected) spanning tree. Moreover, learning an optimal TAN classifier for scoring functions that are only decomposable, but not score equivalent, can also be done in polynomial time. In this case, however, an edge between X_i and X_j may score differently depending on its direction, and so a directed spanning tree must be found (instead of an undirected one). The idea is to weight each directed edge from X_i and X_j with $\phi_j(\{X_i, C\}, T) - \phi_j(\{C\}, T)$ and then, for each node X_r, find an optimal directed spanning tree rooted at X_r with Edmonds' algorithm [12]

3 BFS-Consistent Bayesian Network Classifiers

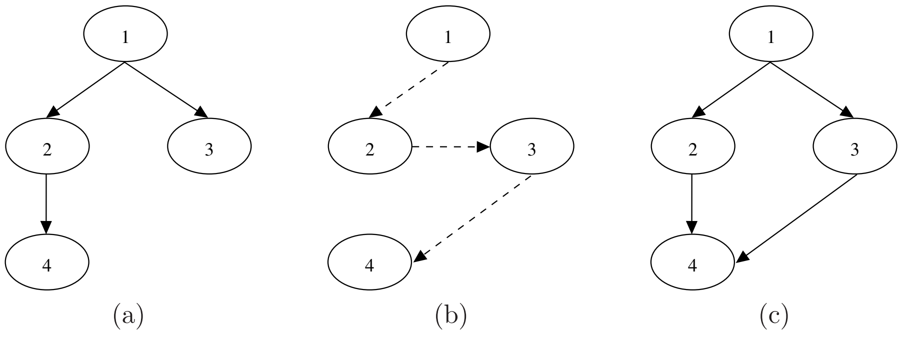
We now introduce the main contribution of this paper, a simple and effective heuristic for a causality order between the attributes based on a *breadth-first search* (BFS) over an optimal TAN. The main idea is to take the total order induced by the BFS over an optimal TAN and then search for an optimal network (of bounded in-degree) consistent with it. It is easy to show that the score of the resulting network is always greater than or equal to the score of TAN and NB. The foremost benefit of this approach is that learning such an optimal network can be done efficiently, that is, in polynomial time over the number of attributes. Moreover, the class of the networks consistent with the BFS order is exponentially larger, in the number of variables, when compared to TAN networks.

We start by introducing some auxiliary concepts. A *k-graph* is a graph where each node has in-degree at most k. Trees and forests are 1-graphs.

Definition 1 (BFS-consistent k-graph). Given a TAN R with a set of attributes N, a graph $G = (N, E)$ is said to be a *BFS-consistent k-graph* (BCkG) w.r.t R if it is a k-graph and for any edge in E from X_i to X_j the node X_i is visited in *breadth-first search* (BFS) of R before X_j. Henceforward, we denote by $\mathcal{B}_R^k$ the set of all BCkG's w.r.t. R.

The above definition of consistency imposes that there can only exist an edge from X_i to X_j in $G \in \mathcal{B}_R^k$ if X_i is less than or as deep as X_j in R. We assume that if $i < j$ and X_i and X_j are at the same level, then the BFS over R reaches X_i before X_j. Other approaches to order attributes at the same level are discussed in the conclusions.

Example 1. Given the underlying graph for the attributes of a TAN R in (a), its BFS is represented by a dashed line in (b). A BC2G *w.r.t R is presented in (c).*



The core idea of the BCkG learning algorithm is to compute an optimal TAN R and improve it by adding/removing dependencies which were omitted/present because of the TAN structure restrictions. For efficiency purposes, the modified model must be a BFS-consistent k-graph w.r.t. R. In this context, the total order induced by the BFS over R might add dependencies from higher nodes to deeper nodes. In detail, the algorithm starts by computing an optimal TAN as described in Section 2.2. Then it performs a BFS over the optimal TAN to construct a total order. Finally, it ranges over each attribute X_i, generates the set α_i of all attributes less than X_i, and takes as parents of X_i the set $S \subseteq \alpha_i$ such that $\phi_i(S \cup \{C\}, T)$ is maximal over all subsets of α_i with at most k attributes. The pseudocode of the algorithm is presented in Algorithm 1.

Algorithm 1. Learning BCkG network classifiers

```
1. Run a (deterministic) algorithm that outputs an optimal TAN $R$ according to $\phi$.
2. Compute the total order $\sqsubseteq$ induced by the BFS over $R$ (ignoring the class variable).
3. For each attribute $X_i$ in $R$ do:
   (a) Compute the set $\alpha_i = \{X_j \in R : X_i \sqsubseteq X_j$ and $X_i \neq X_j\}$.
   (b) For each subset $S$ of $\alpha_i$ with at most $k$ attributes:
        i. Compute $\phi_i(S \cup \{C\}, T)$.
       ii. If $\phi_i(S \cup \{C\}, T)$ is the maximal score for $X_i$, set $\Pi_i = S \cup \{C\}$.
4. Output the directed graph $G$ such that the parents of an attribute $X_i$ are $\Pi_i$.
```

Theorem 1 (Soundness). Algorithm 1 constructs a BCkG Bayesian network classifier that maximizes the ϕ-score given data T.

Proof. Since all potential parents for each node are checked, the algorithm returns the k-graph G BSF-consistent w.r.t R with the highest score. Moreover, this graph is acyclic since the parents of a node X_i must be in α_i, that is, must belong to the path in R from its root to X_i (excluding X_i). Moreover, it is easy to see that for any path $X_{i_1}, X_{i_2}, \ldots X_{i_k}$ in G we have that $X_{i_j} \in \alpha_{i_k}$ for $1 \leq j < k$. If there existed a cycle $X_{i_1}, X_{i_2}, \ldots X_{i_1}$ it would imply that $X_{i_1} \in \alpha_{i_1}$ which is absurd. □

Proposition 1. Algorithm 1 constructs a BCkG Bayesian network classifier whose ϕ-score is always greater than, or equal to, the ϕ-score of the optimal TAN.

Proof. Start by noticing that the soundness of Algorithm 1 assures that the resulting BCkG w.r.t R is the maximal among all BCkG's in $\mathcal{B}_R^k$. Moreover, observe that the underlying graph G_R of the TAN R (without the class variable) is BFS-consistent w.r.t R, that is, $G_R \in \mathcal{B}_R^k$ for all $k \geq 1$. Hence, the soundness of Algorithm 1 guarantees that the BN classifier B_G, constructed by adding an edge from the class variable C to all attributes in the output k-graph G, is such that $\phi(B_G, T) \geq \phi(R, T)$. □

Theorem 2 (Complexity). Algorithm 1 constructs a BCkG Bayesian network classifier in $\mathrm{O}(n^{k+1}\gamma(k, T))$ time where $\gamma(k, T)$ is an upper bound for computing $\phi_i(S \cup \{C\}, T)$.

Proof. Step 2 takes $\mathrm{O}(n)$ time. Step 3a) takes $\mathrm{O}(n)$ time, while step 3b) takes $\mathrm{O}(n^k\gamma(S, T))$ time because it ranges over all subsets S with at most k elements (which takes $\mathrm{O}(n^k)$ time) and for each of this sets it computes $\phi_i(S \cup \{C\}, T)$ (which takes $\mathrm{O}(\gamma(S,T))$ time. Thus, the overall complexity of the algorithm is $\mathrm{O}(n^{k+1}\gamma(S,T))$ time. $\square$

The theorems above assert the soundness and polynomial-time bound of the BCkG learning algorithm. At this point it remains to show that, despite considering an optimal TAN to confine the search space, the number of graphs searched increases exponentially, in the number of attributes, when compared to TAN's.

Proposition 2. Let R be a TAN with n attributes, then the number of non-trees in $\mathcal{B}_R^k$ is at least $2^{nk-\frac{k^2}{2}-\frac{k}{2}-1}$ when $n \geq k$.

Proof. We denote by $(N, \sqsubseteq)$ the total order induced by BFS over R (ignoring the class variable). Since, this order is total (also called a linear), for any pair of nodes X_i and X_j in N, with $i \neq j$, we can say that a node X_i is *lower than* X_j if and only if $X_i \sqsubseteq X_j$. Given this, notice that the i-th node of R has precisely $(i-1)$ lower nodes. We conclude that, when $i > k(\leq n)$, there are at least 2^k subsets of N with at most k lower nodes. Moreover, when $(1 \leq)i \leq k$, only 2^{i-1} subsets of N with at most k lower nodes exist. Thus,

$$|\mathcal{B}_R^k| \geq \left(\prod_{i=k+1}^{n} 2^k\right) \times \left(\prod_{i=1}^{k} 2^{i-1}\right) = 2^{nk-\frac{k^2}{2}-\frac{k}{2}}$$

give us a lower bound for the total number of possible BCkG w.r.t R (recall that a BC1G is also a BC2G, both a BC1G and a BC2G are also a BC3G, and so on). Now, consider that X_i is the root, and X_j is the child of the root in R. The only two subsets of N with at most k lower elements than X_j are $\emptyset$ and $\{X_i\}$. This choice splits in two all BCkG's in $\mathcal{B}_R^k$. Those for which the set of parents of X_j is $\emptyset$ cannot be trees since X_i has no parents as well. Therefore, there are at lest $\frac{|\mathcal{B}_R^k|}{2} \geq 2^{nk-\frac{k^2}{2}-\frac{k}{2}-1}$ in $\mathcal{B}_R^k$ that are non trees. $\square$

4 Experimental Methodology and Results

We compared the BC2G classifier with TAN and NB classifiers. We only considered discrete attributes and removed instances with missing values from the datasets. As suggested by Friedman et al [14] we improved the performance of both TAN and BCkG classifiers by introducing an additional smoothing operation. This is particularly important in small datasets where the estimation of the conditional probabilities, given the parent attributes plus the class variable, is unreliable. NB classifiers are almost not affected by this problem since the data is partitioned according to the class variable and, usually, the class variables are adequately represented in the training set. The parameters of TAN and BCkG networks were smoothed using *Dirichlet priors* [17]. This amounts to adding 5 pseudo instances with conditional probabilities, given the parent attributes plus the class variable, distributed according to the frequency of the corresponding attribute in the training set.

Modeling Transcription Factor Binding Sites

We wanted to evaluate our method in the context of computational biology. There is a straightforward application of the BCkG model in the representation of transcription factor binding sites.

An important part of gene regulation is mediated by specific proteins, called the *transcription factors*, which influence the transcription of a particular gene by binding to specific sites on DNA sequences, called *transcription factor binding sites* (TFBS). Such binding sites are relatively short stretches of DNA, normally 5 to 25 nucleotides long. A commonly used representation of TFBS is a *position specific scoring matrices* (PSSM). This representation assumes independence of nucleotides in the binding sites, and so can be modeled by a Naive Bayes network. Some works appeared that argued in the direction of non-additivity in protein-DNA interactions [19] making a way for more complex models to appear which account for nucleotide interactions. Barash et al had already obtained good results modeling TFBS with trees and mixtures of trees [1]. Recently, Sharon and Segal also contributed in this direction [20]. Herein, we do preliminary evaluation of the extent to which the richer BC2G models are beneficial in representing TFBS.

The TRANSFAC database [13] contains hundreds of biologically validated TFBS. We extracted 14 data sets of aligned binding sites from the TRANSFAC database for which there were 20 or more sites. For each binding site we evaluated the ability of NB, TAN and BC2G to describe the distribution underlying the TFBS. We performed a *10 fold cross-validation* test in each data set and the results of the evaluation are presented in Figure 1.

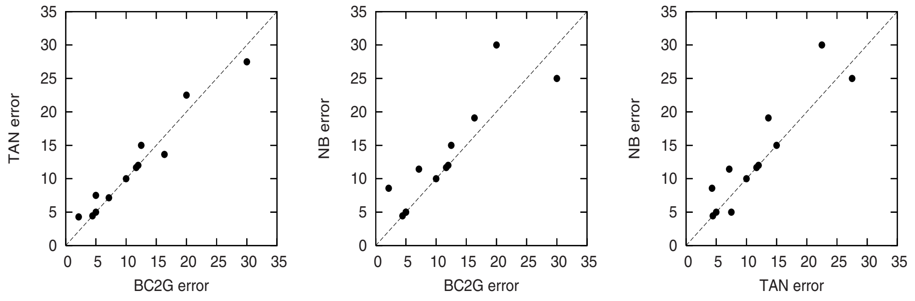


Fig. 1. Scatter plot comparing NB, TAN, and BC2G networks when modeling transcription factor binding sites. Points above the diagonal line corresponds to data sets on which the model in the x axis performs better than the model in the y axis.

5 Conclusions

This paper introduced a new heuristic to learn Bayesian network classifiers. The proposed heuristic consists of improving an optimal TAN classifier by adding important dependencies and removing irrelevant ones, guiding this process with the total order induced by the BFS over the optimal TAN. The advantage of this restriction is twofold. First, the learning algorithm is polynomial. Second, as Friedmann et al observed [14], unrestricted learning does not necessarily outperforms TAN and NB due to overfitting,[1] and for this reason structure restriction helps avoiding this problem. The proposed heuristic is an improvement over the partial order based heuristic introduced in [2], adapted for classification. The proposed classifier scores always better than both TAN and NB classifiers. Moreover, experiments on modeling transcription factor binding sites show that, in many cases, the improved scores translate into increased classification accuracy.

Direction of future work include: instead of fixing an order for attributes at the same level in the BFS (c.f. Section 3 comment after Definition 1), (i) consider a random order over attributes at the same level or (ii) apply the TAN algorithm solely to attributes at the same level and order them with a BFS over the resulting TAN; combine and compare exhaustively our approach with other state-of-the-art Bayesian network learning methods; extending BCkG to deal with missing values and non discretized continuous variables; applying BCkG to a wider variety of datasets.

Acknowledgments

The first author thanks EU FEDER through FCT grant SFRH/BD/18660/2004. This work was partially supported by EU FEDER via FCT project POSC/EIA/57398/2004.

References

1. Barash, Y., Elidan, G., Friedman, N., Kaplan, T.: Modeling dependencies in protein-DNA binding sites. In: Proc. RECOMB 2003, pp. 28–37 (2003)
2. Carvalho, A.M., Oliveira, A.L., Sagot, M.-F.: Learning Bayesian networks consistent with the optimal branching. In: Proc. ICMLA 2007 (to appear, 2007)
3. Chickering, D.M.: Learning Bayesian networks is NP-Complete. In: Learning from data: AI and statistics V, pp. 121–130. Springer, Heidelberg (1996)
4. Chow, C.K., Liu, C.N.: Approximating discrete probability distributions with dependence trees. IEEE Trans. Info. Theory 14(3), 462–467 (1968)
5. Cooper, G.F.: The computational complexity of probabilistic inference using Bayesian belief networks. Artif. Intell. 42(2-3), 393–405 (1990)
6. Cooper, G.F., Herskovits, E.: A Bayesian method for the induction of probabilistic networks from data. Machine Learning 9, 309–347 (1992)
7. Dagum, P., Luby, M.: Approximating probabilistic inference in Bayesian belief networks is NP-hard. Artif. Intell. 60(1), 141–153 (1993)
8. Dasgupta, S.: Learning polytrees. In: Proc. UAI 1999, pp. 134–141 (1999)

[1] Friedmann et al also noticed that the learning algorithm for Bayesian classifiers should maximize the conditional likelihood scoring function (CLL) instead of the likelihood. However, CLL is not decomposable, and therefore learning it seems to be intractable.

9. Domingos, P., Pazzani, M.J.: Beyond independence: Conditions for the optimality of the simple Bayesian classifier. In: Proc. ICML1996, pp. 105–112 (1996)
10. Domingos, P., Pazzani, M.J.: Simple Bayesian classifiers do not assume independence. In: Proc. AAAI/IAAI 1996, vol. 2, p. 1386 (1996)
11. Duda, R.O., Hart, P.E.: Pattern Classification and Scene Analysis. John Wiley and Sons, New York (1973)
12. Edmonds, J.: Optimum branchings. J. Research of the National Bureau of Standards 71B, 233–240 (1967)
13. Wingender, E., et al.: The TRANSFAC system on gene expression regulation. Nuc. Ac. Res. 29(1), 281–283 (2001)
14. Friedman, N., Geiger, D., Goldszmidt, M.: Bayesian network classifiers. Machine Learning 29(2-3), 131–163 (1997)
15. Friedman, N., Koller, D.: Being Bayesian about network structure: A Bayesian approach to structure discovery in Bayesian networks. Machine Learning 50(1-2), 95–125 (2003)
16. Grossman, D., Domingos, P.: Learning Bayesian network classifiers by maximizing conditional likelihood. In: Proc. ICML 2004, pp. 46–53. ACM Press, New York (2004)
17. Heckerman, D., Geiger, D., Chickering, D.M.: Learning Bayesian networks: The combination of knowledge and statistical data. Machine Learning 20(3), 197–243 (1995)
18. Lam, W., Bacchus, F.: Learning Bayesian belief networks: An approach based on the MDL principle. Comp. Intell. 10, 269–294 (1994)
19. O'Flanagan, R.A., Paillard, G., Lavery, R., Sengupta, A.M.: Non-additivity in protein-DNA binding. Bioinformatics 21(10), 2254–2263 (2005)
20. Sharon, E., Segal, E.: A feature-based approach to modeling protein-DNA interactions. In: Proc. RECOMB 2007, LNCS(LNBI), vol. 4453, pp. 77–91. Springer, Heidelberg (2007)
21. Su, J., Zhang, H.: Full Bayesian network classifiers. In: Proc. ICML 2006, pp. 897–904. ACM Press, New York (2006)
22. Suzuki, J.: A construction of Bayesian networks from databases based on an MDL principle. In: Proc. UAI 1993, pp. 266–273 (1993)
23. Teyssier, M., Koller, D.: Ordering-based search: A simple and effective algorithm for learning Bayesian networks. In: Proc. UAI 2005, pp. 584–591 (2005)

Mixture of the Robust L1 Distributions and Its Applications

Junbin Gao and Richard Y. Xu

School of Acc & Computer Science
Charles Sturt University, Bathurst, NSW 2795, Australia
{jbgao,rxu}@csu.edu.au

Abstract. Recently a robust probabilistic L1-PCA model was introduced in [1] by replacing the conventional Gaussian noise model with the Laplacian L1 model. Due to the heavy tail characteristics of the L1 distribution, the proposed model is more robust against data outliers. In this paper, we generalized the L1-PCA into a mixture of L1-distributions so that the model can be used for possible multiclustering data. For the model learning we use the property that the L1 density can be expanded as a superposition of infinite number of Gaussian densities to include a tractable Bayesian learning and inference based on the variational EM-type algorithm.

1 Introduction

In many applications, multivariate observations have frequently been represented using a mixture of Gaussian models [2]. These are convenient in many respects, they have a closed-form probability density function and the parameter can easily be obtained using the EM algorithm [3]. The Gaussian mixture model is a generative model and it is useful to consider the process of generating samples from the density it represents.

To extract intrinsic structures in a data set, the mixture of PPCA (probabilistic PCA) [4] was proposed to enable one to model non-linear relationships by aligning a collection of such local models. J. Verbeek's work [5] on variational model for the mixture of PPCA provided inner sight to the global alignment of local linear models. However, PPCA, FA (Factor Analyzer) and their mixture counterparts have severe limitations in practice. They all are based on Gaussian density assumptions, particularly for the likelihood or noise process. It is clear that not all real-world data sets can be modelled well by Gaussian distributions. In particular, constraining a model to Gaussian assumption whose tails rapidly decay is a well-characterized disadvantage when the true underlying distribution is heavy-tailed (has high kurtosis) or when there exist outliers in the data set. In such cases, it is obviously more appropriate to assume a heavy-tailed distribution within the analysis, but this assumption raises significant difficulties in tractability.

Generally speaking a general mixture distribution comprises a finite or even infinite number of components, possibly of different distributional types, that can describe different features of data. There are lots of attempts to use non-Gaussian components in a mixture model to facilitate much more careful description of complex systems, as evidenced by the enthusiasm with which they have been adopted in such diverse areas as astronomy, ecology, bioinformatics, computer science, ecology, economics, engineering, robotics and biostatistics. For example, in bioinformatics, a two components mixture model [6] was introduced, one of which is the Laplace distribution as a long-tailed alternative to the Gaussian distribution, to identify differentially expressed genes in microarray experiments. And, in computer science, spam filters and web context analysis [7] start from a mixture assumption to distinguish spams from regular emails and group pages by topic, respectively.

Among non-Gaussian components, the Student-t component is a very interesting one. It is one of heavy-tailed distributions. In fact, the Student-t distribution is a heavy-tailed generalization of the Gaussian distribution. Compared with Gaussian distributions, using Student-t distributions for the model significantly increases the robustness of the model. Such kind of work has already been done by a number of researchers under different assumptions and different algorithm implementations. Under the assumption of the Student-t distribution, the related research includes mixture of Student-t models [8], and more recent work such as robust subspace mixture model [9], in which both the likelihood and the latent variables were supposed to be the Student-t distribution and the EM algorithm was applied to the model. Most recently [10] discussed the robust models in the context of finite mixture models.

Compared to the Gaussian distribution, the so-called centered Laplacian distribution (or L1 distribution or the least absolute deviance) is much less sensitive to outliers. The approach of using the L1 distribution originates from LASSO [11], and has caught some interest in machine learning [12] and statistics. Besides the robustness against outliers context, the L1 distribution assumption is also used as a penalty/regularization term on model parameters to enforce sparsity, or parameter/feature selection, such as sparse PCA [13,14], and logistic regression [12]. To the author's best knowledge the first work towards building a L1 mixture model was proposed by [15] to capture nonlinear dependencies in images of natural scene in an unsupervised manner.

This paper is concerned with the generative modelling for the L1 mixture model by using Bayesian learning and inference approaches. In our approach, we expand the Laplacian distribution as a superposition of infinite number of Gaussian distributions so that the underlying generative modelling can be handled by variational EM-type algorithms. The same approach was firstly used in solving the support vector regression problem, see [16]. The approach is also closely related to the work [17] regarding the Student-t distribution models. In the next section, we introduce the probabilistic L1 mixture model. Then, in Section 3, we show how the proposed L1 mixture model can be solved by the variational Bayesian technique and derive the variational EM-type algorithm. In Section 4, we present the experimental results to evaluate the presented methods. Finally, in Section 5, we present our conclusions.

2 L1 Mixture Model

Let $Y = \{\mathbf{y}_i : i = 1, 2, \ldots, N\}$ be N independently identical random variables with values in $\mathbb{R}^d$. The model we consider assumes that each $\mathbf{y}_i=(y_{i1}, y_{i2}, ..., y_{id})^\top$ is generated independently from a mixture model given by

$$p(\mathbf{y}_i|\{\pi_j, \boldsymbol{\mu}_j, \boldsymbol{\sigma}_j)\}) = \sum_{j=1}^{m} \pi_j p_{L_1}(\mathbf{y}_i|\boldsymbol{\mu}_j, \boldsymbol{\sigma}_j), \text{with } \sum_{j=1}^{m} \pi_j = 1 \tag{1}$$

where the mixing weights denoted by π_j are nonnegative and add up to one, and $\boldsymbol{\mu}_j = (\mu_{j1}, ..., \mu_{jd})$ is the parameter for the mean of the j-th mixture component and $\boldsymbol{\sigma}_j = \text{diag}(\sigma_{j1}, ..., \sigma_{jd})$ the covariance. Each component p_{L_1} is defined by the so-called L1 Laplacian distribution

$$p_{L_1}(\mathbf{y}_i|\boldsymbol{\mu}_j, \boldsymbol{\sigma}_j) = \prod_{k=1}^{d} \frac{1}{2\sigma_{jk}} \exp\left\{-\frac{1}{\sigma_{jk}}|y_{ik} - \mu_{jk}|\right\} \tag{2}$$

We collect all the hyperparameters in the above model in $\Theta = \{\pi_1, ..., \pi_m; \boldsymbol{\mu}_1, ..., \boldsymbol{\mu}_m\}$. The number of parameters depends on the number of components m and the number of data dimension d. For the variance $\boldsymbol{\sigma}_j$ in (1), we specify a Gamma priori on $\rho_{jk} = 1/\sigma_{jk}^2$ given by

$$p(\rho_{jk}|a_{jk}, b_{jk}) = \frac{b_{jk}^{a_{jk}}}{\Gamma(a_{jk})} \rho_{jk}^{a_{jk}-1} \exp\{-b_{jk}\rho_{jk}\} \tag{3}$$

For the given dataset $Y = \{\mathbf{y}_i : i = 1, 2, \ldots, N\}$, the maximum likelihood (ML) estimate of the parameter values is

$$\widehat{\Theta} = \text{argmax}_{\Theta} \log \prod_{i=1}^{N} p(\mathbf{y}_i|\Theta)$$

To handle with the above ML problem for the mixture model (1), a usual way is to introduce a hidden indicator variable $\mathbf{z}_i = (z_{i1}, z_{i2}, ..., z_{im})$ for each data $\mathbf{y}_i$ such that if $z_{ij} = 1$, we say the j-th component generates the i-th data $\mathbf{y}_i$, where z_{ij}s are 0-1 variables and $\sum_j z_{ij} = 1$ and $p(z_{ij} = 1) = \pi_j$.

Then based on (1) the joint distribution of the data and the indicator variables can be represented as

$$p(Y, Z|\Theta) = \prod_{i=1}^{N} \prod_{j=1}^{m} [\pi_j p_{L_1}(\mathbf{y}_i|\boldsymbol{\mu}_j, \boldsymbol{\sigma}_j)]^{z_{ij}} \tag{4}$$

There are still several difficulties when applying the Bayesian variational optimization to the mixture of L1 model (4). In the model, the Laplacian likelihood makes it hard to find or define a good approximation to the posterior of Z. Fortunately the Laplacian likelihood (2) can be expanded as a superposition of infinite number of Gaussian distribution given by the following relation [18]

$$p_{L_1}(\mathbf{y}_i;\boldsymbol{\mu}_j,\boldsymbol{\sigma}_j)=\prod_{k=1}^{d}\int_0^{\infty}\mathcal{N}(\mathbf{y}_i|\boldsymbol{\mu}_j,\Sigma_{ij})p(\boldsymbol{\beta}_{ij})d\boldsymbol{\beta}_{ij}$$

where $\mathcal{N}(\mathbf{y}_i|\boldsymbol{\mu}_j,\Sigma_{ij})$ is a Gaussian with the mean $\boldsymbol{\mu}_j$ and covariance $\Sigma_{ij}=\mathrm{diag}(\sigma_{j1}^2/\beta_{ij1},...,\sigma_{jd}^2/\beta_{ijd})$ and

$$p(\boldsymbol{\beta}_{ij})=\prod_{k=1}^{d}\frac{1}{2}\beta_{ijk}^{-2}\exp\left\{-\frac{1}{2\beta_{ijk}}\right\} \tag{5}$$

Thus the joint distribution of the data set Y, the indicator Z, the precision $\boldsymbol{\beta}$ and $\boldsymbol{\rho}=1/\boldsymbol{\sigma^2}$ is given by

$$p(Y,Z,\boldsymbol{\beta},\boldsymbol{\rho}|\Theta)=\prod_{i=1}^{N}\prod_{j=1}^{m}\left[\pi_j\mathcal{N}(\mathbf{y}_i|\boldsymbol{\mu}_j,\Sigma_{ij})p(\boldsymbol{\beta}_{ij})\right]^{z_{ij}}p(\boldsymbol{\rho}_j) \tag{6}$$

where $p(\boldsymbol{\rho}_j)=\prod_{k=1}^{d}p(\rho_{jk})$ and $p(\rho_{jk})$ is a Gamma prior given by (3).

Unfortunately marginalization of all the latent variables is intractable, i.e., no analytical forms available for $p(Y|\Theta)$. If we wish to proceed, we need to turn to an approximate method. We are going to look at the variational Bayesian inference method.

3 Variational Approximation for L1 Mixture Model

In order to introduce the variational learning method, the following notation is used: Let $(Z,\boldsymbol{\beta},\boldsymbol{\rho})$ be the model latent variables, Θ is the hyperparameters in (6). For the given observation Y, the ML algorithm aims to maximize the log likelihood:

$$\mathcal{L}(\Theta)=\log p(Y|\Theta)=\log\sum_{Z}\int_{\boldsymbol{\beta},\boldsymbol{\rho}}p(Y,Z,\boldsymbol{\beta},\boldsymbol{\rho}|\Theta)d\boldsymbol{\beta}d\boldsymbol{\rho}$$

Using any distribution $Q(Z,\boldsymbol{\beta},\boldsymbol{\rho})$ over the latent variables, called variational distributions, we can obtain a lower bound on $\mathcal{L}$:

$$\begin{aligned}\mathcal{L}(\Theta)&=\log\sum_{Z}\int_{\boldsymbol{\beta}}\int_{\boldsymbol{\rho}}p(Y,Z,\boldsymbol{\beta},\boldsymbol{\rho}|\Theta)d\boldsymbol{\beta}d\boldsymbol{\rho}\\&\geq\sum_{Z}\int_{\boldsymbol{\beta}}\int_{\boldsymbol{\rho}}Q(Z,\boldsymbol{\beta},\boldsymbol{\rho})\log\frac{p(Y,Z,\boldsymbol{\beta},\boldsymbol{\rho}|\Theta)}{Q(Z,\boldsymbol{\beta},\boldsymbol{\rho})}d\boldsymbol{\beta}d\boldsymbol{\rho}\end{aligned} \tag{7}$$

Denote by $\mathcal{F}(Q(Z,\boldsymbol{\beta},\boldsymbol{\rho}),\Theta)$ the right hand side of the above inequality. $\mathcal{F}$ is referred to as the negative free energy. When $Q(Z,\boldsymbol{\beta},\boldsymbol{\rho})$ is equal to the true joint posterior of $(Z,\boldsymbol{\beta},\boldsymbol{\rho})$ given the data set Y, viz. $p(Z,\boldsymbol{\beta},\boldsymbol{\rho}|Y,\Theta)$, then $\mathcal{F}(Q,\Theta)=\mathcal{L}(\Theta)$. The difference between this lower bound $\mathcal{F}(Q,\Theta)$ and $\mathcal{L}(\Theta)$ is the KL-divergence between the true and approximating posteriors. The Bayesian variational learning is to maximise $\mathcal{F}$ with respect to the distribution Q and the parameters θ alternatively.

These steps are iterated as necessary and are analogous to the Expectation (E) and Maximization (M) steps of the EM algorithm. In implementation, a basic strategy is to separate the dependence between the hidden variables $(Z, \boldsymbol{\beta}, \boldsymbol{\rho})$, that is, to assume that $Q(Z, \boldsymbol{\beta}, \boldsymbol{\rho}) = Q(Z)Q(\boldsymbol{\beta})Q(\boldsymbol{\rho})$. In this case, substituting (6) into the right hand side of (7) results in

$$\begin{aligned}\mathcal{F}(Q,\Theta) = {} & H(Q(Z)) + H(Q(\boldsymbol{\beta})) + H(Q(\boldsymbol{\rho})) + \sum_Z \int_{\boldsymbol{\beta},\boldsymbol{\rho}} Q(Z)Q(\boldsymbol{\beta})Q(\boldsymbol{\rho}) \\ & \Bigg[\sum_{i=1}^N \sum_{j=1}^m z_{ij}\Bigg(\log\pi_j - \frac{d}{2}\log(2\pi) + \frac{1}{2}\sum_{k=1}^d \log\rho_{jk} + \frac{1}{2}\sum_{k=1}^d \log\beta_{ijk} \\ & - \frac{1}{2}\sum_{k=1}^d \rho_{jk}\beta_{ijk}(y_{ik}-\mu_{jk})^2 - \log 2\beta_{ijk}^2 - \frac{1}{2\beta_{ijk}}\Bigg) \\ & + \sum_{j=1}^m\sum_{k=1}^d a_{jk}\log(b_{jk}) + (a_{jk}-1)\log\rho_{jk} - b_{jk}\rho_{jk} - \log\Gamma(a_{jk})\Bigg] d\boldsymbol{\beta} d\boldsymbol{\rho}\end{aligned} \tag{8}$$

where Q is any density function with respect to the corresponding random variables and $H(Q)$ is the entropy of the Q.

In order to simplify the notation in the variational algorithm, denote $\overline{u}$ the mean of u and $\underline{u}$ the expection of $\frac{1}{u}$ with respect to the approximate posterior $Q(u)$. For the L1 mixture model maximizing the variational functional $\mathcal{F}$ with respect to $Q(\boldsymbol{\beta})$, $Q(\boldsymbol{\rho})$, and $Q(Z)$ results in the following approximate distributions:

The optimal $Q(\boldsymbol{\rho})$:

$$Q(\rho_{jk}) \propto \exp\left\{-\widetilde{b}_{jk}\rho_{jk} + (\widetilde{a}_{jk}-1)\log\rho_{jk}\right\} \quad \text{with the mean} \quad \overline{\rho}_{jk} = \widetilde{a}_{jk}/\widetilde{b}_{jk},$$

where $Q(\rho_{jk})$ actually is a Gamma density with parameters

$$\widetilde{a}_{jk} = a_{jk} + \frac{1}{2}\sum_{i=1}^N q_{ij} \quad \text{and} \quad \widetilde{b}_{jk} = b_{jk} + \frac{1}{2}\sum_{i=1}^N q_{ij}(y_{ik}-\mu_{jk})^2\overline{\beta}_{ijk}$$

The optimal $Q(\boldsymbol{\beta})$

$$Q(\beta_{ijk}) \propto \exp\left\{-\frac{1}{2}q_{ij}(y_{ik}-\mu_{jk})^2\overline{\rho}_{jk}\beta_{ijk} - \frac{3}{2}q_{ij}\log(\beta_{ijk}) - q_{ij}\frac{1}{2\beta_{ijk}}\right\}$$

Denote $\lambda_{ij} = 1 - \frac{3}{2}q_{ij}$, $\chi_{ij} = q_{ij}$ and $\psi_{ijk} = q_{ij}(y_{ik}-\mu_{jk})^2\overline{\rho}_{jk}$, then $Q(\beta_{ijk})$ follows the Generalized Inverse Gaussian (GIG) distribution

$$G(\beta_{ijk}|\lambda_{ij},\chi_{ij},\psi_{ijk}) = \frac{(\psi_{ijk}/\chi_{ij})^{\lambda_{ij}/2}}{2K_{\lambda_{ij}}(\sqrt{\chi_{ij}\psi_{ijk}})}\beta_{ijk}^{\lambda_{ij}-1}\exp\left\{-\frac{1}{2}(\frac{\chi_{ij}}{\beta_{ijk}} + \psi_{ijk}\beta_{ijk})\right\}$$

where K_λ is the modified Bessel function of the second kind. Thus we have

$$\overline{\beta}_{ijk} = E_Q(\beta_{ijk}) = \frac{K_{\lambda_{ij}+1}(\sqrt{\chi_{ij}\psi_{ijk}})}{\sqrt{\frac{\psi_{ijk}}{\chi_{ij}}}K_{\lambda_{ij}}(\sqrt{\chi_{ij}\psi_{ijk}})}$$

$$\underline{\beta}_{ijk} = E_Q(\frac{1}{\beta_{ijk}}) = \frac{\sqrt{\frac{\psi_{ijk}}{\chi_{ij}}}K_{\lambda_{ij}-1}(\sqrt{\chi_{ij}\psi_{ijk}})}{K_{\lambda_{ij}}(\sqrt{\chi_{ij}\psi_{ijk}})}$$

For the optimal discrete distribution $Q(z_{ij} = 1) = q_{ij}$, substituting the optimal $Q(\boldsymbol{\rho})$ and $Q(\boldsymbol{\beta})$ into (8) and maximizing $\mathcal{F}$ with respect to q_{ij} result in

$$q_{ij} = \frac{\exp\{\mathcal{E}_{ij}\}}{\sum_{j=1}^{m}\exp\{\mathcal{E}_{ij}\}}$$

where

$$\begin{aligned}\mathcal{E}_{ij} =& \log\pi_j - \frac{1}{2}\sum_{k=1}^{d}\underline{\beta}_{ijk} - \frac{1}{2}\sum_{k=1}^{d}(y_{ik}-\mu_{jk})^2\overline{\beta}_{ijk}\overline{\rho}_{jk}\\ &+\frac{1}{2}\sum_{k=1}^{d}E_Q(\log(\rho_{jk})) - \frac{3}{2}\sum_{k=1}^{d}E_Q(\log(\beta_{ijk}))\end{aligned}$$

In the M-step, we can work out the optimal hyperparameters

$$\widehat{\pi}_j = \frac{1}{N}\sum_{i=1}^{N}q_{ij} \quad \text{and} \quad \boldsymbol{\mu}_j = (\sum_{i=1}^{N}q_{ij}M_{ij})^{-1}\sum_{i=1}^{N}q_{ij}M_{ij}\mathbf{y}_i$$

where $M_{ij} = \mathrm{diag}(\overline{\beta}_{ij1}\overline{\rho}_{j1}, ..., \overline{\beta}_{ijd}\overline{\rho}_{jd})$.

The above procedure has an intuitive interpretation: in the E-step we update one of Q-distributions when keeping others fixed; in M-step we fix all the distributions over the hidden variables and update both $\boldsymbol{\pi}$ and $\boldsymbol{\mu}$ by maximizing $\mathcal{F}$.

4 Experiments

In our experiments, for the Laplacian noise process, we choose $a_\rho = 100$ and $b_\rho = 5$, so as to specify a mean of 4.8 for the prior over the standard derivation. We then cycled through the Q-distribution updates, starting with $Q(\boldsymbol{\rho})$. The initial value for both the centers $\boldsymbol{\mu}$ and q was set up by using the result from the K-mean algorithm with only 5 cycles. The initial value for β was set to the covariance determined by the K-mean algorithm.

When we calculate $\mathcal{E}_{ij}$, we need to calculate both $E_Q(\log(\rho_{jk}))$ and $E_Q(\log(\beta_{ijk}))$. As each ρ_{jk} follows a Gamma distribution, so it can be determined by

$$E_Q(\log(\rho_{jk})) = \Psi(\widehat{a}_{jk}) - \log\widehat{b}_{jk}$$

where Ψ is the Digamma function. Regarding $E_Q(\log(\beta_{ijk}))$ we used the GIG sampler from the matlab function `Randraw.m`[1] for an approximated estimate. We chose to terminate the iterative procedure when there is only little change occuring at each update to the Q-distribution with a tolerance 10^{-6}.

4.1 Synthetic Data

We first demonstrate the performance of the algorithm on bivariate synthetic data. The data set consists of 200 data, 150 of which were sampled from a three-component Gaussian mixture model, and other 50 data from a uniform distribution over the 120% range determined by the first 150 data points. The parameters of the mixture model were

$$\boldsymbol{\mu}_1 = (0,3), \qquad \boldsymbol{\mu}_1 = (3,0), \qquad \boldsymbol{\mu}_3 = (-3,0)$$
$$\Sigma_1 = \begin{pmatrix} 2 & 0.5 \\ 0.5 & 0.5 \end{pmatrix}, \qquad \Sigma_2 = \begin{pmatrix} 1 & 0 \\ 0 & 0.1 \end{pmatrix}, \qquad \Sigma_3 = \begin{pmatrix} 2 & -0.5 \\ -0.5 & 0.5 \end{pmatrix}$$

with mixing coefficients $\pi_1 = \pi_2 = \pi_3 = 1/3$. The data are shown in each plot of Figure 1 in which the noised points are marked as $+$.

Ideally we hope the extra 50 data won't do much impact on the model as the majority data come from a Gaussian mixture model. However it is clear from Figure 1(a) that the standard Gaussian mixture model attempts to model the noised data[2]. However the L1 mixture model has easily revealed the true component patterns without much impact from the noised data, see Figure 1(b). In the experiment, we also noted that the β value associated with the extra outliers is significantly smaller than the β values for the other points. Taking β as an indicator we may drop any possible outliers in the procedure.

4.2 Handwritten Digits

Experiments were also conducted on MNIST handwritten digits database[3] to demonstrate the effectiveness of the L1 mixture model in assisting dimensionality reduction algorithm for the purpose of clustering. Recently several unsupervised dimensionality reduction algorithms based on kernel similarity have been introduced, e.g. KLE (Kernel Laplacian Embedding) [19] and TKE (Twin Kernel Embedding) [20] algorithms. The algorithms are implemented based on the similarity measure provided by a kernel function. We can preprocess the data set with the L1 mixture model to give rough clustering based on the learned information like q_{ij} and to update the kernel measure for example by setting

[1] See http://www.mathworks.com/matlabcentral/fileexchange/loadFile.do?objectId=7309&object Type=file

[2] To compare with the approach proposed in this paper, we deliberately model the data with a diagonal Gaussian mixture model.

[3] MNIST handwritten digits database is available at `http://yann.lecun.com/exdb/mnist`.

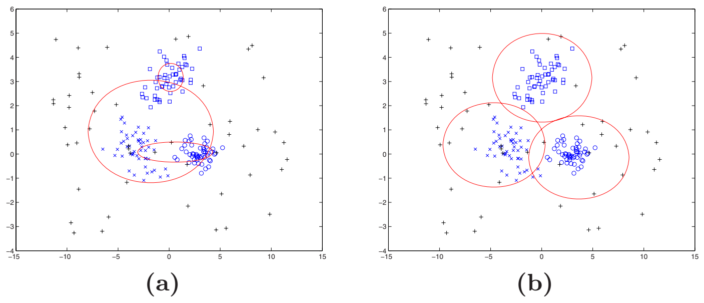


Fig. 1. The comparison of standard Gaussian and L1 Mixtures Models: (a) The estimated three components by using diagonal Gaussian mixture model; (b) Obviously the L1 mixture model revealed the component structures

kernel values to zero (or a small value) for the data pair who are not in the same "cluster" revealed by the L1 mixture model.

We chose 50 images for each digit as the training data. All images are in grayscale and have a uniform size of 28×28 pixels. So the input data are in a very high dimensional space, i.e., $D = 784$.

To test the stability of the L1 mixture model we also added 5 additional corrupted images from each digit. The noise added is generated from a uniform distribution on the interval $[10, 600]$ and then the corrupted images are scaled to the standard gray levels from 0 to 255.

In the implementation of the L1 mixture model for this database, we took the number of component $m = 12$ although actually there are only 10 clusters. The initial values for all the parameters were chosen in the same way as that in the experiment for the 2-D synthetic dataset. However for the number of iteration k, we tested a range of values between 200 to 500 and found that $k = 279$ gave a better visual result. In fact, after 279 iterations, most of the parameter values in Q distributions do not change much.

The comparison between the KLE algorithm with and without the L1 mixture preprocessing is shown in Figure 2. It can be seen that the visual effect from the KLE with the L1 mixture preprocessing is not significantly improved due the complexity of the dataset, however it is very clear that the clusters for both digit 7 (triangle marker) and 5 (diamond marker) in (b) are much more packed than that in (a). In the experiment, the kernel values were shrunk by $1e^{-9}$ when the data pairs are not in the same cluster revealed by the L1 mixture preprocessing.

We conducted another test on only 5 selected digit groups: digits 3, 4, 5, 6 and 7. Figure 3(a) shows the result from the KLE algorithm without any L1 mixture preprocessing while (b) shows the result of KLE algorithm with a simple L1 mixture preprocessing. The result in (b) has a significant improvement from (a). It can be seen that digits 3 and 4 are clearly separated while digit 7 has been compressed into a single point (blue square in (b)).

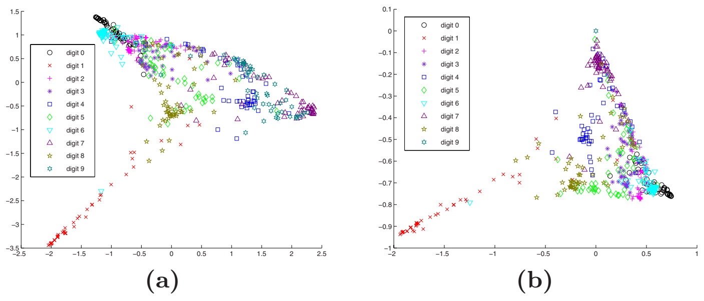


Fig. 2. The comparison of the KLE algorithm with 10 digits: (a) the result without L1 mixture model, (b) the result with L1 mixture model

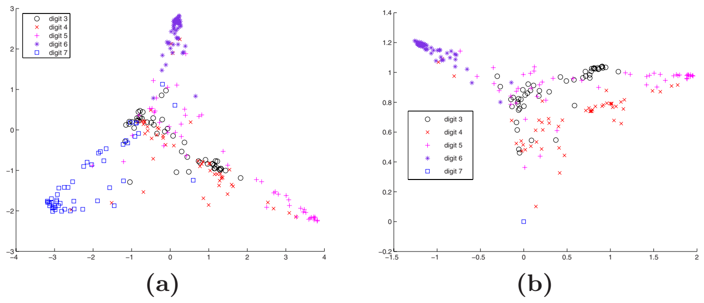


Fig. 3. The comparison of the KLE algorithm with five digits: (a) the result without L1 mixture model, (b) the result with L1 mixture model

5 Conclusions

Many probabilistic models largely rely on a Gaussian assumption. In practice, however, this crude assumption may seem unrealistic as the resulting models are very sensitive to non-Gaussian noise processes. A possible approach is to employ kinds of non-Gaussian distribution especially heavy-tailed distributions such as L1 Laplacian densities. In this paper, we have shown that the L1 mixture model can be constructed and solved under the framework of general variational Bayesian learning and inference. In order to find tractable solutions for the model parameter in L1 mixture model, we express the L1 Laplacian distribution as a superposition of infinite number of Gaussian with precisions controlled by another distribution. Doing so enables us to employ the variational approximate to posterior density of all the uncertainty involved with the model. The algorithm has been designed based on the variational version of EM scheme. The approach works well on several illustrative examples and practical examples.

References

1. Gao, J.: Robust L1 principal component analysis and its bayesian variational inference. Neural Computation (to appear, 2008)
2. McLachlan, G., Peel, D.: Finite Mixture Models. John Wiley, New York (2000)
3. Dempster, A., Laird, N., Rubin, D.: Maximum-likelihood from incomplete data via the EM algorithm. J. Royal Statistical Soceity, Ser. B 39, 1–38 (1977)
4. Tipping, M., Bishop, C.: Mixtures of probabilistic principal component analyzers. Neural Computation 11, 443–482 (1999)
5. Verbeek, J.: Learning nonlinear image manifolds by global alignment of local linear models. IEEE Trans. on Pattern Analysis and Machine Intelligence 28(8), 1236–1250 (2006)
6. Bhowmick, D., Davison, A., Goldstein, D., Ruffieux, Y.: A Laplace mixture model for identification of differential expression in microarray experiments. Biostatistics 7, 630–641 (2006)
7. Jordan, M.: Graphical models. Statistical Science (Special Issue on Bayesian Statistics) 19, 140–155 (2004)
8. Peel, D., McLachlan, G.: Robust mixture modelling using the *t* distribution. Statistic and Computing 10, 339–348 (2000)
9. Ridder, D.D., Franc, V.: Robust subspace mixture models using *t*-distributions. In: Harvey, R., Bangham, A. (eds.) BMVC 2003. Proceedings of the 14th British Machine Vision Conference, pp. 319–328 (2003)
10. Archambeau, C.: Probabilistic models in noisy environments and their application to a visual prosthesis for the blind. Doctoral dissertation, Université Catholique de Louvain, Belgium (2005)
11. Tibshirani, R.: Regression shrinkage and selection via the lasso. J. Royal. Statist. Soc B. 58, 267–288 (1996)
12. Ng, A.: Feature selection, L1 vs. L2 regularization, and rotational invariance. In: Proceedings of Intl Conf. Machine Learning (2004)
13. Jolliffe, I.: Principal component analysis, 2nd edn. Springer, New York (2002)
14. Zou, H., Hastie, T., Tibshirani, R.: Sparse principal component analysis. Technical report, Statistics Department, Stanford University (2004)
15. Park, H.J., Lee, T.W.: Modeling nonlinear dependencies in natural images using mixture of laplacian distribution. In: Saul, L.K., Weiss, Y., Bottou, L. (eds.) Advances in Neural Information Processing Systems 17, pp. 1041–1048. MIT Press, Cambridge (2005)
16. Gao, J., Gunn, S., Kandola, J.: Adapting kernels by variational approach in svm. In: McKay, B., Slaney, J.K. (eds.) AI 2002. LNCS (LNAI), vol. 2557, pp. 395–406. Springer, Heidelberg (2002)
17. Tipping, M., Lawrence, N.: Variational inference for Student-t models: Robust Bayesian interpolation and generalized component analysis. NeuroComputing 69, 123–141 (2005)
18. Pontil, M., Mukherjee, S., Girosi, F.: On the noise model of support vector machine regression. In: A.I. Memo 1651, AI Laboratory, MIT, Cambridge (1998)
19. Guo, Y., Gao, J.B., Kwan, P.W.: Kernel Laplacian eigenmaps for visualization of non-vectorial data. In: Sattar, A., Kang, B.-H. (eds.) AI 2006. LNCS (LNAI), vol. 4304, pp. 1179–1183. Springer, Heidelberg (2006)
20. Guo, Y., Gao, J.B., Kwan, P.W.: Visualization of non-vectorial data using twin kernel embedding. In: Ong, K., Smith-Miles, K., Lee, V., Ng, W. (eds.) AIDM 2006. Proceedings of the International Workshop on Integrating AI and Data Mining, pp. 11–17. IEEE Computer Society Press, Los Alamitos (2006)

On Using a Hierarchy of Twofold Resource Allocation Automata to Solve Stochastic Nonlinear Resource Allocation Problems

Ole-Christoffer Granmo[1] and B. John Oommen[2,⋆]

[1] Dept. of ICT, University of Agder, Grimstad, Norway
[2] School of Computer Science, Carleton University, Ottawa, Canada

Abstract. Recent trends in AI attempt to solve difficult NP-hard problems using intelligent techniques so as to obtain approximately-optimal solutions. In this paper, we consider a *family* of such problems which fall under the general umbrella of "knapsack-like" problems, and demonstrate how we can solve *all of them* fast and accurately using a hierarchy of Learning Automata (LA). In a multitude of real-world situations, resources must be allocated based on incomplete and noisy information, which often renders traditional resource allocation techniques ineffective. This paper addresses one such class of problems, namely, *Stochastic Non-linear Fractional Knapsack Problems*. We first present a completely new on-line LA system — the *Hierarchy of Twofold Resource Allocation Automata* (H-TRAA). The primitive component of the H-TRAA is a *Twofold Resource Allocation Automaton* (TRAA), which in itself possesses novelty in the field of LA. For both the TRAA and H-TRAA, we then provide formal convergence results. Finally, we demonstrate empirically that the H-TRAA provides *orders of magnitude* faster convergence compared to state-of-the-art. Indeed, in contrast to state-of-the-art, the H-TRAA scales sub-linearly. As a result, we believe that the H-TRAA opens avenues for handling demanding real-world applications, such as the allocation of resources in large-scale web monitoring.

1 Introduction

Although the field of Artificial Intelligence (AI) has matured over the past five decades, there are still a myriad of problems yet to be solved. It is true that the literature reports numerous strategies such as the benchmark graph-searching and theorem proving principles (which have proven to be generic methods for numerous problems). More recently, though, a lot of research has gone into finding approximately-optimal solutions for intractable NP-hard problems. This paper demonstrates how we can solve a family of "knapsack-like" problems - all of which are related to resource allocation - using the general LA paradigm.

To present the problem studied in its most general perspective, consider the following scenario: *Imagine that you have to allocate a limited amount of time*

⋆ *Chancellor's Professor*; *Fellow : IEEE* and *Fellow : IAPR*. The Author also holds an *Adjunct Professorship* with the Dept. of ICT, University of Agder, Norway.

among n different activities. The problem is such that spending a time instant on an activity randomly produces one of two possible outcomes — the time instant is either spent "fruitfully" or "unfruitfully". In this generic setting, your goal is to maximize the expected amount of fruitfully spent time. Unfortunately, you are only given the following information regarding the activities: (1) Each instant of time spent on an activity has a certain probability of being fruitful, and (2) this probability decreases with the amount of time spent on the activity. To render the problem even more realistic, you do not have access to the probabilities themselves. Instead, you must rely on solving the problem by means of trial-and-failure, i.e., by attempting different allocations, and observing the resulting random outcomes.

To permit an analytically rigorous treatment, as in [1], the above problem is formulated as a *Stochastic Non-linear Equality Fractional Knapsack (NEFK) Problem*, which is a generalization of the classical *linear* Fractional Knapsack (FK) Problem (explained below), and which has direct applications to resource allocation in web polling, and to the problem of determining the optimal sample size required for estimation purposes.

The Linear Fractional Knapsack (FK) Problem: The linear FK problem is a classical continuous optimization problem which also has applications within the field of resource allocation. The problem involves n materials of different value v_i per unit volume, $1 \leq i \leq n$, where each material is available in a certain amount $x_i \leq b_i$. Let $f_i(x_i)$ denote the value of the amount x_i of material i, i.e., $f_i(x_i) = v_i x_i$. The problem is to fill a knapsack of fixed volume c with the material mix $\boldsymbol{x} = [x_1, \ldots, x_n]$ of maximal value $\sum_1^n f_i(x_i)$ [2].

The Nonlinear Equality FK (NEFK) Problem: One important extension of the above classical problem is the *Nonlinear Equality* FK problem with a separable and concave objective function. The problem can be stated as follows [3]:

$$\begin{aligned}&\text{maximize } f(\boldsymbol{x}) = \textstyle\sum_1^n f_i(x_i)\\ &\text{subject to } \textstyle\sum_1^n x_i = c \text{ and } \forall i \in \{1, \ldots, n\}, x_i \geq 0.\end{aligned}$$

Note that since the objective function is considered to be concave, the value function $f_i(x_i)$ of each material is also concave. This means that the derivatives of the material value functions $f_i(x_i)$ with respect to x_i, (hereafter denoted f_i'), are non-increasing.

The Stochastic NEFK Problem: In this paper we generalize the above NEFK problem. First of all, we let the material value per unit volume for any x_i be a *probability* function $p_i(x_i)$. Furthermore, we consider the distribution of $p_i(x_i)$ to be *unknown.* That is, each time an amount x_i of material i is placed in the knapsack, we are only allowed to observe an instantiation of $p_i(x_i)$ at x_i, and not $p_i(x_i)$ itself. As an additional complication, $p_i(x_i)$ is nonlinear in the sense that it decreases monotonically with x_i, i.e., $x_{i_1} \leq x_{i_2} \Leftrightarrow p_i(x_{i_1}) \geq p_i(x_{i_2})$.

Given this stochastic environment, we intend to devise an on-line incremental scheme that learns the mix of materials of maximal *expected* value, through a series of informed guesses.

Stochastic Knapsack Problems — State-of-the-Art: The first reported generic treatment of the stochastic NEFK problem itself can be found in [1]. Various instantiations of the problem have, however, appeared sporadically, particularly within the web monitoring domain. In these latter instantiations, the unknown parameters are *estimated* by means of a tracking phase where web pages are polled mainly for estimation purposes [4]. One major disadvantage of such an approach is that the parameter estimation phase significantly delays the implementation of an optimal solution. This disadvantage is further aggravated in *dynamic* environments where the optimal solution changes over time, introducing the need for parameter re-estimation [1].

Learning Automata (LA): In contrast to the above approaches, we base our work on the principles of LA [5]. LA have been used to model biological systems [6], and have attracted considerable interest in the last decade because they can learn the optimal actions when operating in (or interacting with) unknown stochastic environments. Furthermore, they combine rapid and accurate convergence with low computational complexity.

The novel Learning Automata Knapsack Game (LAKG) scheme that we proposed in [1] does not rely on estimating parameters, and can be used to solve the stochastic NEFK problem in both static and dynamic settings. Indeed, empirical results verify that the LAKG finds the optimal solution with arbitrary accuracy, guided by the principle of Lagrange Multipliers. Furthermore, the empirical results show that the performance of the LAKG is superior to that of parameter-estimation-based schemes, both in static and dynamic environments. Accordingly, we believe that the LAKG can be considered to represent the state-of-the-art when it concerns the stochastic NEFK problem. This landmark is now extended to develop the TRAA, (which, in itself is the first reported LA which is *artificially* ergodic), and its hierarchical version, the H-TRAA.

Contributions of This Paper: The contributions of this paper are the following: (1) We report the first *analytical* results for schemes that solve the Stochastic NEFK Problem. (2) We propose a novel scheme for the *two-material* resource allocation problem, namely, the *Twofold Resource Allocation Automaton (TRAA)*. As mentioned, from the perspective of LA, the TRAA, in itself, is the first reported LA which is *artificially* rendered ergodic. (3) We provide convergence results for the TRAA. (4) We report the first *hierarchical* solution to the Stochastic NEFK Problem, based on a hierarchy of TRAAs, namely, the H-TRAA, which also is asymptotically optimal. (5) We verify empirically that the H-TRAA provides orders of magnitude faster convergence than the LAKG.

2 A Hierarchy of Twofold Resource Allocation Automata

2.1 Overview of the H-TRAA Solution

In order to put our work in the right perspective, we start this section by providing a brief review of the concepts found in [1] - which are also relevant for more "primitive" variants of the knapsack problem.

As indicated in the introduction, solving the classical linear FK problem involves finding the most valuable mix $\boldsymbol{x}^* = [x_1^*, \dots, x_n^*]$ of n materials that fits within a knapsack of fixed capacity c. The material value per unit volume for each material i is given as a constant v_i, and each material is available in a certain amount $x_i \leq b_i$, $1 \leq i \leq n$. Accordingly, the value of the amount x_i of material i, $f_i(x_i) = v_i x_i$, is linear with respect to x_i. In other words, the derivative of $f_i(x_i)$ — i.e., the material value per unit volume — is fixed: $f_i'(x_i) = v_i$. Because a fraction of each material can be placed in the knapsack, the following greedy algorithm from [2] finds the most valuable mix: *Take as much as possible of the material that is most valuable per unit volume. If there is still room, take as much as possible of the next most valuable material. Continue until the knapsack is full.*

Let us now generalize this and assume that the material unit volume values are *random* variables with *constant* and *known* distributions. Furthermore, for the sake of conceptual clarity, let us only consider binary variables that *either* instantiate to the values of 0 or 1. Since the unit volume values are random, let p_i denote the probability of the unit volume value $v_i = 1$ for material i, $1 \leq i \leq n$, which means that the probability of the unit volume value $v_i = 0$ becomes $1 - p_i$. With some insight, it becomes evident that under such conditions, the above greedy strategy can again be used to maximize the *expected* value of the knapsack, simply by selecting material based on the *expected* unit volume values, $E[v_i] = 0 \times (1 - p_i) + 1 \times p_i$, rather than actual unit volume values.

The above indicated solution is, of course, inadequate when the p_i's are unknown. Furthermore, the problem becomes even more challenging when the p_i's are no longer constant, but rather depend on their respective material amounts x_i, $1 \leq i \leq n$. Let $p_i(x_i)$ denote the probability that the current unit volume value of material i is $v_i = 1$, given that the amount x_i has already been placed in the knapsack. Then, the expected value per unit volume of material i, $1 \leq i \leq n$, becomes $E[v_i] = 0 \times [1 - p_i(x_i)] + 1 \times p_i(x_i) = p_i(x_i)$, and accordingly, the expected value of the amount x_i becomes $f_i(x_i) = \int_0^{x_i} p_i(u)du$.

Our aim, then, is to find a scheme that moves towards optimizing the following NEFK problem on-line:

$$\begin{array}{l}\text{maximize } f(\boldsymbol{x}) = \sum_1^n f_i(x_i), \text{where} f_i(x_i) = \int_0^{x_i} p_i(u)du, \text{ and } p_i(x_i) = f_i'(x_i), \\ \text{subject to } \sum_1^n x_i = c \text{ and } \forall i \in \{1, \dots, n\}, x_i \geq 0.\end{array}$$

Note that we allow only instantiations of the material values per unit volume to be observed. That is, each time an amount x_i of material i is placed in the knapsack, an instantiation v_i at x_i is observed.

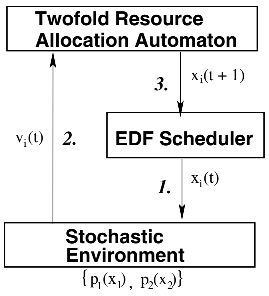


Fig. 1. The Twofold Resource Allocation Automaton (TRAA)

Because of the above intricacies, we approach the problem by relying on informed material mix *guesses*, i.e., by experimenting with different material mixes and learning from the resulting random unit volume value outcomes. We shall assume that x_i is any number in the interval $(0, 1)$. The question of generalizing this will be considered later. The crucial issue that we have to address, then, is that of determining how to change our current guesses on x_i, $1 \leq i \leq n$. We shall attempt to do this in a discretized manner by subdividing the unit interval into N points $\{\frac{1}{N+1}, \frac{2}{N+1}, \dots, \frac{N}{N+1}\}$, where N is the resolution of the learning scheme. It turns out that a larger value of N ultimately implies a more accurate solution to the knapsack problem.

2.2 Details of the TRAA Solution

We first present our LA based solution to *two-material* Stochastic NEFK Problems. The two-material solution forms a critical part of the hierarchic scheme for multiple materials that is presented subsequently. As illustrated in Fig. 1, our solution to two-material problems constitutes of the following three modules:

Stochastic Environment: The *Stochastic Environment* for the two-material case can be characterized by: (1) The capacity c of the knapsack; (2) Two material unit volume value probability functions $p_1(x_1)$ and $p_2(x_2)$. In brief, if the amount x_i of material i is suggested to the Stochastic Environment, the Environment replies with a unit volume value $v_i = 1$ with probability $p_i(x_i)$ and a unit volume value $v_i = 0$ with probability $1 - p_i(x_i)$, $i \in \{1, 2\}$. It should be emphasized that to render the problem both interesting and non-trivial, we assume that $p_i(x_i)$ is unknown to the TRAA.

Twofold Resource Allocation Automaton: The scheme which attempts to learn the optimal allocation $\boldsymbol{x}^* = [x_1^*, x_2^*]$ can be described as follows. A finite fixed structure automaton with the states $s(t) \in \{1, 2, \dots, N\}$ is used to decide the allocation of resources among the two materials. Let the current state of the automaton be $s(t)$. Furthermore, let $q_{s(t)}$ refer to the fraction $\frac{s(t)}{N+1}$, and let $r_{s(t)}$ refer to the fraction: $1 - q_{s(t)}$. Then the automaton's current guess is $\boldsymbol{x} = [q_{s(t)}, r_{s(t)}]$. If the Stochastic Environment tells the automaton that the unit volume value of material i is $v_i(t)$ at time t, the automaton updates its state as follows:

$$\begin{array}{llll} s(t+1) := s(t)+1 & \textbf{If} & \textbf{rand}() \leq r_{s(t)} \textbf{ and } v_i(t)=1 & (1) \\ & & \textbf{and } 1 \leq s_i(t) < N \textbf{ and } i=1 & \\ s(t+1) := s(t)-1 & \textbf{If} & \textbf{rand}() \leq q_{s(t)} \textbf{ and } v_i(t)=1 & (2) \\ & & \textbf{and } 1 < s_i(t) \leq N \textbf{ and } i=2 & \\ s(t+1) := s(t) & \textbf{Otherwise} & & (3). \end{array}$$

Notice how the stochastic state transitions are designed to offset the learning bias introduced by accessing the materials with frequencies proportional to $\boldsymbol{x} = [q_{s(t)}, r_{s(t)}]$. Also observe that the overall learning scheme does not produce any absorbing states, and is, accordingly, ergodic supporting dynamic environments.

Finally, after the automaton has had the opportunity to change its state, it provides output an EDF Scheduler. That is, it outputs the material amounts $\boldsymbol{x} = [q_{s(t+1)}, r_{s(t+1)}]$ that have been changed.

Earliest Deadline First (EDF) Scheduler: The EDF Scheduler takes material amounts $\boldsymbol{x} = [x_1, \ldots, x_n]$ as its input (for the two-material case the input is $\boldsymbol{x} = [x_1, x_2]$). The purpose of the Scheduler is to: (1) provide accesses to the Stochastic Environment in a sequential manner, and (2) make sure that the unit volume value functions are accessed with frequencies proportional to $\boldsymbol{x}$.

The reader should note that our scheme does not rely on accessing the unit volume value functions sequentially with frequencies proportional to $\boldsymbol{x}$ for solving the knapsack problem. However, this restriction is obviously essential for solving the problem *incrementally* and *on-line* (or rather in a "real-time" manner).

Lemma 1. *The material mix $\boldsymbol{x} = [x_1, \ldots, x_n]$ is a solution to a given Stochastic NEFK Problem if (1) the derivatives of the expected material amount values are all equal at $\boldsymbol{x}$, (2) the mix fills the knapsack, and (3) every material amount is positive, i.e.:*

$$f_1'(x_1) = \cdots = f_n'(x_n)$$
$$\textstyle\sum_1^n x_i = c \text{ and } \forall i \in \{1, \ldots, n\}, x_i \geq 0.$$

The above lemma is based on the well-known principle of Lagrange Multipliers [8], and its proof is therefore omitted here for the sake of brevity. We will instead provide our main result for the *two-material* problem and the TRAA. For the two-material problem, let $\boldsymbol{x}^* = [x_1^*, x_2^*]$ denote a solution, as defined above. Note that since x_2^* can be obtained from x_1^*, we will concentrate on finding x_1^*.

Theorem 1. *The TRAA solution scheme specified by (1)–(3) is asymptotically optimal. I.e., as the resolution, N, is increased indefinitely, the expected value of the TRAA output, $x_1(t)$, converges towards the solution of the problem, x_1^*:*

$$\lim_{N\to\infty} \lim_{t\to\infty} E[x_1(t)] \to x_1^*.$$

Proof. The proof is quite involved and is found in [7]. It is omitted here in the interest of brevity. □

2.3 Details of the H-TRAA Solution

In this section we propose a hierarchical scheme for solving n-material problems. The scheme takes advantage of the TRAA's ability to solve two-material problems asymptotically, by organizing them hierarchically.

2.3.1 H-TRAA Construction

The hierarchy of TRAAs, which we hereafter will refer to as H-TRAA, is constructed as follows[1]. First of all, the hierarchy is organized as a balanced binary tree with depth $D = \log_2(n)$. Each node in the hierarchy can be related to three entities: (1) a set of materials, (2) a partitioning of the material set into two subsets of equal size, and (3) a dedicated TRAA that allocates a given amount of resources among the two subsets.

Root Node: The hierarchy root (at depth 1) is assigned the complete set of materials $S_{1,1} = \{1, \ldots, n\}$. These n materials are partitioned into two disjoint and exhaustive subsets of equal size: $S_{2,1}$ and $S_{2,2}$. An associated TRAA, $T_{1,1}$, decides how to divide the full knapsack capacity c (which, for the sake of notational correctness will be referred to as $c_{1,1}$) among the two subsets. That is, subset $S_{2,1}$ receives the capacity $c_{2,1}$ and subset $S_{2,2}$ receives the capacity $c_{2,2}$, with $c_{2,1} + c_{2,2} = c_{1,1}$. Accordingly, *this* TRAA is given the power to prioritize one subset of the materials at the expense of the other.

Nodes at Depth d: Node $j \in \{1, \ldots, 2^{d-1}\}$ at depth d (where $1 < d \leq D$) refers to: (1) the material subset $S_{d,j}$, (2) a partitioning of $S_{d,j}$ into the subsets $S_{d+1,2j-1}$ and $S_{d+1,2j}$, and (3) a dedicated TRAA, $T_{d,j}$. Observe that since level $D+1$ of the H-TRAA is non-existent, we use the convention that $S_{D+1,2j-1}$ and $S_{D+1,2j}$ refer to the primitive materials being processed by the leaf TRAA, $T_{D,j}$. Assume that the materials in $S_{d,j}$ has, as a set, been assigned the capacity $c_{d,j}$. The dedicated TRAA, then, decides how to allocate the assigned capacity $c_{d,j}$ among the subsets $S_{d+1,2j-1}$ and $S_{d+1,2j}$. That is, subset $S_{d+1,2j-1}$ receives the capacity $c_{d+1,2j-1}$ and subset $S_{d+1,2j}$ receives the capacity $c_{d+1,2j}$, with $c_{d+,2j-1} + c_{d+1,2j} = c_{d,j}$.

At depth D, then, each individual material can be separately assigned a fraction of the overall capacity by way of recursion, using the above allocation scheme.

2.3.2 Interaction of H-TRAA with EDF Scheduler and Environment

As in the single TRAA case, H-TRAA interacts with an EDF Scheduler, which suggests which unit volume value function $p_i(x_i)$ to access next. A response is then generated from the Stochastic Environment using $p_i(x_i)$. This response is given to all the TRAAs that were involved in determining the material amount x_i, that is, the TRAAs in the hierarchy that have allocated capacacity to a material subset that contains material i. Finally, a new candidate material mix $\boldsymbol{x} = [x_1, \ldots, x_n]$ is suggested by the H-TRAA to the EDF Scheduler.

[1] We assume that $n = 2^\gamma, \gamma \in \mathbb{N}^+$, for the sake of clarity.

2.3.3 Analysis of the H-TRAA Solution

In the previous section we stated the asymptotic optimality of the individual TRAA. We will now consider the H-TRAA and its optimality. In brief, it turns out that when each individual TRAA in the hierarchy has solved its own two-material problem, a solution to the complete n-material Knapsack Problem has also been produced.

Theorem 2. *Let $T_{d,j}$ be an arbitrary TRAA at level d of the H-TRAA associated with the node whose index is j. Then, if every single TRAA, $T_{d,j}$, in the H-TRAA has found a local solution with proportions $c_{d+1,2j-1}$ and $c_{d+1,2j}$ satisfying $f'_{d+1,2j-1}(c_{d+1,2j-1}) = f'_{d+1,2j}(c_{d+1,2j})$, the overall Knapsack Problem involving n materials that are hierarchically placed in $\log_2 n$ levels of TRAAs, also attains the global optimum solution.*

Proof. The proof of this theorem is also quite deep. It is included in [7], and omitted here due to space limitations. □

3 Experimental Results

The H-TRAA solution has been rigorously tested for numerous cases and the solutions obtained have been, in our opinion, categorically remarkable. Its performance in terms of speed, convergence accuracy and scalability have consistently been far more than we initially anticipated.

In order to achieve a comprehensive test suite, we have conducted our experiments for two classes of objective functions (hereafter referred to as E_i and L_i): $E_i(x_i) = \frac{0.7}{i}(1 - e^{-ix_i})$ and $L_i(x_i) = 0.7 \cdot x_i - \frac{1}{2}i \cdot x_i^2$ for $x_i \leq \frac{0.7}{i}$, with $L_i(x_i) = \frac{0.7^2}{i}$ for $x_i > \frac{0.7}{i}$. The two corresponding probability functions are given presently for a material with index i: $E'_i(x_i) = 0.7 \cdot e^{-i \cdot x_i}$ and $L'_i = \text{Max}\,[0.7 - i \cdot x_i, 0]$.

Given the above considerations, our aim is to find $\boldsymbol{x}^*$, the amounts of the materials that have to be included in the knapsack so as to maximize its value. In the first instance we aim to:

$$\begin{aligned}&\text{maximize } E(\boldsymbol{x}) = \textstyle\sum_1^n E_i(x_i), \text{ where } E_i(x_i) = \int_0^{x_i} E'_i(u)du,\\ &\text{subject to } \textstyle\sum_1^n x_i = c \text{ and } \forall i \in \{1, \ldots, n\}, x_i \geq 0.\end{aligned}$$

The second optimization problem is identical, expect that the E_i and E'_i functions are replaced with L_i and L'_i, respectively.

3.1 H-TRAA Solution

In order to find a H-TRAA Solution to the above problem we must define the Stochastic Environment that the LA are to interact with. As seen in Sect. 2, the Stochastic Environment consists of the unit volume value functions $\mathcal{F}' = \{f'_1(x_1), f'_2(x_2), \ldots, f'_n(x_n)\}$, which are unknown to H-TRAA. We identify the nature of these functions by applying the principle of Lagrange multipliers to the above maximization problems. In short, after some simplification, it can be seen that the following conditions characterize the optimal solution for the exponential functions:

$$E_1'(x_1) = E_2'(x_2) = \cdots = E_n'(x_n)$$
$$\sum_1^n x_i = c \text{ and } \forall i \in \{1, \ldots, n\}, x_i \geq 0.$$

The conditions for the linear functions can be obtained by replacing E_i and E_i' in the equations above with L_i and L_i', respectively.

3.2 Empirical Results

In this section we evaluate our learning scheme by comparing it with three classical policies using synthetic data. We have implemented the following classical policies:

Uniform: The uniform policy allocates resources uniformly. This is the only classical policy that can be applied directly in an unknown environment.
Optimal: The optimal policy finds the optimal solution based on the principle of Lagrange multipliers.
LAKG: The state-of-the-art solution for the nonlinear knapsack problem which is based on the so-called LAKG described in [1].

As we will see in the following, it turns that one of the strengths of the H-TRAA is its ability to take advantage of so-called spatial dependencies among materials. In the above experimental setup, materials are spatially related in the sense that the updating probabilities decreases with the rank-index k. In order to starve the H-TRAA from this information, we opted to perturb this spatial structure. Each perturbation swapped the updating probabilities of a randomly selected material and the material succeeding it in the ranking.

3.2.1 Configuring H-TRAA

The H-TRAA can be configured by various means. First of all, the material amount space $(0, 1)$ need not be discretized uniformly. Instead, a nonlinear material amount space can be formed, as done for the LAKG in [1]. Furthermore, the discretization resolution, N, must also be set for each TRAA, possibly varying from TRAA to TRAA in the hierarchy. In short, the performance achieved for a particular problem can be optimized using these different means of configuring the H-TRAA. In this section, however, our goal is to evaluate the overall performance of the H-TRAA, without resorting to fine tuning. Therefore, we will only use a linear material amount space, as specified in Sect. 2. Furthermore, in the experiments reported here, we have chosen to use the same resolution, $N = 2,000$, for all the TRAAs in the hierarchy, independent of the specific knapsack problem at hand. Thus, our aim is to ensure a fair comparison with the present state of the art, namely, the LAKG scheme.[2]

[2] Since we in this paper emphasize speed of learning, we will presently utilize material unit values of both 0 and 1 (the state transitions for unit value 0 is obtained by inverting the state transitions of the individual TRAAs from Sect. 2).

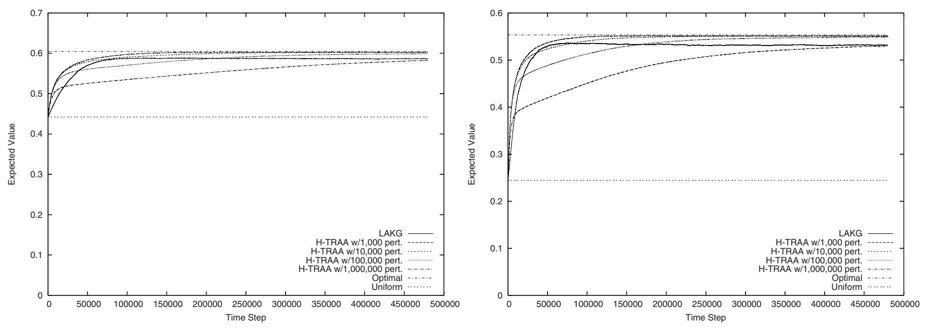


Fig. 2. Convergence for the Exponential (left) and Linear functions (right)

3.2.2 Static Environments

In this section we present the results of the H-TRAA, LAKG and the other schemes for static environments. We see from Fig. 2 that the optimal policy provides a solution superior to the uniform policy solution. The figure also shows that the performance of the H-TRAA increases significantly quicker than the LAKG. However, when the number of perturbations is increased, the performance of the H-TRAA tends to fall. Note that even with 10^6 perturbations, the H-TRAA provides a performance equal to the LAKG if each TRAA in the hierarchy is given a resolution N that is twice as large as the resolution applied by any of its children. Furthermore, the H-TRAA is more flexible than the LAKG, performing either better or similar to LAKG, when it is optimized for the problem at hand. Observe too that the performance of both the H-TRAA and LAKG improve online *without* invoking any parameter estimation phase. Both the LAKG and H-TRAA approach the performance boundary set by the optimal policy, and converge towards near-optimal solutions. However, the H-TRAA converges faster than the LAKG.

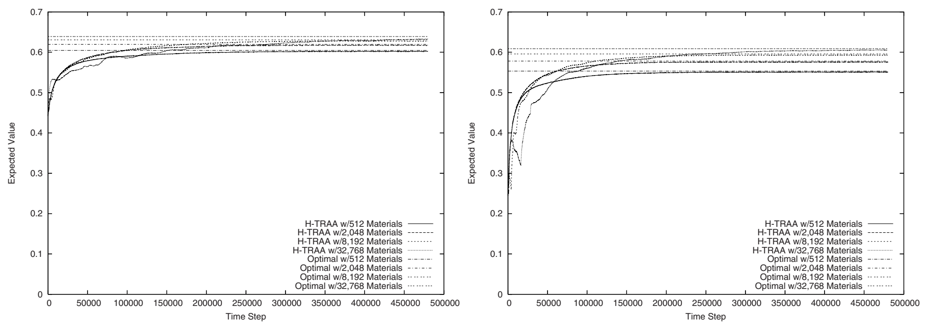


Fig. 3. Convergence as the number of materials is increased significantly

3.2.3 Scalability

One of the motivations for designing the H-TRAA was the improved scalability obtained by means of hierarchical learning. As reported in [7], extending the number of materials significantly increases the convergence time of LAKG. An increased initial learning phase may be unproblematic in cases where the system will run correspondingly longer, adapting to less dramatic changes as they occur. Fig. 3 displays the performance of the H-TRAA under identical settings. The reader should observe that increasing the number of materials does *not* significantly increase its convergence time. It is not unfair to assert that under identical settings, the H-TRAA scheme is far superior to the LAKG. Indeed, the H-TRAA scales sub-linearly with the number of materials.

4 Conclusions and Further Work

In this paper we have considered the fractional knapsack problem and extended the non-LA state-of-the-art in two ways. First of all, we have treated the unit volume values of each material as a *stochastic* variable whose distribution is *unknown*. Secondly, we have worked with the model that the expected value of a material may decrease after each addition to the knapsack. The learning scheme we proposed for solving this knapsack problem was based on a hierarchy of so-called Twofold Resource Allocation Automata (TRAA). Each TRAA works with two materials and moves along the probability space discretized by a resolution parameter, N, with a random walk whose coefficients are not constant. The asymptotic optimality of the TRAA has been stated. We have then presented a formal theory by which an ensemble of TRAAs (the H-TRAA) can be structured in a hierarchical manner to deal with a very large number of materials. Comprehensive experimental results demonstrated that performance of the H-TRAA is superior to previous state-of-the-art schemes.

In our further work, we intend to utilize the H-TRAA solution to resolve the web polling problem, and the problem of determining the optimal sample size required for estimation purposes. We are also currently investigating how other classes of LA can form the basis for solving knapsack-type problems.

References

1. Granmo, O.C., Oommen, B.J., Myrer, S.A., Olsen, M.G.: Learning Automata-based Solutions to the Nonlinear Fractional Knapsack Problem with Applications to Optimal Resource Allocation. IEEE Transactions on Systems, Man, and Cybernetics, Part B (2006)
2. Black, P.E.: Fractional knapsack problem. Dictionary of Algorithms and Data Structures (2004)
3. Kellerer, H., Pferschy, U., Pisinger, D.: Knapsack Problems. Springer, Heidelberg (2004)
4. Pandey, S., Ramamritham, K., Chakrabarti, S.: Monitoring the Dynamic Web to Respond to Continuous Queries. In: 12th International World Wide Web Conference, pp. 659–668. ACM Press, New York (2003)

5. Narendra, K.S., Thathachar, M.A.L.: Learning Automata: An Introduction. Prentice-Hall, Englewood Cliffs (1989)
6. Tsetlin, M.L.: Automaton Theory and Modeling of Biological Systems. Academic Press, London (1973)
7. Granmo, O.C., Oommen, B.J.: Solving stochastic nonlinear resource allocation problems using a hierarchy of twofold resource allocation automata, Unabridged version of this paper (submitted for publication)
8. Bretthauer, K.M., Shetty, B.: The Nonlinear Knapsack Problem — Algorithms and Applications. European Journal of Operational Research 138, 459–472 (2002)

An Investigation into the Interaction Between Feature Selection and Discretization: Learning How and When to Read Numbers

Sumukh Ghodke[1] and Timothy Baldwin[1,2]

[1] Department of Computer Science and Software Engineering
University of Melbourne, VIC 3010, Australia
[2] NICTA Victoria Laboratories
University of Melbourne, VIC 3010, Australia
sumukh.ghodke@gmail.com, tim@csse.unimelb.edu.au

Abstract. Pre-processing is an important part of machine learning, and has been shown to significantly improve the performance of classifiers. In this paper, we take a selection of pre-processing methods—focusing specifically on discretization and feature selection—and empirically examine their combined effect on classifier performance. In our experiments, we take 11 standard datasets and a selection of standard machine learning algorithms, namely one-R, ID3, naive Bayes, and IB1, and explore the impact of different forms of preprocessing on each combination of dataset and algorithm. We find that in general the combination of wrapper-based forward selection and naive supervised methods of discretization yield consistently above-baseline results.

1 Introduction

Machine learning is an experimental science encompassing the areas of probability theory, statistics, information theory and theoretical computer science. It deals with the automatic extraction of useful and comprehensible knowledge from data. Much of machine learning is focused on classification and other predictive tasks, where the value of an attribute is predicted based on prior knowledge.

In classification, a model is built based on known data which is then used to predict the discrete values of new data, in the form of a class label. The input to a classifier is called the training data, while the unseen data to which results of the learning process are applied is called the test data. The classification process can be modelled as a pipeline of one or more components, obligatorily containing a classifier module, and optionally including a pre-processor module. When a pre-processor is present it precedes the classifier and transforms the input data into a form that (ideally) enhances classifier performance.

Data is the crux of all classification tasks. The way in which data is presented to a learning algorithm often has a significant impact on its performance. In classification tasks, data is usually represented in the form of instances, i.e. single data records characterised by one or more attributes (also known as features) and a unique class label. For our purposes, we assume that there are two types of attributes: **discrete** attributes, where there is a finite set of values with no explicit or implicit order (e.g. HIGH, MEDIUM or LOW as values for a TEMPERATURE feature); and **continuous** attributes, which take real numeric values (e.g. a temperature value in degrees Celcius).

There are many ways to pre-process data when building a classifier, but in this paper we focus exclusively on discretization and feature selection. The aim of discretization is to convert continuous attributes into meaningful discrete attributes. Feature selection is the process of pre-selecting a subset of features to train the classifier over, generally with the intention of removing redundant and/or irrelevant attributes from the dataset.

1.1 Case Study

Due to the importance of pre-processing in machine learning, there is a considerable volume of literature proposing new methods or comparing existing ones. Examples of such studies include [5], [3], [9] and [10] where the effects of discretization and feature selection on classifier performance were studied independently. As many learners prefer to operate over discrete attributes, it is important to select the best discretization method for a given dataset. Things are complicated further when feature selection is included as a second form of preprocessing.

The task of finding the best combination of feature selection and discretization has largely been based on heuristics or prior knowledge with similar datasets and exhaustive experimentation. Our study attempts to unearth general trends in the interaction between feature selection and discretization by applying various combinations of methods to a range of datasets and evaluating their output relative to a sample of learner algorithms.

1.2 A New Framework for Machine Learning Experimentation

While the machine learning community has generally been good at sharing datasets and toolkits, and developing fully automated evaluation methodologies, it would be a stretch to say that full reproducibility is the norm in machine learning research. This is due to effects such as a lack of documentation of data splits/sampling, variability in the preprocessing of data, unpublished parameter settings and scope for interpretation in the implementation of learning algorithms. As part of this research, we have developed an open-source machine learning toolkit from scratch which includes all code used to perform these experiments, all datasets in exactly the form used to carry out our experimentation, and a script to automatically run all experiments in their entirety, for maximum transparency. All the above features were built into a new machine learning component of the NLTK-Lite toolkit [2], which is available from `nltk.sourceforge.net`.

An object-oriented modular implementation of the algorithms targeted in this research enhances the readability and understandability of the implemented algorithms. Additionally, the choice of python as the programming language makes the implementations much closer to pseudocode than is the case with existing machine learning toolkits. It also has a fair coverage of algorithms required for experimentation, bundled with 11 datasets from the UCI machine learning repository. It provides tools and utilities which enhance productivity and experimental reproducibility like the presence of a batch testing utility.

2 Methodology

This section outlines the algorithms used in our case study, then goes on to describe the discretization and feature selection techniques. A brief description of the datasets used in our experiments is then followed by findings from past research.

2.1 Classification Algorithms

Our toolkit currently implements five classification algorithms: (1) zero-R, as a baseline classifier; (2) the one-R decision stump induction algorithm; (3) the ID3 decision tree induction algorithm; (4) the naive Bayes algorithm; and (5) the IB1 instance-based learning algorithm. For full descriptions of these algorithms, see [7], [6] and [1]. Note that of these algorithms, naive Bayes and IB1 have an in-built handling of continuous attributes (in the form of a Gaussian PDF, and continuous distance metric, respectively). The remainder of the algorithms are implemented to ignore all continuous attributes, i.e. to operate only over the discrete attributes in a given dataset.

2.2 Discretization Methods

Discretization is the process of converting continuous attributes into discrete ones. Some classification algorithms like C4.5 [6] have the facility to perform localized discretization at a given node in a decision tree. For our purposes, however, we focus exclusively on global discretization, in discretizing the training dataset once to produce discrete intervals for each attribute. The intervals are then used to map continuous values to discrete values for both the training and test datasets.

Discretizers can be categorised into **supervised** and **unsupervised** methods, based on whether they take the labels of the training instances into account in determining the breakpoints between discrete intervals. We focus on two unsupervised discretization algorithms, namely unsupervised equal width and unsupervised equal frequency, and two supervised methods of discretization, the first of which has two modified variants.

Unsupervised Equal Width (UEW). The unsupervised equal width method is the simplest of all discretization algorithms. In this algorithm, the lowest and highest continuous attributes are used to define the input range. This range is then divided into a fixed number of intervals of equal width. If the attribute values range from x_{min} to x_{max} and the number of intervals is given by k, the width of each interval δ is given by $\delta = \frac{x_{max} - x_{min}}{k}$. k intervals are created with the first interval starting from x_{min} by adding δ to the end of each previous interval. In our experiments, k was set to 10.

The main disadvantage of equal width discretization is that it is severely affected by outliers. The intervals are created solely based on the extreme values, which may be erroneous data.

Unsupervised Equal Frequency (UEF). The unsupervised equal frequency discretization (a.k.a. histogram equalization) method attempts to improve on equal width discretization by focussing on individual continuous values, rather than on extreme values in an attribute. A breakpoint is inserted in the ordered sequence of continuous values after every k^{th} instance in the sequence ($k = 10$ in all our experiments). If the values on either side of a breakpoint are the same the breakpoint is shifted until it separates two distinct values. This process thus creates unevenly sized intervals and gives a closer representation to the actual distribution of continuous values.

Despite its simplicity, this method has been shown to give excellent results when combined with some algorithms, e.g. with k set to the square root of the number of instances [8]. K-means clustering can also be used to find the value of k, but, in general, the task of calculating and specifying k still rests with the user, which is a disadvantage of this method.

Naive Supervised Discretization (NS). The naive supervised discretization method is supervised. When discretizing using this algorithm, continuous attribute values are evaluated along with their corresponding class labels. The list of continuous values is sorted to form an ordered sequence and breakpoints are placed at all positions where the class membership changes. The breakpoints are then used to form intervals, where each interval represents a discrete value.

This algorithm is simple to implement and guarantees that the classes of instances within an interval are the same in almost all cases. As breakpoints are placed on the basis of change in class membership, it lacks any sense of ordering. In the worst scenario, if the class membership changes for every instance there will be as many discrete values as there are instances. The other disadvantage of this algorithm is that it can lead to overfitting. Two modified versions of this algorithm are discussed in the following subsections, which change the way breakpoints are inserted in order to overcome the issue of overfitting.

Naive Supervised Modified Version 1 (NS1). In this modified version of naive supervised discretization, the breakpoints are not inserted every time the class membership changes. Instead, they are inserted only after a minimum of N instances of the majority class are contained in each interval. The threshold number of instances is a user defined value, and should be computed based on the type of data. For our experiments, we set N to one-fifteenth the number of training instances.

Naive Supervised Modified Version 2 (NS2). The second modified version tries to avoid overfitting by merging two previously formed intervals until they contain at least N instances of the majority class. This can lead to results which are quite different from the first version, depending on how frequently the class membership changes and the value of N. Once again, we set N to one-fifteenth the number of training instances in our experiments.

Entropy-based Supervised Discretization (ES). In the final form of supervised discretization, we identify the interval with the highest entropy and select the binary breakpoint which minimizes the mean information of the two generated intervals. This is carried out recursively until N partitions are produced.

2.3 Feature Selection

The task of feature selection is to remove low-utility attributes such that the resulting attribute set generates a simpler classifier without compromising the classifier's performance. To be completely sure of the attribute selection, we would ideally have to test all the enumerations of attribute subsets, which is infeasible in most cases as it will result in 2^n subsets of n attributes. The two main types of feature selection methods are filter and wrapper methods, each of which use greedy search strategies to reduce the search space.

Filter feature selection methods evaluate attributes prior to the learning process, and without specific reference to the classification algorithm that will be used to generate the final classifier. The filtered dataset may then be used by any classification algorithms. Rank-based feature selection is a commonly-used filter method and is discussed in the following section.

Wrapper methods use a controlled enumeration strategy and apply the classification algorithm that will be used to generate the final classifier to test the performance of each attribute subset. The ways in which the attribute sets are incrementally updated results in two types of wrapper models: forward selection and backward elimination, as detailed below.

Rank based Feature Selection (Rank). Rank based feature selection is one of the simplest and most efficient means of filter-based feature selection. In this process each feature is evaluated in turn to assign a relative figure of merit, based on which the features are ranked. The rank can be calculated using information gain (**Rank-IG**) or gain ratio (**Rank-GR**), for example. Once the attributes are ranked, the top N ranked attributes are selected. In our experiments, we set N to two-thirds the number of attributes for datasets with less than 10 attributes, half the number of attributes for datasets with between 10 and 20 attributes, and 20 attributes for larger datasets (up to 100 attributes).

A disadvantage of this algorithm is that it does not capture the interaction of features or detect feature redundancy. The algorithm used to rank attributes also tends to have biases, and requires that the user specifies a value for N.

Forward Selection (FS). In forward selection, we start with a seed set of zero attributes and add in one attribute at a time to the seed set [4]. Each feature subset thus formed is evaluated using stratified cross validation, and a greedy selection strategy is used to select the attribute set which results in the highest performance at each level. This attribute is then added to the seed set, and each of the remaining attributes is experimentally added to the expanded feature set on the next iteration of the algorithm. This process continues until all attributes are included, or the increment in classifier performance falls below a δ value.

Backward Elimination (BE). Backward elimination is similar to forward selection in operation, except that the starting state is all attributes, from which one attribute is removed at a time until the increment in classifier performance from one iteration to the next fails to grow. The backward elimination algorithm tends to favour a higher number of attributes as compared to forward selection.

Table 1. Outline of the UCI datasets used in this research

Dataset name	No. of instances	No. of discrete attributes	No. of continuous attributes	No. of classes
`annealing`	798	32	6	6
`australian`	690	8	6	2
`breast-cancer`	286	9	0	2
`diabetes`	768	0	8	2
`german`	1000	13	7	2
`glass`	214	0	10	7
`heart`	270	8	5	2
`iris`	150	0	4	3
`monks-1`	124	7	0	2
`monks-2`	169	7	0	2
`monks-3`	122	7	0	2
`vehicle`	846	0	18	4
`votes`	435	16	0	2

2.4 Datasets

Datasets from the UCI machine learning repository are commonly used to benchmark machine learning research. The data used in our experiment was imported into NLTK-Lite (including establishing canonical data splits where they weren't predefined) to facilitate full reproducibility when conducting future experiments using our toolkit. Table 1 lists the datasets imported along with instance, attribute and class composition.

3 Past Research

Effect of discretization. Dougherty et al. [3] compared the effects of unsupervised and supervised methods of discretization. They identified the characteristics of individual methods and performed an empirical evaluation by experimenting with different discretization methods and classifiers.

The results obtained from their study show that the performance of classifiers almost always improves with global discretization. It is also found that the discretized version of the naive Bayes algorithm slightly outperforms the C4.5 algorithm on these datasets. A combination of naive Bayes with entropy based supervised discretization is said to have resulted in marked improvements, although even the binning method of discretization improved results for the naive Bayes classifiers. The supervised version of discretization is claimed to be generally better than the unsupervised method. The relatively small effects of supervised methods on C4.5 in general was attributed to either the induction algorithm not being able to fully utilize the local discretization options or the local discretization not helping with the induction algorithm on the given algorithms.

These general results were independently verified by Yang and Webb [10] with respect to the naive Bayes algorithm, at the same time as looking at a wider variety of supervised discretization methods and proposing a novel method.

Effect of feature selection. John et al. [5] conducted experiments with the ID3 and C4.5 algorithms on 6 artificial and real datasets, and combinations of filter and wrapper methods of feature selection, to identify general trends. They observed that the wrapper method of feature selection does not generally improve the performance of classifiers significantly. The size of decision trees constructed using these datasets was smaller than ones built using other data sets, although there was an increase in accuracy in a few cases.

4 Experiments

To analyse the effects of pre-processing on classifier performance, we take a range of datasets and classification algorithms, and apply a selection of combinations of pre-processing methods to each. In this, we first discretize each dataset using one of 6 different discretization algorithms (see above). We then optionally perform feature selection (including on non-discretized data) based on one of four methods (see above). In sum, pre-processing results in 4 different types of dataset: unchanged datasets (without discretization or feature selection), datasets with discretized features, datasets with a subset of attributes based on feature selection, and discretized, feature-selected datasets. We evaluate the performance of each classification algorithm over the different pre-processed versions of a given dataset. In all cases, we use simple classification accuracy to evaluate our classifiers.

For all our experiments, we use a zero-R classifier without any preprocessing as our baseline classifier. All results are presented in terms of error rate reduction (e.r.r.) relative to the classification accuracy for this baseline.

In the experiments related to wrapper based methods, 25-fold stratified cross validation is used internally to find the accuracy at each level, and a delta value of 0.1 on classification accuracy is used as the stopping criterion.

Classification is performed based on 5-fold stratified cross validation on each data set and the reported accuracies are the average of results obtained in each trial.

As mentioned above, the complete code and suite of datasets used to run these experiments is contained in the NLTK-Lite distribution.

5 Results

For easier analysis, the classification accuracy of each classifier is calculated with respect to the baseline performance. Averages and variations of accuracy improvements (or degradations) over all datasets are given in Table 2.

5.1 Observations

From Table 2, we observe that wrapper-based feature selection almost always results in improved or equal performance relative to no feature selection, for a given discretization method. Of the two wrapper-based methods, forward selection (FS) is in most cases superior to backward elimination (BE). There are some interesting incompatibilities with the wrapper-based feature selection methods:

Table 2. Mean error rate reduction and standard deviation (in parentheses) over the 11 datasets, relative to a Zero-R baseline (the highest mean e.r.r. for each algorithm is presented in **bold face**)

	Feature Selection	**Discretization**						
		None	UEW	UEF	NS	NS1	NS2	ES
1-R	None	.1130 (±.1988)	.3024 (±.2506)	.3042 (±.2548)	.3029 (±.2428)	.2980 (±.2578)	.3038 (±.2617)	.2765 (±.2541)
	Rank-IG	.1130 (±.1988)	.3024 (±.2506)	.3042 (±.2548)	.3029 (±.2428)	.2980 (±.2578)	.3038 (±.2617)	.2765 (±.2541)
	Rank-GR	.1130 (±.1988)	.3024 (±.2506)	.3042 (±.2548)	.3029 (±.2428)	.2980 (±.2578)	.3038 (±.2617)	.2765 (±.2541)
	FS	.2033 (±.1827)	.3083 (±.2461)	.3101 (±.2504)	**.3209** (±.2469)	.3106 (±.2518)	.3148 (±.2538)	.2908 (±.2511)
	BE	.2033 (±.1827)	.3024 (±.2506)	.3042 (±.2548)	.3029 (±.2428)	.2980 (±.2578)	.3038 (±.2617)	.2765 (±.2541)
ID3	None	.1041 (±.2050)	.3134 (±.2502)	.3149 (±.2519)	.2750 (±.2568)	.3098 (±.2475)	.3078 (±.2460)	.2946 (±.2389)
	Rank-IG	.1014 (±.1972)	.3169 (±.2493)	.3117 (±.2511)	.2776 (±.2620)	.3160 (±.2492)	.3046 (±.2520)	.2956 (±.2399)
	Rank-GR	.1038 (±.1950)	.3109 (±.2538)	.3037 (±.2581)	.2847 (±.2564)	.3054 (±.2551)	.3036 (±.2516)	.2868 (±.2450)
	FS	.2552 (±.2280)	.3083 (±.2461)	**.3222** (±.2654)	.3209 (±.2469)	.3106 (±.2518)	.3148 (±.2538)	.2908 (±.2511)
	BE	.2244 (±.2188)	.3134 (±.2502)	.3149 (±.2519)	.3065 (±.2344)	.3098 (±.2475)	.3078 (±.2460)	.2946 (±.2389)
NB	None	.2756 (±.2366)	.2662 (±.2678)	.2743 (±.2687)	.2902 (±.2646)	.2758 (±.2673)	.2753 (±.2660)	.2283 (±.2399)
	Rank-IG	.2411 (±.2187)	.2841 (±.2682)	.2848 (±.2580)	.3120 (±.2684)	.2909 (±.2647)	.2990 (±.2706)	.2596 (±.2423)
	Rank-GR	.2411 (±.2187)	.2880 (±.2680)	.2859 (±.2573)	.3084 (±.2619)	.2856 (±.2609)	.2920 (±.2649)	.2587 (±.2423)
	FS	.2875 (±.2325)	.3031 (±.2539)	.3055 (±.2555)	**.3157** (±.2515)	.2991 (±.2531)	.3081 (±.2610)	.2855 (±.2564)
	BE	.2756 (±.2366)	.2662 (±.2678)	.2743 (±.2687)	.2902 (±.2646)	.2758 (±.2673)	.2753 (±.2660)	.2283 (±.2399)
IB1	None	.2444 (±.2895)	.3225 (±.2649)	.3249 (±.2626)	.3439 (±.2598)	.3252 (±.2611)	.3388 (±.2604)	.3081 (±.2341)
	Rank-IG	.2297 (±.2382)	.3288 (±.2632)	.3251 (±.2657)	.3487 (±.2472)	.3279 (±.2549)	.3436 (±.2490)	.3094 (±.2325)
	Rank-GR	.2448 (±.2354)	.3250 (±.2706)	.3183 (±.2673)	**.3509** (±.2556)	.3176 (±.2590)	.3330 (±.2585)	.3011 (±.2378)
	FS	.2939 (±.2397)	.3083 (±.2461)	.3227 (±.2660)	.3209 (±.2469)	.3106 (±.2518)	.3148 (±.2538)	.2908 (±.2511)
	BE	.2444 (±.2895)	.3225 (±.2649)	.3249 (±.2626)	.3439 (±.2598)	.3252 (±.2611)	.3388 (±.2604)	.2785 (±.2362)

forward selection is the worst of the feature selection methods (including no feature selection) in the majority of cases when combined with the IB1 algorithm; backward selection, on the other hand, has almost no impact when combined with each of the supervised discretization methods, and has no impact whatsoever in combination with naive Bayes (NB) and IB1 in all but one case. In general, forward selection tends to select a very small number of attributes (often just a single attribute!), whereas backward selection *eliminates* a relatively small number of attributes.

There is little separating the two rank-based filter feature selection methods (Rank-IG and Rank-GR), perhaps unsurprisingly given that they are both based on mean information. Both have absolutely no impact on the performance of one-R, as it uses largely the same means of selecting a single attribute to form a decision stump as the rank-based feature selection methods, meaning that the preferred attribute for one-R is always included in the selected subset of attributes.

It is interesting to note that both one-R and ID3 carry out implicit feature selection in the process of generating a decision stump and decision tree, respectively, and yet both benefit from explicit feature selection in pre-processing. That is, it appears to be the case that feature selection optimizes classifier performance even in cases where the classification algorithm carries out its own feature selection.

Supervised discretization generally outperforms unsupervised discretization. Naive supervised discretization (NS) is the pick of the supervised methods, surprisingly given its inherent naivety, and the two modified variants of the basic method generate a marked improvement in results only in combination with ID3. Entropy based supervised discretization (ES) is the weakest of the supervised methods. Of the two unsupervised discretization methods, unsupervised equal frequency (UEF) performs the best, and in fact consistently outperforms the supervised discretization methods in combination with ID3.

Recall that naive Bayes and IB1 both have in-built mechanisms for handling continuous attributes, such that it is meaningful to compare the results of the different discretization methods with the native algorithms. There is a remarkable increment in discretizing the data and effectively bypassing this facility in both cases, with naive supervised discretization being the pick of the discretization methods in almost all cases. That is, despite the ability of these algorithms to deal natively with continuous attributes, better results are achieved through explicit pre-discretization of the data.

Importantly, feature selection tends to complement discretization, i.e. any gains in performance tend to be additive when the two forms of pre-processing are combined.

To summarise, from Table 2 we can conclude that the combination of naive supervised discretization and forward selection wrapper based feature selection is optimal across different datasets and classification algorithms.

6 Conclusions

We have surveyed a range of pre-existing discretization and feature selection methods, and investigated their performance in combination with a selection of classification algorithms when used to preprocess a range of datasets. The results of our experiments show that the forward selection wrapper based method is the pick of the feature selection algorithms, and complements even classification algorithms which include implicit feature selection of their own. Among the discretization methods, the naive supervised method was found to perform best in most cases, and that explicitly discretizing the data is superior to relying on native handling of continuous attributes within naive Bayes and IB1. Additionally, these two forms of preprocessing complement each other to often improve the overall performance of a given classification algorithm over a given dataset.

References

1. Aha, D.W., Kibler, D., Albert, M.K.: Instance-based learning algorithms. Machine Learning 6(1), 37–66 (1991)
2. Bird, S.: NLTK-Lite: Efficient scripting for natural language processing. In: Bird, S. (ed.) ICON. Proc. of the 4th International Conference on Natural Language Processing, Kanpur, India, pp. 11–18 (2005)
3. Dougherty, J., Kohavi, R., Sahami, M.: Supervised and unsupervised discretization of continuous features. In: Proc. of the 12th International Conference on Machine Learning, pp. 194–202 (1995)
4. Draper, N.R., Smith, H.: Applied Regression Analysis. Wiley-Interscience, Chichester (1998)
5. John, G.H., Kohavi, R., Pfleger, K.: Irrelevant features and the subset selection problem. In: Proc. of the 11th International Conference on Machine Learning, pp. 121–129 (1994)
6. Quinlan, J.R.: Induction of decision trees. In: Shavlik, J.W., Dietterich, T.G. (eds.) Readings in Machine Learning, Morgan Kaufmann, San Francisco (1990)
7. Tan, P.-N., Steinbach, M., Kumar, V.: Introduction to Data Mining. Addison Wesley, Reading (2006)
8. Witten, I.H., Frank, E.: Data Mining: Practical Machine Learning Tools and Techniques. Morgan Kaufmann, San Francisco (2005)
9. Yang, Y., Pedersen, J.O.: A comparative study on feature selection in text categorization. In: Proc. of the 14th International Conference on Machine Learning (1997)
10. Yang, Y., Webb, G.I.: On why discretization works for Naive-Bayes classifiers. In: Gedeon, T.D., Fung, L.C.C. (eds.) AI 2003. LNCS (LNAI), vol. 2903, pp. 440–452. Springer, Heidelberg (2003)

Safe State Abstraction and Reusable Continuing Subtasks in Hierarchical Reinforcement Learning

Bernhard Hengst[1,2]

[1] Making Sense of Data Research Group, NICTA, Sydney, Australia
[2] Computer Science and Engineering, UNSW, Sydney, Australia
bernhard.hengst@nicta.com.au

Abstract. Hierarchical reinforcement learning methods have not been able to simultaneously abstract and reuse subtasks with discounted value functions. The contribution of this paper is to introduce two completion functions that jointly decompose the value function hierarchically to solve this problem. The significance of this result is that the benefits of hierarchical reinforcement learning can be extended to discounted value functions and to continuing (infinite horizon) reinforcement learning problems. This paper demonstrates the method with the an algorithm that discovers subtasks automatically. An example is given where the optimum policy requires a subtask never to terminate.

Keywords: hierarchical reinforcement learning, state abstraction, task hierarchies, decomposition.

1 Introduction

Reinforcement learning (RL) and Markov Decision Problems(MDP)[1] are appropriate machine learning techniques when rational agents need to learn to act with delayed reward in unknown stochastic environments. It is well known that the state space in RL generally grows exponentially with the number of state variables. Approaches to contain the state space explosion include function approximation and state abstraction in hierarchical reinforcement learning (HRL). More recent approaches to HRL include include HAMQ [2], Options [3] and MAXQ [4]. This paper solves an important but specialised problem in HRL, namely, that simple hierarchical decomposition of the value function is incompatible with discounting. Given space limitations, we will assume that the reader is somewhat familiar with both RL and MAXQ.

We use a simple maze to motivate the problem and to demonstrate the solution. The maze on the left in figure 1 depicts nine interconnected 25 grid position rooms. One-step agent actions are North, South, East or West with reward -1 per step. The task is to find the minimum cost policy from anywhere in the maze to the exit marked "goal". A state is defined by a room variable and a position-in-room variable. Thus we have a total of 225 states. Reinforcement learning can find the shortest route out of the maze. 900 table entries are required to store the action-value function with Q-learning [1].

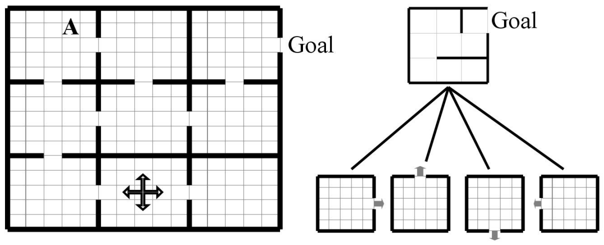


Fig. 1. A simple room maze problem (left) and its decomposed task hierarchy (right)

HRL can decompose the problem into a task hierarchy of smaller subproblems. A task hierarchy for the maze is shown on the right in figure 1. As all the rooms are assumed to be internally equivalent, it is only necessary to learn how to leave a typical room. This room-leaving skill can then be transferred and reused in any room without the need to relearn it. The compass bearing action description means that four room-leaving skills are required, one to leave a room by the North, South, East or West doorway. It is now possible to represent the maze problem more abstractly with just 9 abstract room states and 4 abstract room-leaving actions. The abstract problem is represented by the top level in the task hierarchy in figure 1 (right). This is a small semi-MDP. The 4 room-leaving skills are the policies learnt at the lower level. Subtask policies are interpreted as abstract (or temporally extended) actions from the perspective of the level above.

State abstraction refers to the aggregation of base level states to capture some invariant structure of the problem. For example the same position in each room of the original problem may be aggregated into one abstract position-in-room state. This type of state abstraction is related to eliminating irrelevant variables [4] and model minimization [5]. The room identity is irrelevant to the navigation policies inside rooms. Another type of abstraction is to aggregate all the positions in a room into one abstract state. Each room becomes a state. Using both types of state abstraction it is possible to reduce the storage requirements to represent the value function for the simple maze problem by more than 50% to 436 Q values. At the top level in figure 1 (right) there are 9 abstract states and 4 abstract actions requiring 36 values. At the bottom level, the four subtasks require 25×4 Q values each. State abstraction has been shown to be important for scaling in HRL [6,7,4,8].

Reinforcement learning uses an optimality criterion such as maximising the sum of future rewards from any state. In HRL this state value function may be decomposed over the task hierarchy. To reconstitute the overall value of a state, the rewards accumulated inside a subtask are added to the sum of rewards after leaving the subtask. The latter sum is called the completion value in MAXQ [4]. In this way a state value is calculated by summing the completion values of all the subtasks in a path to the leaf node for any hierarchical policy.

For example, consider an agent that is in position A in figure 1 with reward -1 per step. The nine room states at the top level, only store the completion value following a transition into a new room. The distance to reach the goal *after* leaving the room from A to the south (the "south" completion value) is the same as the distance to reach the goal *after* leaving the room to the east (the "east" completion value). The top level completion values in this instance are not sufficient to determine the shortest path to the goal. It is necessary to take into consideration the local distance to each doorway. From A it is better to try to exit the room via the East doorway. To determine the overall value function therefore, the distance to each doorway is added to the completion values at the top level and the overall shortest path found. In general determining the optimal value for any state requires a search by branching down the task hierarchy for that state (a best first search).

The property that makes subtask abstraction possible, is that the completion values at higher levels are independent of the value of the current state of subtasks.

To ensure that components of the decomposed value function are additive when state abstraction is introduced, MAXQ must use *undiscounted* rewards [4]. Unfortunately, when the value function is defined as the sum of future *discounted* rewards, the simple additive property breaks down. The reason is that parent task completion values for a child subtask depend on the number of steps to terminate the subtask. Discounting means that they will differ in general for each state in the subtask. In other words, discounting does not permit subtask abstraction with a single decomposed value function. Dietterich identified this problem in his discussion of *result distribution irrelevance* [4], concluding that subtask abstractions are only possible in an *undiscounted* setting for MAXQ. Attempting reusable subtask state abstraction with Options, HAMQ or ALisp is similarly problematical.

This is a serious issue for HRL and affects the large class of infinite horizon MDPs. An infinite horizon MDP is one that may continue forever and never terminate, for example, pole-balancing. It is an issue because any infinite horizon MDP whose value function is based on the infinite sum of future rewards must discount them to ensure the value function remains bounded.

The main contribution of this paper is the formulation of a set of decomposition equations that allow both discounting and state abstraction. The method combines both a value and action discount function recursively to decompose a discounted value function. MAXQ uses two value functions, one that stores the real values and a pseudo value function that is used to ensure that subtasks terminate in goal termination states. The introduction of a third decomposed function will increase the storage requirements, but only by a factor of 1.5. The scaling potential of HRL is retained and HRL can be extended to include infinite horizon problems.

The rest of this paper is organised as follows. The next substantial section will derive the system of state abstracted decomposition equations for a discounted value function over a multi-level task hierarchy. We then introduce an infinite-horizon variant of the taxi task to illustrate the benefit of the foregoing decomposition. Results show that even for this simple task learning is faster and the value function can converge to the same value found by a 'flat' learner.

2 State Abstraction and Discounting

The objective in this section is to derive a system of equations to decompose and compact a value function representing the sum of *discounted* future rewards. In the process decomposition constraints will be stated that provide a method to design task hierarchies that are guaranteed to enable state abstraction of subtasks. The decomposition was inspired by Dietterich [4] but differs in the way termination rewards are handled. Importantly it extends the decomposition with an additional decomposed discount function.

Given a well defined reinforcement learning problem, m, with a finite set of states and actions, the value function of discounted future rewards for state s can be written as:

$$V_m^\pi(s) = \mathbf{E}\{r_1 + \gamma r_2 + \gamma^2 r_3 \ldots\} \tag{1}$$

where $\mathbf{E}$ is the expectation operator, π a stationary policy, r_n the primitive rewards received at each time step n and γ the discount factor.

If a subtask, $m-1$ is invoked in state s and takes a random number of time steps, N, to terminate, the above value function can be written as the expected sum of rewards accumulated inside the subtask plus the expected sum of rewards following termination of the subtask.

$$V_m^\pi(s) = \mathbf{E}\{\sum_{n=1}^{N-1} \gamma^{n-1} r_n\} + \mathbf{E}\{\sum_{n=N}^{\infty} \gamma^{n-1} r_n\} \tag{2}$$

The first series is just the local discounted value function for subtask $m-1$ where the termination is defined by a zero reward exit to an absorbing state. Isolating the primitive reward on termination in the second series we have

$$V_m^\pi(s) = V_{m-1}^\pi(s) + \mathbf{E}\{\gamma^{N-1} r_N + \sum_{n=N+1}^{\infty} \gamma^{n-1} r_n\} \tag{3}$$

If s' is the state reached after terminating subtask $m-1$, R the expected primitive reward on termination to s' after N steps and defining $P_m^\pi(s', N|s, \pi(s))$ as the joint probability of reaching state s' in N steps starting in state s and following policy π, then equation 3 becomes

$$V_m^\pi(s) = V_{m-1}^\pi(s) + \sum_{s',N} P^\pi(s', N|s, \pi(s))\gamma^{N-1}[R + \gamma V_m^\pi(s')] \tag{4}$$

It is desirable to design the subtasks in the task hierarchy in such a way that the state reached on subtask termination is independent of the number of steps to termination. This can be achieved if all the subtasks can be defined to terminate from just one of their states for some policy. In other words, if it is possible to find a policy for each subtask that can reach the termination state with probability one. This type of decomposition will be referred to as *subtask termination independent*. We will assume from here on, that the task hierarchy has been designed to comply with the subtask termination independence condition. This condition is related to MAXQ "funnel actions" that permit subtask abstraction [4] and is a feature of HEXQ [8].

While it may appear restrictive to terminate each subtask from just one state, it must be remembered that it is of course possible to define multiple tasks to terminate in a region of state space in all the possible ways allowed in the underlying MDP. The restriction is not on the number of exits, but rather that each termination can be reliably reached and executed from within each subtask. The final transition out of a subtask is always Markov and stationary as the overall problem is assumed to be a well defined reinforcement learning problem. The next state s' reached after termination of a subtask is only dependent on the final state inside the subtask not on any other state in the subtask. This is the key property that makes it possible to safely abstract subtasks. Independence means that

$$P^\pi(s', N|s, a) = P^\pi(s'|s, a)P^\pi(N|s, a) \tag{5}$$

Equation 4, using abbreviation $a = \pi(s)$ becomes

$$V^\pi_m(s) = V^\pi_{m-1}(s) + \sum_N P^\pi(N|s, a)\gamma^{N-1} \times \sum_{s'} P^\pi(s'|s, a)[R + \gamma V^\pi_m(s')] \tag{6}$$

The probability of the next state s' is now independent of any particular state s in the subtask for policy π. If g represents the set of states in the subtask then $P^\pi(s'|s, a) = P^\pi(s'|g, a)$ for all $s \in g$. Importantly, the final sum in equation 6 is therefore independent of the state s and the number of steps N to termination from s.

We define the *completion value function*, E, as the expected value of future rewards after termination of the subtask associated with abstract action a and define the *discount function*, D, as the expected discount to be applied to the completion value. That is:

$$E^\pi_m(g, a) = \sum_{s'} P^\pi(s'|g, a)[R + \gamma V^\pi_m(s')] \tag{7}$$

and

$$D^\pi_{m-1}(s) = \sum_N P^\pi(N|s, \pi(s))\gamma^{N-1} \tag{8}$$

Equation 6 can be succinctly written as

$$V^\pi_m(s) = V^\pi_{m-1}(s) + D^\pi_{m-1}(s)E^\pi_m(g, a) \tag{9}$$

The discount function D is the *multi-time model* for the termination of the subtask interpreted as an Option [3]. It can itself be recursively represented as follows:

$$D^\pi_{m-1}(s) = \gamma D^\pi_{m-2}(s)\Gamma^\pi_{m-1}(g, a) \tag{10}$$

where the *completion discount function*, $\mathit{\Gamma}$, for all states $s \in g$ is the expected value of discount that is applied to state s' reached after terminating the subtask associated with abstract action a and following the policy π thereafter. $\mathit{\Gamma}$ is constructed as a discounted value function with the termination value of subtask $m-1$ set to 1 and all internal rewards set to zero.

$$\mathit{\Gamma}^{\pi}_{m-1}(g,a) = \sum_{s'} P^{\pi}(s'|g,a) D^{\pi}_{m-1}(s') \tag{11}$$

Equations 7, 9, 10 and 11 are the recursive decompositions equations for a discounted value function following policy π for a task hierarchy. At the leaf nodes in the task hierarchy there are no child subtasks. For leaf subtasks, child subtask values, V^{π}_{m-1}, in the above equations are defined to be zero and discount values, D^{π}_{m-2}, set to 1. The recursively optimal value and discount functions, where abstract action a invokes subtask $m-1$ are:

$$\begin{aligned} V^*_m(s) &= \max_a [V^*_{m-1}(s) + D^*_{m-1}(s).E^*_m(g,a)] \\ D^*_{m-1}(s) &= \gamma \max_a [D^*_{m-2}(s).\mathit{\Gamma}^*_{m-1}(g,a)] \end{aligned} \tag{12}$$

This formulation requires only one action-value to be stored for each of the functions E and $\mathit{\Gamma}$ for states in a reusable subtask. In other words safe abstraction of the subtask states can be retained as for the non-discounted case.

To illustrate, figure 2 shows the decomposed value and discount functions, and the compaction achieved for the simple maze task hierarchy in figure 1. The discount factor γ is 0.9. The reward is -1 per step and 20 for problem termination. Part (a) illustrates the optimal value function for the subtask of leaving a room to the East. This corresponds to $V^*_{m-1}(s)$ in equation 12. Part (b) shows the on-policy discount function for the same task, i.e. $D^*_{m-1}(s)$ in equation 12. Part (c) shows the exit value $E^*_m(g,a)$ for each room for each of the room leaving abstract actions. To compose the optimal value of location A in figure 1 using equation 12 we find that leaving the room via the East gives the maximum value. Substituting the position A values from figure 2 (shaded) in $V^*_m(s) = V^*_{m-1}(s) + D^*_{m-1}(s).E^*_m(g,a)$ evaluates to $-1.9 + 0.81 \times -6.35 = -7.05$. It can be verified that -7.05 is the discounted value of location A in the "flat" problem. The sum of the 22 discounted rewards to termination is $-1 - .9 - .9^2 - .9^3 \ldots - .9^{21} + 20 \times .9^{22} = -7.05$.

The completion discount function and the action-value function tables for each subtask have different purposes. The former stores discount values by which to discount subtask completion values. The latter stores local discounted rewards accumulated inside a subtask. Together they allow an overall discounted value function to be losslessly compacted. The values within a subtask need to be stored only once and can be reused for each occurrence of the subtask. Subtask themselves can be represented by an aggregate state at a higher level as the completion value associated with a specific instance of the subtask is now adjusted by the action discount function.

In this way the value of any state of the original problem is exactly and compactly represented for any policy that can be implemented over the task

(a)

-4.69	-4.10	-3.44	-2.71	-1.90
-4.10	-3.44	-2.71	-1.90	-1.00
-3.44	-2.71	-1.90	-1.00	0.00
-4.10	-3.44	-2.71	-1.90	-1.00
-4.69	-4.10	-3.44	-2.71	-1.90

(b)

0.53	0.59	0.66	0.73	0.81
0.59	0.66	0.73	0.81	0.90
0.66	0.73	0.81	0.90	1.00
0.59	0.66	0.73	0.81	0.90
0.53	0.59	0.66	0.73	0.81

(c)

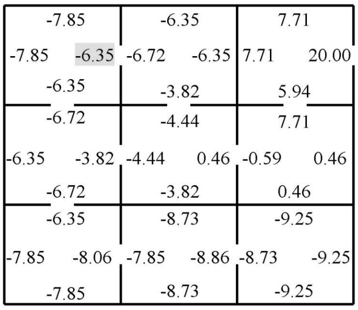


Fig. 2. The decomposed discounted value function for the simple maze in figure [1]. (a) shows the optimal value function and (b) the on-policy discount function for the room leaving subtask to the East. (c) shows the completion values, E, for each room and each of the room leaving abstract actions, e.g. E(top-right-room,leave-to-east) = 20.

hierarchy. This means that reusable subtask abstraction and hence scaling of RL problems can be achieved in the face of discounting. Importantly, subtask abstraction can now be used for infinite horizon problems for which discounting the sum of future rewards is mandatory. We now show how infinite horizon problems can be tackled using HRL.

3 Hierarchical Decomposition of Infinite Horizon MDPs

To allow a recursively optimal[1] solution to be found for an infinite horizon MDP task hierarchy we need to guarantee termination of subtask MDPs in the presence of positive and negative rewards. Even when all the rewards are negative an optimal policy does not guarantee that the subtasks will terminate when using a discounted value function. A barrier of high negative rewards may dissuade an agent from reaching a termination state, particularly for higher discount factors.

We therefore define a pseudo value function, $\overline{E}$, that sets a large enough positive termination value for the subtask so that the subtask will terminate starting in any of its states[2].

In total, therefore, there are three decomposed value functions that are required for each subtask. The pseudo reward completion value function, $\overline{E}$, that determines the policies available as abstract actions at the next level up in the hierarchy, the function, Γ, which holds discount values to be applied to the real completion value function at the next level up, and the real completion value function, E, which holds (in Ditterich's words) the "uncontaminated" exit values for each subtask. $\overline{E}$ is updated on-line and Γ and E are simultaneously updated following $\overline{E}$'s policy.

[1] A recursively optimal policy is one that finds the optimal policy for a (reusable) subtask without reference to its context.

[2] In contrast, MAXQ uses a pseudo value function that sets large negative values on undesirable terminations.

Discounting and positive rewards mean that an optimal policy may now require that the learner never leaves a subtask! For infinite horizon problems each region of state space associated with a subtasks will therefore need a separate subtask to be defined that does not terminate. A non-terminating subtask will not return control to its parent and a means of interrupting the execution of the subtask is required. To accommodate this non-terminating subtasks during learning, a subtask timeout is required. Currently we count the number of steps that a subtask executes and if it exceeds a manually set threshold value the execution is interrupted and control is passed back up the task hierarchy to the top level without updating value functions along the way.

MAXQ, HEXQ, ALisp and other HRL programs that use decomposed value functions can be augmented with the dual discounted value function decomposition equations presented in this paper to allow safe state abstraction of subtasks when subtasks are termination independent. HRL can now be applied to infinite horizon problems that can be decomposed by an appropriate task hierarchy.

4 Results

We will use a modified version of Dietterich's familiar taxi problem to demonstrate the decomposition of a discounted value function. For our implementation we chose the HRL algorithm HEXQ [8] and modified it to: (1) include the new completion discount function to decompose a discounted value function and (2) automatically generate one non-terminating subtask per region.

The taxi task is shown in figure 3. A taxi trans-navigates a grid world to pick up and drop off a passenger at four possible locations, designated R, G, Y and B. Actions move the taxi one square to the north, south, east or west, pickup or putdown the passenger. Navigation actions have a 20% chance of slipping to either the right or left of the intended direction. The default cost for each action is -1. If the taxi executes a pickup action at a location without the passenger or a putdown action at the wrong destination it receives a reward of -10.

To make the task continuing, a passenger location and destination are created at random after the passenger is delivered and the taxi is teleported to a random grid location. The reward for delivering the passenger is 200. The taxi domain is augmented with another source of income for the taxi. If the taxi transitions

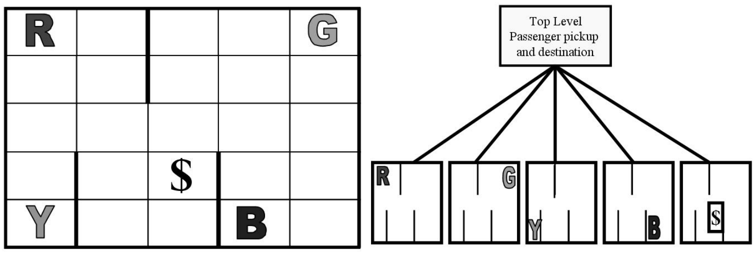


Fig. 3. The taxi task

from below to the grid location marked with the $ sign in figure 3, the reward is a positive number. This may be interpreted as a local pizza delivery task with guaranteed work but at a different rate of pay.

The taxi problem is modelled as an infinite horizon MDP with a two dimensional state vector $s =$ (taxi location, passenger source/destination). There are 25 possible taxi locations (grid points) and 20 possible pickup-destination combinations (5 passenger locations, including in the taxi $\times$ 4 drop-off locations).

This problem has two type of optimal solutions depending on the value of the $ reward. If the $ reward is low, the taxi will maximise its income by continuing to pick up and deliver passengers as per the original taxi problem. For larger $ values the taxi will prefer to make local delivery runs and visit the $ location as frequently as possible, ignoring the passenger.

The problem is hierarchically decomposed into a two level hierarchy as shown in figure 3. The top level task represents a semi-MDP where the abstract states describe the source and destination of the passenger. The bottom level subtasks are the four navigation MDPs that allow the taxi to navigate to one of the four pickup or drop-off locations and either pickup or drop-off the passenger. A fifth non-terminating subtask is now also generated as an available option.

As the infinite horizon taxi problem is solvable as a "flat" problem, it is possible to confirm that that the decomposition method developed in this paper does indeed converge to the "flat" value function. We test this by observing the convergence of both the "flat" and hierarchically decomposed value function for an arbitrary state when rewards at the $ location are varied.

Both the flat and HRL use a discount rate of 0.9, a learning rate of 0.1 and an ϵ-greedy exploration strategy with ϵ set to 0.8. We arbitrarily track the value (discounted sum of future reward) of the state where the taxi is at location "R", the passenger is at "Y" with destination "B". Figure 4 (a) shows the convergence of the value for this state when the reward at location $ $= -1$ meaning that there is no special advantage for visiting this location repeatedly. The three solid lines

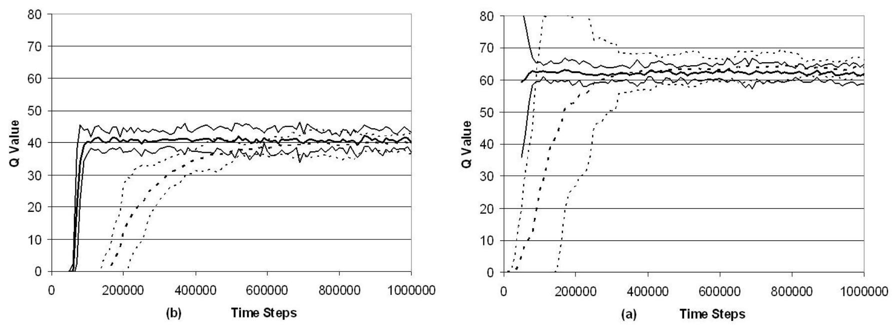


Fig. 4. (a) Tracking the convergence for an arbitrary Q value with $ $= -1$. The three solid lines show the mean and the 95% confidence interval for HEXQ. The dashed lines show the same for a "flat" learner. (b) Convergence results as for (a) but with $ $= 40$.

show the convergence with HEXQ. They show the mean over 10 runs together with its 95% confidence interval. The dashed lines show the same information for the "flat" learner. The overlapping confidence intervals indicate that there is no significant difference between the value functions learnt by the two learners. HEXQ converges faster even thought it first must uncover and construct the task hierarchy.

Figure 4 (b) shows a similar result when the $ reward is set to 40. At this value it is more profitable to remain in the non-terminating subtask and for the taxi to continue to deliver pizzas locally rather than pick up and deliver passengers.

5 Conclusion

The value function decomposition technique introduced in this paper has been applied to other infinite horizon problems involving more than two levels of hierarchy and for which the "flat" problem is intractable. Some of these problems were able to be solved in seconds with orders of magnitude of savings in storage for the value function. Space limitations do not permit a more detailed exposition.

The main contribution of this paper has been to show how a system of value function decomposition equations can resolve the issue of reusable subtask abstraction in the face of discounting. This now allows the benefits of subtask abstraction in HRL to be applied to infinite horizon Markov decision problems.

Acknowledgments. National ICT Australia is funded by the Australian Government's Backing Australia's Ability initiative, in part through the Australian Research Council.

References

1. Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introduction. MIT Press, Cambridge (1998)
2. Parr, R.E.: Hierarchical Control and learning for Markov decision processes. PhD thesis, University of California at Berkeley (1998)
3. Precup, D.: Temporal Abstraction in Reinforcement Learning. PhD thesis, Univeristy of Massachusetts, Amherst (2000)
4. Dietterich, T.G.: Hierarchical reinforcement learning with the MAXQ value function decomposition. Journal of Artificial Intelligence Research 13, 227–303 (2000)
5. Dean, T., Givan, R.: Model minimization in markov decision processes. In: AAAI / IAAI, pp. 106–111 (1997)
6. Dayan, P., Hinton, G.E.: Feudal reinforcement learning. In: NIPS. Advances in Neural Information Processing Systems, vol. 5 (1992)
7. Kaelbling, L.P.: Hierarchical learning in stochastic domains: Preliminary results. In: Machine Learning Proceedings of the Tenth International Conference, pp. 167–173. Morgan Kaufmann, San Francisco (1993)
8. Hengst, B.: Discovering Hierarchy in Reinforcement Learning. PhD thesis, School of Computer Science and Engineering, University of New South Wales, Sydney, Australia (2003)

Local Learning Algorithm for Markov Blanket Discovery

Shunkai Fu[*] and Michel Desmarais

Ecole Polytechnique de Montreal,
C.P.6079, Succ. Centre-ville, Montreal, Quebec, Canada
{shukai.fu, michel.desmarais}@polymtl.ca

Abstract. Learning of Markov blanket can be regarded as an optimal solution to the feature selection problem. In this paper, we propose a local learning algorithm, called Breadth-First search of MB (BFMB), to induce Markov blanket (MB) without having to learn a Bayesian network first. It is demonstrated as (1) easy to understand and prove to be sound in theory; (2) data efficient by making full use of the knowledge of underlying topology of MB; (3) fast by relying on fewer data passes and conditional independent test than other approaches; (4) scalable to thousands of variables due local learning. Empirical results on BFMB, along with known Iterative Association Markov blanket (IAMB) and Parents and Children based Markov boundary (PCMB), show that (i) BFMB significantly outperforms IAMB in measures of data efficiency and accuracy of discovery given the same amount of instances available (ii) BFMB inherits all the merits of PCMB, but reaches higher accuracy level using only around 20% and 60% of the number of data passes and conditional tests, respectively, used by PCMB.

Keywords: Markov blanket, local learning, feature selection.

1 Introduction

Classification is a fundamental task in data mining that requires learning a classifier from a data sample. Basically, a classifier is a function that maps instances described by a set of attributes to a class label. How to identify the minimal, or close to minimal, subset of variables that best predicts the target variable of interest is known as feature (or variable) subset selection (FSS). In the past three decades, FSS for classification has been given considerable attention, and it is even more critical today in many applications, like biomedicine, where high dimensionality but few observations are challenging traditional FSS algorithms.

A principle solution to the feature selection problem is to determine a subset of attributes that can render the rest of all attributes independent of the variable of interest [8,9,16]. Koller and Sahami (KS)[9] first recognized that the Markov blanket (see its definition below) of a given target attribute is the theoretically optimal set of attributes to predict the target's value, though the Markov blanket itself is not a new concept that can be traced back to 1988[11].

[*] This work was done during the author's time in SPSS.

A Markov blanket of a target attribute T renders it statistically independent from all the remaining attributes, that is, given the values of the attributes in the Markov blanket, the probability distribution of T is completely determined and knowledge of any other variable(s) becomes superfluous [11].

Definition 1 (Conditional independent). Variable X and T are conditionally independent given the set of variables $\boldsymbol{Z}$ (bold symbol is used for set), iff. $P(T \mid X, \boldsymbol{Z}) = P(T \mid \boldsymbol{Z})$, denoted as $T \perp X \mid \boldsymbol{Z}$.

Similarly, $T \not\perp X \mid \boldsymbol{Z}$ is used to denote that X and T are **NOT** conditionally independent given $\boldsymbol{Z}$.

Definition 2 (Markov blanket, $\boldsymbol{MB}$). Given all attributes $\boldsymbol{U}$ of a problem domain, a Markov blanket of an attribute $T \in \boldsymbol{U}$ is any subset $\boldsymbol{MB} \subseteq \boldsymbol{U} \backslash \{T\}$ for which

$$\forall X \in \boldsymbol{U} \setminus \{T\} \setminus \boldsymbol{MB},\ T \perp X \mid \boldsymbol{MB}$$

A set is called **Markov boundary** of T if it is a minimal Markov blanket of T.

Definition 3 (Faithfulness). A Bayesian network G and a joint distribution P are faithful to one another, if and only if every conditional independence encoded by the graph of G is also present in P, i.e., $T \perp_G X \mid \boldsymbol{Z} \Leftrightarrow T \perp_P X \mid \boldsymbol{Z}$ [12].

Pearl [11] points out that if the probability distribution over $\boldsymbol{U}$ can be faithfully represented by a Bayesian network (BN), which is one kind of graphical model that compactly represent a joint probability distribution among $\boldsymbol{U}$ using a directed acyclic graph, then the Markov blanket of an attribute T is unique, composing of the T's parents, children and spouses (sharing common children with T). So, given the faithfulness assumption, learning an attribute's Markov blanket actually corresponds to the discovery of its Markov boundary, and therefore can be viewed as selecting the optimal minimum set of feature to predict a given T. In the remaining text, unless explicitly mentioned, Markov blanket of T will refer to its Markov boundary under the faithfulness assumption, and it is denoted as $\boldsymbol{MB}(T)$.

$\boldsymbol{MB}(T)$ can be easily obtained if we can learn a BN over the $\boldsymbol{U}$ first, but the BN's structure learning is known as NP-complete, and readily becomes non-tractable in large scale applications where thousands of attributes are involved. Until now, none of existing known BN learning algorithms claims to scale correctly over more than a few hundred variables. For example, the publicly available versions of the PC [12] and the TPDA (also known as PowerConstructor)[2] algorithms accept datasets with only 100 and 255 variables respectively.

The goal of this paper is to develop an efficient algorithm for the discovery of Markov blanket from data without having to learn a BN first.

2 Related Work

A reasonable compromise to learning the full BN is to discover only the local structure around an attribute T of interest. We refer to the conventional BN learning as *global learning* and the latter as *local learning*. Local learning of $\boldsymbol{MB}(T)$ is expected to remain a viable solution in domains with thousands of attributes.

Local learning of $\boldsymbol{MB}$ began to attract attention after the work of KS [9]. However, the KS algorithm is heuristic, and provides no theoretical guarantee of success. Grow-Shrink (GS) algorithm [10] is the first provably correct one, and, as indicated by its name, it contains two sequential phases, growing first and shrinking secondly. To improve the speed and reliability, several variants of GS, like IAMB, InterIAMB [15,16] and Fast-IAMB[17], were proposed. They are proved correct given the faithfulness assumption, and indeed make the $\boldsymbol{MB}$ discovery more time efficient, but none of them are data efficient. In practice, to ensure reliable independence test tests, which is the basis for this family of algorithm, IAMB and its variants decide a test is reliable when the number of instances available is at least five times the number of degree of freedom in the test. This means that the number of instances required by IAMB to identify $\boldsymbol{MB}(T)$ is at least exponential in the size of $\boldsymbol{MB}(T)$, because the number of degrees of freedom in a test is exponential in the size of conditioning set and the test to add a new node in $\boldsymbol{MB}(T)$ will be conditioned on at least the current nodes in $\boldsymbol{MB}(T)$ (Line 4, Table 1) [8].

Several trials were made to overcome this limitation, including MMPC/MB[14], HITON-PC/MB[1] and PCMB[8]. All of them have the same two assumptions as IAMB, i.e. faithfulness and correct independence test, but they differ from IAMB by taking into account the graph topology, which helps to improve data efficiency through conditioning over a smaller set instead of the whole $\boldsymbol{MB}(T)$ as done by IAMB. However, MMPC/MB and HITON-PC/MB are shown not always correct by the authors of PCMB since false positives will be wrongly learned due to their inner defect [8]. So, based on our knowledge, PCMB is the only one proved correct, scalable and truly data-efficient means to induce the MB when this paper is prepared.

In this paper, we propose a novel MB local learning algorithm, called Breadth First search of Markov Blanket (BFMB). It is built on the same two assumptions of IAMB and PCMB. BFMB algorithm is compared with two of the algorithms discussed above: IAMB and PCMB. IAMB is a well known algorithm and referred to as MB local discovery. PCMB is the most successful break over IAMB to our knowledge and our own work is based on this algorithm. To allow for convenient reference and comparison, we include the complete IAMB and partial PCMB algorithms here in Table 1.

Akin to PCMB, BFMB is designed to execute an efficient search by taking the topology into account to ensure a data efficient algorithm. We believe this approach is an effective means to conquer the data inefficiency problem occurring in GS, IABM and their variants. As its name implies, BFMB starts the search of $\boldsymbol{MB}(T)$ from its neighbors first, which actually are the parents and children of T , denoted as $\boldsymbol{PC}(T)$. Then, given each $X \in \boldsymbol{PC}(T)$, it further searches for $\boldsymbol{PC}(X)$ and checks each $Y \in \boldsymbol{PC}(X)$ to determine if it is the spouse of T or not. So, our algorithm is quite similar to PCMB, but it finds the $\boldsymbol{PC}$ of an attribute in a much more efficient manner. More detail about the algorithm can be found in Section 3. Considering that the discovery of $\boldsymbol{PC}(X)$ is a common basic operation for PCMB and BFMB, its efficiency will directly influence the overall performance of algorithm. Experiment results of algorithms comparison are reported and discussed in Section 4.

Table 1. IAMB and partial PCMB algorithms

IAMB

```
IAMB(
   $D$ : Dataset,
   $\varepsilon$ :threshold
)
{
   // Grow phase.
1  $U = \{\textbf{attributes} \text{ in D}\}$ ;
2  do
3      $X_i \in \arg\min_{X_i \in U \setminus MB} I_D(T, X_i \mid \mathbf{MB})$ ;
4      if ( $I(T, X_i \mid \mathbf{MB}) < \varepsilon$ ) then
5          $\mathbf{MB} = \mathbf{MB} \cup X_i$
6  while( MB has change )
   // Shrink phase
7  for(each $X_i \in$ MB ) do
8    if ( $I_D(T, X_i \mid \mathbf{MB} \setminus \{X_i\}) < \varepsilon$ ) then
9        $\mathbf{MB} = \mathbf{MB} \setminus \{X_i\}$ ;
10 return MB ;
}
```

PCMB

```
GetPCD( $T$ )
{
1  $PCD = \phi$ ;
2  $CanPCD = U \setminus \{T\}$ ;
3  do
   /*remove false positives*/
4   for(each $X_i \in CanPCD$ ) do
5      $Sep[X_i] = \arg\min_{Z \subseteq PCD} I_D(T, X_i \mid \mathbf{Z})$
```

PCMB (cont.)

```
6   for(each $X_i \in CanPCD$ )do
7      if( $T \perp X_i \mid Sep[X_i]$ ) then
8        $CanPCD = CanPCD \setminus \{X_i\}$ ;
     /*add the best candidate */
9
$Y = \arg\min_{X \in CanPCD} I_D(T, X_i \mid Sep[X])$
10  $PCD = PCD \cup \{Y\}$ ;
11  $CanPCD = CanPCD \setminus \{Y\}$ ;
12  for( each $X_i \in PCD$ ) do
13
$Sep[X_i] = \arg\min_{Z \subseteq PCD \setminus \{X_i\}} I_D(T, X_i \mid \mathbf{Z})$ ;
14   for(each $X_i \in PCD$ )do
15      if ( $I_D(T, X_i \mid Sep[X_i]) < \varepsilon$ ) then
16         $PCD = PCD \setminus \{X_i\}$ ;
17 while( $PCD$ has change &&
              $CanPCD \neq \phi$ )
18 return $PCD$ ;
}

GetPC( $T$ )
{
1  $PC = \phi$ ;
2  for(each $X_i \in$ GetPCD( $T$ )) do
3      if ( $T \in$ GetPCD( $X_i$ )) then
4          $PC = PC \cup \{X_i\}$
}
```

3 Local Learning Algorithm of Markov Blanket: BFMB

3.1 Overall Design

As discussed in Section 1 and 2, the BFMB algorithm is based on two assumptions, faithfulness and correct conditional test, based on which the introduction and proof of this algorithm will be given.

Table 2. BFMB algorithm

```
RecognizePC (
  $T$ : target,
  $ADJ_T$ :Adjacency set to search
  D : Dataset, $\varepsilon$ :threshold
)
{
1  $NonPC = \phi$ ;
2  cutSetSize = 1;
3  do
4    for(each $X_i \in ADJ_T$ )do
5      for(each $S \subseteq ADJ_T \setminus \{X_i\}$ with $|S|$ = cutSetSize)do
6        if( $I_D(X_i, T \mid S) \le \varepsilon$ )then
7          $NonPC = NonPC \cup \{X_i\}$ ;
8          break;
9    if( $|NonPC| > 0$ )then
10     $ADJ_T = ADJ_T \setminus NonPC$ ;
11     $Sepset_{T,X_i} = S$ ;
12     cutSetSize +=1;
13     $NonPC = \phi$ ;
14   else
15     break;
16 while( $|ADJ_T| >$ cutSetSize)
17 return $ADJ_T$ ;
}

RecognizeMB(
  $D$ : Dataset,
  $\varepsilon$ :threshold
)
{
  // Recognize T'parents/children
1  $CanADJ_T = U \setminus \{T\}$ ;
2  $PC$ =RecognizePC( $T$ , $CanADJ_T$ , $D$ , $\varepsilon$ );
3  $MB = PC$ ;
  // Recognize the spouses of T.
4  for(each $X_i \in PC$ )do
5    $CanADJ_{X_i} = U \setminus \{X_i\}$ ;
6    $CanSP$ = RecognizePC( $X_i$ , $CanADJ_{X_i}$ , $D$ , $\varepsilon$ );
7    for(each $Y_i \in CanSP$ and $Y_i \notin MB$ ) do
8      if( $I_D(T, Y_i \mid Sepset_{T,Y_i} \cup X_i) > \varepsilon$ )then
9        $MB = MB \cup \{Y_i\}$ ;
10 return $MB$ ;
}
```

On a BN over U , the $MB(T)$ contains parents and children of T , i.e. those nodes directly connected to T , and its spouses, i.e. parents of T 's children. We denote these two sets as $PC(T)$ and $SP(T)$ respectively. With considerations in mind, learning $MB(T)$ amounts to deciding which nodes are directly connected to T and which directly connect to those nodes adjacent to T (connect to T with an arc by ignoring the orientation).

Given a Bayesian network, it is trivial to extract a specific $MB(T)$ given attribute T . However, learning the topology of a Bayesian network involves a global search that can prove intractable. Actually, we can avoid obstacle by following what we discussed above on the underlying topology information. We need only decide (1) which are adjacent to T among $U \setminus \{T\}$, i.e. $PC(T)$ here, and (2) which are adjacent to $PC(T)$ and point to children of T in the remaining attributes $U \setminus \{T\} \setminus PC$, i.e. $SP(T)$. Since it is actually a breadth-first search procedure, we name our algorithm as BFMB.

We need not care about the relations among $\boldsymbol{PC}(T)$, $\boldsymbol{SP}(T)$ and between $\boldsymbol{PC}(T)$ and $\boldsymbol{SP}(T)$, considering that we are only interested in which attributes belong to $\boldsymbol{MB}(T)$. Therefore, this strategy will allow us to learn $\boldsymbol{MB}(T)$ solely through local learning, reducing the search space greatly.

3.2 Theoretical Basis

In this section, we provide theoretical background for the correctness of our algorithm.

Theorem 1. If a Bayesian network G is faithful to a probability distribution P, then for each pair of nodes X and Y in G, X and Y are adjacent in G iff. $X \not\perp Y \mid \mathbf{Z}$ for all $\mathbf{Z}$ such that X and $Y \notin \mathbf{Z}$. [12]

Lemma 1. If a Bayesian network G is faithful to a probability distribution P, then for each pair of nodes X and Y in G, if there exists $\mathbf{Z}$ such that X and $Y \notin \mathbf{Z}$, $X \perp Y \mid \mathbf{Z}$, then X and Y are **NOT** adjacent in G.

We get **Lemma 1** from **Theorem 1**, and its proof is trivial. The first phase of BFMB, *RecognizePC* (Table 2), relies upon this basis. In fact, the classical structure learning algorithm PC [12, 13] is the first one designed on this basis.

Theorem 2. If a Bayesian network G is faithful to a probability distribution P, then for each triplet of nodes X, Y and Z in G such that X and Y are adjacent to Z, but X and Y are not adjacent, $X \rightarrow Z \leftarrow Y$ is a subgraph of G iff $X \not\perp Y \mid \mathbf{Z}$ for all $\mathbf{Z}$ such that X and $Y \notin \mathbf{Z}$, and $Z \notin \mathbf{Z}$. [12]

Theorem 2 plus **Theorem 1** form the basis of BFMB's second phase, the discovery of T's spouses (Table 2). Given $X \in \boldsymbol{PC}(T)$, which is the output of phase 1 in BFMB, we can learn $\boldsymbol{PC}(X)$ as we learn $\boldsymbol{PC}(T)$. For each $Y \in \boldsymbol{PC}(X)$, if we known $T \not\perp Y \mid \mathbf{Z}$ for all $\mathbf{Z}$ such that T, $Y \notin \mathbf{Z}$ and $X \in \mathbf{Z}$, $T \rightarrow X \leftarrow Y$ is a subgraph of G; therefore Y is a parent of X; since X is the common child between Y and T, Y is known as one spouse of T. This inference brings us Lemma 2.

Lemma 2. In a Bayesian network G faithful to a probability distribution P, given $X \in \boldsymbol{PC}(T)$, and $Y \in \boldsymbol{PC}(X)$, if $T \not\perp Y \mid \mathbf{Z}$ for all $\mathbf{Z}$ such that T, $Y \notin \mathbf{Z}$ and $X \in \mathbf{Z}$, then Y is a spouse of T.

3.3 Breadth-First Search of Markov Blanket

Learn Parents/Children

The following table is the algorithm to find which variables should be joined by arcs to, i.e. dependent on, target T. We name it *RecognizePC*, and its output contains the complete and only set of parents and children of T .The soundness of *RecognizePC* is based on the DAG-faithfull and correct independence test assumptions.

RecognizePC procedure (Table 2) is quite similar to the conventional PC structure learning algorithm, but it limits the search to the neighbors of the target we are studying, which means that local, instead of global, learning is required by this MB learning algorithm.

Theorem 3. Each $X_i \in \boldsymbol{PC}(T)$ returned by *RecognizePC* is a parent or child of T , and $\boldsymbol{PC}(T)$ contains all the parents and children of T.

Proof.(i)For each $X_i \in \boldsymbol{PC}(T)$*, we scan each possible subset* $S \subseteq ADJ_T \setminus \{X_i\}$, and only those X_i *satisfying* $I_D(T, X_i \mid S) > \varepsilon$*, where* $I_D(\hbar)$ *is conditional independence test and* ε *is pre-defined threshold value,* can finally be included into $\boldsymbol{PC}$. *With* **Theorem 1***, we can infer that T and* X_i *should be adjacent. (ii) Since we start with* $ADJ_T = \boldsymbol{U} \setminus \{T\}$, and check *all* $X_i \in ADJ_T$ *, and given the correctness of statement (i), we cannot miss any* X_i *adjacent to* T *.*

The extraction of the orientation between T and $X_i \in \boldsymbol{PC}(T)$ is not the goal of this paper since we won't distinguish which ones are parents and which are children of T . With the example shown in Figure 1, *RecognizePC* correctly finds its $\boldsymbol{PC}(T) = \{2,4,6,7\}$.

Learn Spouses

Our search of $\boldsymbol{MB}(T)$ consists in finding parents/children first, and then the spouses. During implementation, we take the strategy of breadth first by looking for those neighbors of T in its first round, and secondly further check the neighbors of those variables found in its first step, enrolling them into $\boldsymbol{MB}(T)$ if they are found to share common children with T . The outcome of this second step, as shown soon, will be the remaining part of $\boldsymbol{MB}(T)$, i.e., its spouse nodes.

The following *RecognizeMB* is designed by this idea. It takes dataset as the input, and output $\boldsymbol{MB}(T)$.

Theorem 4. The result given by *RecognizeMB* is the complete Markov blanket of T .

Proof.(i) Based on Theorem 3, we know that $\boldsymbol{PC}(T)$ *will contain all and only the parents and children of* T *. (ii) Once again, with Theorem 3, we know that* CanSP *contains all and only the parents and children of each* $X_i \in$ PC *. (iii) During our decision of whether or not to enroll one* Y_i returned by *RecognizePC(* X_i) (line 6 in *RecognizeMB,* $X_i \in \boldsymbol{PC}(T)$), ***we need refer the*** $Sepset_{T,Y_i}$ ***(got when we call*** *RecognizePC(T))* ***conditioned on which*** *T* ***and*** Y_i ***are independent****. If* Y_i *and T are conditionally dependent given* $Sepset_{T,Y_i} \cup X_i$ *, that is* $I_D(T, Y_i \mid Sepset_{T,Y_i} \cup X_i) > \varepsilon$. Since both Y_i and T ***are adjacent to*** X_i***, and*** Y_i *is not adjacent to* T *, we can infer the existence of topology* $Y_i \rightarrow X \leftarrow Z$ *, based on Theorem*

2. (iv) *Because we look into each parent and child set of* $X_i \in \mathrm{PC}$*, our algorithm will not miss any of the spouses of* T *. Therefore, it is correct and complete.*

3.4 Expected Merits over IAMB and PCMB

BFMB algorithm is introduced and proved correct in the preceding sections. Before we go ahead to report the experimental results, we discuss the expected merits of BFMB over IAMB and PCMB, and confirm them with empirical results in the following section.

BFMB is more complex than IAMB since knowledge of the underlying topology is used here. It is designed with the divide-and-conquer idea as IAMB, but a different strategy is followed. IAMB is composed with growing and shrinking, which is sound as well as simple by ignoring the underlying graph topology. BFMB divides the learning into looking for parents/children first, then spouses. Besides, it always tests by conditioning on the minimum set. These two aspects enhance the data efficiency of BFMB greatly over IAMB, but without scarifying correctness and scalability.

BFMB's overall framework is quite similar to PCMB by taking account of the graph topology, but BFMB learns the $\boldsymbol{PC}(T)$ in a more efficient manner. In PCMB, before a new $\boldsymbol{PC}(T)$'s candidate is determined, it first needs to find a series of $Sep[X]$ (line 5 in GetPCD), which will cost one data pass to scan the data to collect necessary statistics required by the CI test in practice. After a best candidate is added to $\boldsymbol{PCD}$, PCMB needs another search for $Sep[X]$ (line 13 in GetPCD), which requires additional data pass. Noticing the GetPCD on line 3 in GetPC, we find that many data passes are required in the algorithm. Normally, in each data pass, we only collect those information obviously demanded in the current round. These data passes cannot be avoided by PCMB. By opposition, BFMB starts from the conditioning set with size 1, and all possible conditioning sets are expected at the beginning of each new round, so we need only one data pass to collect all statistics and remove as many variables as possible that are conditionally independent of T given someone conditioning set with the current *cutSetSize*(line 5-6 in *RecognizePC*). This approach also ensures that BFMB will find the minimum conditioning set, $Sepset_{T,X_i}$, during the first pass, without having to scan all possible conditioning sets to find the minimum one to ensure maximum data efficiency as does in PCMB. Considering that PCMB is expected to perform a greater number of data passes and CI tests, it will lose to BFMB in terms of time efficiency.

4 Experiment and Analysis

4.1 Experiment Design

We only compare our algorithm with IAMB and PCMB. In the experiment, we use synthetic data sampled from known Alarm BN[7] which is composed of 37 nodes. The Alarm network is well-known as it has been used in a large number of studies on probabilistic reasoning. The network modeling situations arise from the medicine world. We run IAMB, PCMB and BFMB with each node in the BN as the target variable T iteratively and, then, report the average performance when different size of data is given, including accuracy, data efficiency, time efficiency, scalability, and usefulness of information found.

4.2 Evaluation and Analysis

One of the basic assumptions of these three algorithms is that the independence tests are valid. To make them three, IAMB, PCMB and BFMB, feasible in practice, we perform a test to check if the conditional test to do is reliable, and skip the result if not, which can ensue the learning outcome trustable. As indicated in [15], IAMB considers a test to be reliable when the number of instances in D is at least five times the number of degrees of freedom in the test. PCMB follows this standard in [8], and so does our algorithm BFMB to maintain a comparable experiment result.

Accuracy and Data Efficiency

We measure the accuracy of induction through the precision and recall over all the nodes for the BN. **Precision** is the number of true positives in the returned output divided by the number of nodes in the output. **Recall** is the number of true positives in the output divided by the number of true positives known in the true BN model. We also combine precision and recall as

$$\sqrt{(1-precision)^2+(1-recall)^2}$$

to measure the Euclidean **distance** from precision and recall[8].

Table 3. Accuracy comparison of IAMB, PCMB and BFMB over Alarm network

Instances	Algorithm	Precision	Recall	Distance
1000	IAMB	.81±.03	.78±.01	.29±.03
1000	PCMB	.76±.04	.83±.07	.30±.06
1000	BFMB	.92±.03	.84±.03	.18±.04
2000	IAMB	.79±.03	.83±.02	.27±.03
2000	PCMB	.79±.04	.91±.04	.23±.05
2000	BFMB	.94±.02	.91±.03	.11±.02
5000	IAMB	.77±.03	.88±.00	.26±.02
5000	PCMB	.80±.05	.95±.01	.21±.04
5000	BFMB	.94±.03	.95±.01	.08±.02
10000	IAMB	.76±.03	.92±.00	.26±.03
10000	PCMB	.81±.03	.95±.01	.20±.03
10000	BFMB	.93±.02	.96±.00	.08±.02
20000	IAMB	.73±.04	.93±.00	.28±.04
20000	PCMB	.81±.02	.96±.00	.20±.01
20000	BFMB	.93±.03	.96±.00	.08±.02

Table 3 shows the average precision, recall and distance performance about IAMB, PCMB and BFMB given different size of data sampled from Alarm network. From which, as expected, we observe that IAMB has the poorest performance, which actually reflects its data inefficiency due to its possible large conditioning set. Indeed, we know that given the same about of data, requirement on larger conditioning set will result with less precise decision. PCMB is better than IAMB, which is consistent with the result shown in [8]. However, it is worse than BFMB, which can be explained by its search strategy of minimum conditioning set. It needs to go through conditioning sets with size ranging from small to large, so PCMB has the similar problem like IAMB when conditioned on large set. However, BFMB's strategy prevents it from this weakness, so it has the highest score on this measure.

Time Efficiency

To measure time efficiency, we refer to the number of data pass and CI test occuring in IAMB, PCMB and BFMB. One data pass corresponds to the scanning of the whole data for one time. To make memory usage efficient, we only collect all the related statistics information (consumed by CI tests) that can be expected currently. In Table 4, "# rounds" refers to the total number of data passes we need to finish the MB induction on all the 37 nodes of Alarm BN. "# CI test" is defined similarly. Generally, the larger are these two numbers, the slower is the algorithm.

Table 4. Comparison of time complexity required by different MB induction algorithms, in terms of number of data pass and CI test

Instances	Algorithms	# rounds	# CI test
5000	IAMB	211±5	5603±126
5000	PCMB	46702±6875	114295±28401
5000	BFMB	5558±169	57893±3037
10000	IAMB	222±4	6044±119
10000	PCMB	46891±3123	108622±13182
10000	BFMB	5625±121	62565±2038
20000	IAMB	238±10	6550±236
20000	PCMB	48173±2167	111100±9345
20000	BFMB	5707±71	66121±1655

As Table 4 shows, IAMB requires the fewest number of data pass and CI tests. Though it wins this point over the other two algorithms, its accuracy performance is quite poor (refer Table 3). PCMB and BFMB outperform IAMB in learning accuracy, but at a much higher cost of time; comparing with BFMB, PCMB is even worse. In this study, BFMB requires less than 20% and 60% of the total amount of data passes and CI tests done by PCMB respectively. Though some optimization techniques can probably be designed, it reflects the implementation complexity of this algorithm in practice.

Scalability

IAMB and its variants are proposed to do feature selection in microarray research [14, 15]. From our study, it is indeed a fast algorithm even when the number of features and number of cases become large. Reliable results are expected when there are enough data. PCMB is also shown scalable by its author in [8], where it is applied to a KDD-Cup'2001 competition problem with 139351 features. Due to the short of such large scale observation, we haven't tried BFMB in the similar scenario yet. However, our empirical study, though there are only 37 variables, have shown that BFMB runs faster than PCMB. Therefore, we have confidence to do this inference that BFMB can also scale to thousands of features as IAMB and PCMB claim. Besides, due to the relative advantage on data efficiency among the three algorithms, BFMB is supposed to work with best results in challenging applications where there is large number of features but small amount of samples.

Usefulness of Information Found

Markov blanket contains the target's parents, children and spouses. IAMB and its variants only recognize that variables of MB render the rest of variables on the BN independent of target, which can be a solution to the feature subset selection. Therefore, IAMB only discovers which variables should fall into the Markov blanket, without further distinguishing among spouse/parents/children.PCMB and BFMB goes further by discovering more topology knowledge. They not only learn MB, but also distinguish the parents/children from the spouses of target. Among parents/children, those children shared by found spouses and target are also separated (Fig. 1).

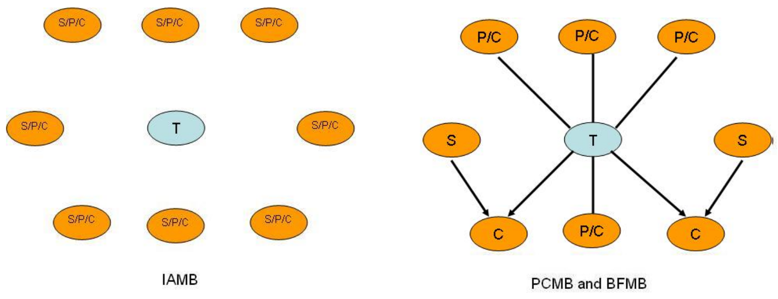


Fig. 1. Output of IAMB (left) vs. that of PCMB and BFMB(right)

5 Conclusion

In this paper, we propose a new Markov blanket discovery algorithm, called BFMB. It is based on two assumptions, DAG-faithful distribution and correct independence test. Like IAMB and PCMB, BFMB belongs to the family of local learning of MB, so it is scalable to applications with thousands of variables but few instances. It is proved correct, and much more data-efficient than IAMB, which allows it perform much better in learning accuracy than IAMB given the same amount of instances in practice. Compared with PCMB, BFMB provides a more efficient approach for learning, requiring much fewer number of CI tests and data passes than PCMB. Therefore, we can state that BFMB shows a high potential for practical MB discovery algorithm, and is a good tradeoff between IAMB and PCMB.

Future work includes the quantiative analysis of the algorithm's complexity, and how well it works with existing known classifiers as a feature selection pre-processing step.

References

1. Aliferis, C.F., Tsamardinos, I., Statnikov, A.: HITON, a Novel Markov blanket algorithm for optimal variable selection. In: Proceedings of the 2003 American Medical Informatics Association Annual Symposium, pp. 21–25 (2003)
2. Cheng, J., Greiner, R.: Learning Bayesian networks from data: An information-theory based approach. Artificial Intelligence 137, 43–90 (2002)
3. Cheng, J., Greiner, R.: Comparing Bayesian Network classifiers. In: Proceedings of the 15th Conference on UAI (1999)
4. Cheng, J., Bell, D.A., Liu, W.: Learning belief networks from data: An information theory based approach. In: Proceedings of the sixth ACM International Conference on Information and Knowledge Management (1997)
5. Cooper, G.F.: The computational complexity of probabilistic inference using Bayesian belief networks. Artificial Intelligence 42, 395–405 (1990)
6. Friedman, N., Geiger, D., Goldszmidt, M.: Bayesian network classifiers. Machine Learning 29, 131–163 (1997)
7. Herskovits, E.H.: Computer-based probabilistic-netork construction. Ph.D Thesis, Stanford University (1991)
8. Pena, J.M., Nilsson, R., Bjorkegren, J., Tegner, J.: Towards scalable and data efficient learning of Markov boundaries. International Journal of Approximate Reasoning 45(2), 211–232 (2007)
9. Koller, D., Sahami, M.: Toward optimal feature selection. In: Proceedings of International Conference on Machine Learning, pp. 284–292 (1996)
10. Margaritis, D., Thrun, S.: Bayesian network induction via local neighborhoods. In: Proceedings of NIPS (1999)
11. Pearl, J.: Probabilistic reasoning in intelligent systems: Networks of plausible inference. Morgan Kaufmann, San Francisco (1988)
12. Spirtes, P., Glymour, C., Scheines, R.: Causation, Prediction, and Search. Lecture Notes in Statistics. Springer, Heidelberg (1993)
13. Spirtes, P., Glymour, C.: An algorithm for Fast Recovery of Sparse Casual Graphs. Philosophy Methodology Logic (1990)
14. Tsamardinos, I., Aliferis, C.F., Statnikov, A.: Time and sample efficient discovery of Markov blankets and direct causal relations. In: Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 673–678 (2003)
15. Tsamardinos, I., Aliferis, C.F.: Towards principled feature selection: Relevancy, filter and wrappers. In: AI&Stats 2003. 9th International Workshop on Artificial Intelligence and Statistics (2003)
16. Tsamardinos, I., Aliferis, C.F., Stantnikov, A.: Time and sample efficient discovery of Markov blankets and direct causal relations. In: Proceedings of SIGKDD 2003 (2003)
17. Yaramakala, S., Margaritis, D.: Speculative Markov blanket discovery for optimal feature selection. In: ICDM 2005. Proceedings of IEEE International Conference on Data Mining (2005)

A Novel Ordering-Based Greedy Bayesian Network Learning Algorithm on Limited Data

Feng Liu[1], Fengzhan Tian[2], and Qiliang Zhu[1]

[1] Department of Computer Science, Beijing University of Posts and Telecommunications,
Xitu Cheng Lu 10, 100876 Beijing, China
lliufeng@hotmail.com,zhuqiliang@tom.com

[2] Department of Computer Science, Beijing Jiaotong University,
Shangyuan Cun 3, 100044 Beijing, China
fztian@mail.bjtu.edu.cn

Abstract. Existing algorithms for learning Bayesian network (BN) require a lot of computation on high dimensional itemsets, which affects accuracy especially on limited datasets and takes up a large amount of time. To alleviate the above problem, we propose a novel BN learning algorithm OMRMRG, Ordering-based Max Relevance and Min Redundancy Greedy algorithm. OMRMRG presents an ordering-based greedy search method with a greedy pruning procedure, applies Max-Relevance and Min-Redundancy feature selection method, and proposes Local Bayesian Increment function according to Bayesian Information Criterion (BIC) formula and the likelihood property of overfitting. Experimental results show that OMRMRG algorithm has much better efficiency and accuracy than most of existing BN learning algorithms on limited datasets.

1 Introduction

During the last two decades, many BN learning algorithms have been proposed. But, the recent explosion of high dimensional and limited datasets in the biomedical realm and other domains has induced a serious challenge to these BN learning algorithms. The existing algorithms must face higher dimensional and smaller datasets.

In general, BN learning algorithms take one of the two approaches: the constraint-based method and the search & score method. The constraint-based approach [1],[2] estimates from the data whether certain conditional independences hold between variables. The search & score approach [3],[4],[5],[8],[11] attempts to find a graph that maximizes the selected score. Score function is usually defined as a measure of fitness between the graph and the data.

Although encouraging results have been reported, the two approaches both suffer some difficulties in accuracy on limited datasets. A high order statistical or information theoretical measure may become unreliable on limited datasets. At the same time, the result of selected score function may also be unreliable on limited datasets.

To further enhance learning efficiency and accuracy, this paper proposes Ordering-based Max-Relevance and Min-Redundancy Greedy (OMRMRG) BN learning algorithm. This paper is organized as follows. Section 2 provides a brief review of Max-Relevance and Min-Redundancy feature selection technology. Section 3 introduces ordering-based greedy search method. Section 4 proposes Local Bayesian Increment function. Section 5 presents OMRMRG algorithm. Section 6 shows an experimental comparison among OMRMRG, OK2, OR and TPDA. Finally, we conclude our work.

2 Max-Dependence and MRMR

Definition 1. *In feature selection, **Max-Dependence scheme** [6] is to find a feature set S^* with m features, which jointly have the largest dependency on the target class C; $S^* = \arg\max_S I(S;C)$. Note: $I(S;C)$ represents the mutual information between the feature set S and the target class C which is a convenient way to measure the largest dependency between S and C,*

$$I(S;C) = \int\int p(s,c) \log \frac{p(s,c)}{p(s)p(c)} dsdc,$$

where $p(.)$ is the probabilistic density function.

Definition 2. *In feature selection, **Max-Relevance criterion** [6] is to select a feature set S^* with m features satisfying $S^* = \arg\max_S(\frac{1}{|S|}\sum_{X_i \in S} I(X_i;C))$, which approximates $I(S^*;C)$ with the mean value of all mutual information values between individual features $X_i, i = 1,\ldots,m$ and class C.*

Definition 3. *In feature selection, **Min-Redundancy criterion** [6] is to select a feature set S^* with m features such that they are mutually minimally similar (mutually maximally dissimilar): $S^* = \arg\min_S(\frac{1}{|S|^2}\sum_{X_i,X_j \in S} I(X_i;X_j))$.*

Definition 4. *In feature selection, **Max-Relevance and Min-Redundancy criterion** [6] is to find a feature set S^* with m features obtained by optimizing the Max-Relevance criterion and the Min-Redundancy criterion simultaneously. Assume that the two conditions are equally important, and use the following formula:*

$$S^* = \arg\max_S(\sum_{X_i \in S} I(X_i;C) - \frac{1}{\mid S \mid}\sum_{X_i,X_j \in S} I(X_i;X_j)). \tag{1}$$

We select the feature set $S_m = \{X_1,\ldots,X_m\}$, the classification variable C. Using the standard multivariate mutual information

$$MI(X_1,\ldots,X_m) = \int\int p(x_1,\ldots,x_m) \log \frac{p(x_1,\ldots,x_m)}{p(x_1)\ldots p(x_m)} dx_1 \ldots dx_n,$$

we can get the following formula:

$$I(S_m; C) = \int\int p(s_m, c) \log \frac{p(s_m, c)}{p(s_m)p(c)} ds_m dc = MI(S_m, C) - MI(S_m).$$

$$S^* = \arg\max_{S_m} I(S_m; C) = \arg\max_{S_m}(MI(S_m, C) - MI(S_m)). \tag{2}$$

Equation (2) is similar to MRMR feature selection criterion (1): The second term requires that the features S_m are maximally independent of each other(that is, minimum redundant), while the first term requires every feature to be maximally dependent on C. In practice, the authors Peng & Ding have shown that if one feature is selected at one time, then MRMR criterion is a nearly optimal implementation of Max-Dependence scheme on limited datasets. [7]

3 Ordering-Based Greedy Search Method

The search & score approach for BN learning is to perform heuristic search over some space. The ordering-based greedy search method proposed by Teyssier & Koller [8] conducts the search over the space of orderings $\prec$. The method defines the score of an ordering as the score of the best network consistent with it, defines a local search operator—flipping a pair of adjacent variables in the ordering—that traverses the space of orderings, and uses greedy hill-climbing search, with a tabu list and random restarts, to find an ordering that maximizes the score.

For a given ordering $\prec$, the candidate parents set $\Psi_{i,\prec}$ for variable X_i is defined $\Psi_{i,\prec} = \{\psi \mid \psi \prec X_i\}$. The optimal parents set is simply $Pa_{\prec}(X_i) = \arg\max_{U \subseteq \Psi_{i,\prec}} score(X_i; U)$.

So, given an ordering, we can find the optimal parents set for each variable and construct the optimal network $G^*_{\prec}$ consistent with $\prec$.

In the unconstrained case, the optimal network with no ordering constraint is the network $G^*_{\prec^*}$, for $\prec^* = \arg\max_{\prec} score(G^*_{\prec})$.

So, we can find the optimal network by finding the optimal ordering, where the score of an ordering is the score of the best network consistent with it.

The search operator over the orderings space defined in [8] is called neighbor-swapping operator: $(X_{i_1}, \ldots, X_{i_j}, X_{i_{j+1}}, \ldots) \longmapsto (X_{i_1}, \ldots, X_{i_{j+1}}, X_{i_j}, \ldots)$.

The method performs the search by considering all $n-1$ candidate successors of the current ordering, compares the delta-scores of the successor orderings obtained by these swaps — the difference between their score and the current one, and takes the one that gives the highest delta-score. The tabu list is used to prevent the algorithm from reversing a swap that was executed recently in the search. The process is continued until a local maximum is reached. [8]

4 Local Bayesian Increment Function

Let X and Y be two discrete variables, $\mathbf{Z}$ be a set of discrete variables, and z be an instantiation for $\mathbf{Z}$. $X, Y \notin \mathbf{Z}$.

Definition 5. *According to Moore's recommendation [9] about the chi-squared test, the dataset D satisfying the following condition is* ***sufficiently large*** *for $\{X \cup Y\}$: All cells of $\{X \cup Y\}$ in the contingency table have expected value greater than 1, and at least 80% of the cells in the contingency table about $\{X \cup Y\}$ have expected value greater than 5.*

Definition 6. *According to Moore's recommendation [9] about the chi-squared test, the sub-dataset $D_{\mathbf{Z}=z}$ satisfying the following condition is* ***locally sufficiently large*** *for $\{X \cup Y\}$ given $\boldsymbol{Z} = z$: All cells of $\{X \cup Y\}$ in the contingency table conditioned on $\boldsymbol{Z} = z$ have expected value greater than 1, and at least 80% of the cells in the contingency table about $\{X \cup Y\}$ on $\boldsymbol{Z} = z$ have expected value greater than 5.*

Learning on limited datasets, we relax the "locally sufficiently large" condition: If the number of cases in $D_{\mathbf{Z}=z}$ is much larger than the number of values for $\{X \cup Y\}$, for example $\|D_{\mathbf{Z}=z}\| \geq 4 \times (\|X\| \times \|Y\|)$; then we assume that the sub-dataset $D_{\mathbf{Z}=z}$ is "locally sufficiently large" for $\{X \cup Y\}$ given $\mathbf{Z} = z$.

Let D be a dataset of m cases. Let V be a set of n discrete variables, where X_i in V has r_i possible values $(x_{i1}, x_{i2}, \ldots, x_{ir_i})$. B_P and B_S denote BN structures containing just the variables in V. B_S exactly has one edge $Y \to X_i$ more than B_P. X_i has the parents set Π_i in B_P and the parents set $\Pi_i \cup Y$ in B_S. N_{ijk} is the number of cases in D, which variable X_i is instantiated as x_{ik} and Π_i is instantiated as $\phi_i[j]$. Let $N_{ijk} = \sum_{y} N_{i,\{j \cup y\},k}, N_{ij} = \sum_{k=1}^{r_i} N_{ijk}$. $\hat{\Theta}_i, \hat{\Theta}$ denote the maximum likelihoods of Θ_i, Θ. π_i^l denotes the instantiation of Π_i in the lth case.

Cases occur independently. The prior distribution of possible Bayesian networks is uniform. Given a Bayesian network model, there exist two properties: Parameter Independence and Parameter Modularity. [4]

We apply the BIC formula also used by Steck in [10] : $BIC(B_S) = \log L\left(\hat{\Theta}\right) - \frac{1}{2}\log(m) dim\left(\hat{\Theta}\right) \approx \log(P(D \mid B_S))$ to control the complexity of BN model. BIC adds the penalty of structure complexity to LBI function to avoid overfitting.

Definition 7 (Local Bayesian Increment Function).

$$Lbi(Y, i, \Pi_i) = \log\left(P(B_S, D)/P(B_P, D)\right) \approx BIC(B_S) - BIC(B_P)$$
$$= \log\left(L\left(\hat{\Theta}^{B_S}\right)/L\left(\hat{\Theta}^{B_P}\right)\right) - \frac{1}{2}\log(m)\left[dim\left(\hat{\Theta}^{B_S}\right) - dim\left(\hat{\Theta}^{B_P}\right)\right]$$
$$\log\left(L\left(\hat{\Theta}^{B_S}\right)/L\left(\hat{\Theta}^{B_P}\right)\right) = \log\left(P\left(D \mid \hat{\Theta}^{B_S}\right)\right) - \log\left(P\left(D \mid \hat{\Theta}^{B_P}\right)\right)$$
$$= \sum_{l=1}^{m} \log\left(P\left(x_i^l \mid \hat{\Theta}_i^{B_S}, \pi_i^l \cup y\right)/P\left(x_i^l \mid \hat{\Theta}_i^{B_P}, \pi_i^l\right)\right)$$

On the sub-dataset $D_{\Pi_i=\phi_i[*]}$ which are not "locally sufficiently large" for $\{X \cup Y\}$, we assume that there is overfitting between X and the parents set $\Pi_i \cup Y$ on the $D_{\Pi_i=\phi_i[*]}$. In general, the likelihood of overfitting on some dataset is usually no less than the likelihood of non-overfitting on this dataset. So, we assume that the log-likelihood does not change on the sub-dataset $D_{\Pi_i=\phi_i[*]}$ which are not "locally sufficiently large" for $\{X \cup Y\}$.

$$\sum_{d_l \in D_{\Pi_i=\phi_i[*]}} \log P\left(x^l \mid \hat{\Theta}^{B_S}, \pi_l \cup y\right) = \sum_{d_l \in D_{\Pi_i=\phi_i[*]}} \log P\left(x^l \mid \hat{\Theta}^{B_P}, \pi_l\right). \quad (3)$$

According to (3), we infer the following results:

$$\begin{aligned}
&\log\left(L\left(\hat{\Theta}^{B_S}\right)\right) - \log\left(L\left(\hat{\Theta}^{B_P}\right)\right) \\
&= \sum_j N_{ij} \times I_j(X,Y), \text{for } j, D_{\Pi_i=\phi_i[j]} \text{ is "locally sufficiently large"} \\
&\dim\left(\hat{\Theta}^{B_S}\right) - \dim\left(\hat{\Theta}^{B_P}\right) = (r_y - 1)(r_i - 1)q_i \\
&Lbi(Y, i, \Pi_i) = \sum_j N_{ij} \times I_j(X,Y) - \frac{1}{2}(r_y - 1)(r_i - 1)q_i \log(m), \\
&\qquad\qquad \text{for } j, D_{\Pi_i=\phi_i[j]} \text{ is "locally sufficiently large".}
\end{aligned}$$

Note: $I_j(X,Y)$ is the mutual information between X and Y on $D_{\Pi_i=\phi_i[j]}$.

5 OMRMRG Algorithm

In this section, we divide Ordering-based Max-Relevance and Min-Redundancy Greedy (OMRMRG) algorithm into 2 parts. The first part is to learn Bayesian network given an ordering on the variables. The second part is to learn Bayesian network without the constraint of an ordering on the variables.

5.1 MRMRG Algorithm

MRMRG algorithm initializes the current parents set Π_i of the variable X_i to NULL, and then adds the variables one by one, which acquire the maximal value for Local Bayesian Increment (LBI) function, into the parents set Π_i from $Pre_i - \Pi_i$, until the result of LBI function is no more than 0. Repeating the above steps for every variable, we can obtain an approximately optimal Bayesian network. The pseudo-code of MRMRG algorithm is shown in Fig.1.

Pre_i denotes the set of variables that precede X_i. Π_i denotes the current parents set of the variable X_i. ($k < 5$).

Given an ordering on the variables, MRMRG algorithm improves greedy BN learning algorithms (such as K2 algorithm [3]) in the following two ways in order to learn more accurately and efficiently on limited datasets.

Firstly, on limited datasets, the results of traditional scoring functions (such as K2 score [3],MDL score [5],BDe score [4], etc) $score(C, \Pi_i \cup X_j)$ have less and less reliability and robustness with the dimension increase of $\Pi_i \cup X_j$, so that the formula $Y = \arg \max_{X_j \in Pre_i - \Pi_i} score(C, \Pi_i \cup X_j)$ cannot obtain the variable Y with the maximal score, even cannot acquire a variable with approximately maximal

```
Input: A set $V$ of $n$ variables, an ordering on the variables, a dataset $D$ containing $m$ cases.
Output: for each variable $X_i$ ($i$=1, . . . , $n$), a printout of the parents set $\Pi_i$ .
Procedure MRMRG()
  For every variable $X_i$, call the procedure GSParentsSet ($X_i$, $Pre_i$) and obtain $\Pi_i$ ;
  For every variable $X_i$, output the parents set $\Pi_i$ of the variable $X_i$
Endproc
Procedure GSParentsSet($C$, $Pre_i$)
  Initialize $\Pi_i$ to NULL and $OK$ to TRUE;
  while $OK$
    For every variable $X \in Pre_i - \Pi_i$ , compute the formula $\left\{ I(X;C) - \frac{1}{|\Pi_i|+1} \sum_{X_j \in \Pi_i} I(X;X_j) \right\}$ (1);
    Sort the variables in $Pre_i - \Pi_i$ by descending order according to the value of (1);
    Obtain the top $k$ variables $\{Y_1, Y_2, \ldots, Y_k\}$ from the sorted variables set $Pre_i - \Pi_i$ ;
    $Y_{\max} = \underset{Y_j \in \{Y_1, Y_2, \ldots Y_k\}}{\arg\max} \left[ Lbi(Y_j, i, \Pi_i) \right]$;
    If $Lbi(Y_{\max}, i, \Pi_i) > 0$ then
      $\Pi_i = \Pi_i \cup \{Y_{\max}\}$;
    Else
      $OK$=FALSE;
    Endif
  Endwhile
  Return $\Pi_i$ ;
Endproc
```

Fig. 1. Max-Relevance and Min-Redundancy Greedy BN Learning

score sometimes. Since MRMR technology only uses 2-dimensional computation, it has much higher reliability and robustness than traditional scoring functions on limited datasets. Furthermore, according to the discussion in section 2, we know that if one feature is selected at one time (that is Greedy search), MRMR technology is nearly optimal implementation scheme of Max-Dependence scheme, which is equivalent to the maximal score method, on limited datasets. We consider that for some variable $X_j \in Pre_i - \Pi_i$, if the value of $\{I(X_j;C) - \frac{1}{|\Pi_i|+1} \sum_{X \in \Pi_i} I(X_j;X)\}$ is the largest, then it is the most probable that the value of the formula $score(C, \Pi_i \cup X_j)$ is the largest. Thus, MRMRG algorithm applies Max-Relevance and Min-Redundancy (MRMR) feature selection technology and replaces $score(C, \Pi_i \cup X_j)$ with the formula $\{I(X_j;C) - \frac{1}{|\Pi_i|+1} \sum_{X \in \Pi_i} I(X_j;X)\}$ to obtain the variable Y which gets the maximal score. Firstly, MRMRG algorithm selects the top k variables from the variables set $Pre_i - \Pi_i$ sorted according to the value of the formula $\{I(X_j;C) - \frac{1}{|\Pi_i|+1} \sum_{X \in \Pi_i} I(X_j;X)\}$ by descendant order. Then, it take the variable Y with the largest value of LBI function among the k variables as the variable with the maximal score.

Secondly, MRMRG algorithm proposes LBI function to replace traditional score increment functions (such as K2 [3],MDL [5],BDe [4]) to control the complexity of Bayesian network and to avoid overfitting. When the dataset D is "sufficiently large" for $\{X \cup Y \cup \Pi_i\}$, LBI function is equivalent to K2 increment function. When the dataset D is not "sufficiently large" for $\{X \cup Y \cup \Pi_i\}$, but there exist sub-datasets $D_{\Pi_i=\phi_i[*]}$ that are "locally sufficiently large" for $\{X \cup Y\}$ given $\Pi_i = \phi_i[*]$, MRMRG algorithm can also apply LBI function to improve accuracy and avoid overfitting (see section 4).

5.2 OMRMRG Algorithm

We apply the neighbor-swapping operator (discussed in section 3) and iterative greedy search method over the space of orderings in OMRMRG algorithm. In each iteration, OMRMRG algorithm firstly performs the search by considering all the $n-1$ candidate successors of the current ordering, then compares the scores of the current ordering and the successor orderings, finally takes the one that gives the highest score. The process is continued until a local maximum is reached. OMRMRG algorithm restarts a new iteration by selecting a new current ordering at random, and try the process MAX times to avoid local maximum. The pseudo-code of OMRMRG algorithm is shown in Fig.2.

currOrd denotes the current ordering. *currBN* denotes the Bayesian network with maximal score given the current ordering. *currScore* denotes the maximal score given the current ordering. $succOrd_k (k = 1, \ldots, n-1)$ denotes the successor ordering of the current ordering *currOrd*. $succBN_k$ denotes the Bayesian network with maximal score given the ordering $succOrd_k$. $succScore_k$ denotes the maximal score given the ordering $succOrd_k$. $CParents(X_i)(i = 1, \ldots, n)$ denotes the candidate parents set of the variable $X_i (i = 1, \ldots, n)$. MAX denotes the maximal iteration number.

There are two differences between OMRMRG algorithm and other ordering-based greedy learning algorithms (such as the algorithm proposed by T&K [8]).

Firstly, given an ordering on the variables, OMRMRG algorithm replaces traditional greedy BN learning algorithms with the procedure **MRMRG**() in order to learn more accurately and efficiently on limited datasets.

Secondly, OMRMRG algorithm uses a greedy pruning procedure based on Max Relevance and Min Redundancy technology, which preselects a more accurate set of candidate parents for each variable X_i than the current candidate parents set selection methods described in [12] on limited datasets.

In OMRMRG, we firstly uses the greedy search procedure **GSParentsSet** (X_i, Pre_i) in Fig.1 given $Pre_i = V \setminus X_i$ to get initial candidate parents set Π_i for each variable X_i. Then, according to the symmetry property of neighbor relation, if $(X_i \in \Pi_j)$ or $(X_j \in \Pi_i)$, then OMRMRG algorithm adds X_i into the candidate parents set $CParents(X_j)$ of the variable X_j and adds X_j into the candidate parents set $CParents(X_i)$ of the variable X_i. This step further improves the accuracy of our method **GetCParentsSet()** on limited datasets.

When we consider swapping a pair of adjacent variables $(X_{i_j}, X_{i_{j+1}})$, if $X_{i_{j+1}} \in CParents(X_{i_j})$, then we call **MRMRG()** to compute the score of the successor ordering, else the score of the current ordering is assigned to the successor ordering. The possible parent sets for a variable X_l other than X_{i_j} and $X_{i_{j+1}}$ remain unchanged, as the set of variables preceding them is the same in the two

```
Input: A set $V$ of $n$ variables, a dataset $D$ containing $m$ cases;
Output: the result Bayesian network.
Procedure OMRMRG()
  $t = 0$;  $bestscore = 0$;  GetCParentsSet();
  Repeat
    $local$ = FALSE;  Initialize randomly $currOrd$ as an ordering $\{X_{i_1} \prec \ldots \prec X_{i_j} \prec X_{i_{j+1}} \prec \ldots \prec X_{i_n}\}$;
    $currBN$ = MRMRG($currOrd$);  $currScore$=Score($currBN$);
    Repeat
      For each $succOrd_j$ $\{X_{i_1} \prec \ldots \prec X_{i_{j+1}} \prec X_{i_j} \prec \ldots \prec X_{i_n}\}$ $j$=1,...,$n$-1
        If $X_{i_{j+1}} \in CParents(X_{i_j})$ then $succBN_j$ = MRMRG($succOrd_j$);  $succScore_j$ =Score($succBN_j$);
        Else  $succScore_j = currScore$;  Endif
      Endfor
      $k = \max_{j=1..n-1}\left(succScore_j\right)$;
      If ($succScore_k > currScore$) then $currScore = succScore_k$; $currOrd = succOrd_k$; $currBN = succBN_k$;
      Else  $local$=TRUE;  Endif
    Until $local$;
    $t = t+1$;
    If ($currOrd > bestscore$) then $bestscore = currOrd$;  $resultBN = currBN$; Endif
  Until $t$ = MAX;
  Return the result BN $resultBN$;
Endproc
Procedure GetCParentsSet()
  For every variable $X_i$
    Initialize $Pre_i$ to $V \setminus X_i$;  $\Pi_i$ =GSParentsSet($X_i$, $Pre_i$);  Endfor
  Initialize $CParents(X_i)$, $i$=1, . . . , $n$ to NULL;
  For $X_i \in V$
    For $X_j \in V$
      If $\left(X_i \in \Pi_j\right)$ OR $\left(X_j \in \Pi_i\right)$ then
        $CParents(X_i)$=$CParents(X_i) \cup X_j$;  $CParents(X_j)$=$CParents(X_j) \cup X_i$; Endif
  Endfor
  Return $CParents(X_i)$, $i$=1, . . . , $n$;
Endproc
```

Fig. 2. OMRMRG Algorithm

orderings $\prec$ and $\prec'$. Thus, when calling the procedure **MRMRG()**, we need only recompute the optimal parent sets for the 2 variables X_{i_j} and $X_{i_{j+1}}$.

6 Experimental Results

We implemented OMRMRG algorithm, OK2 algorithm, OR algorithm [11], TPDA algorithm [2] and presented the comparison of the experimental results for 4 implementations. OK2 algorithm incorporates the ordering-based iterative greedy search method into K2 algorithm [3].

Tests were run on a PC with Pentium4 1.5GHz and 1GB RAM. The operating system was Windows 2000. These algorithms were implemented using Matlab 7.0. 3 Bayesian networks were used. Table 1 shows the characteristics of these networks. The characteristics include the number of nodes, the number of arcs, the maximal number of node parents/children(Max In/Out-Degree), and the minimal/maximal number of node values(Domain Range).

Table 1. Bayesian networks

BN	Nodes Num	Arcs Num	Max In/Out-Degree	Domain Range
Alarm	37	46	4/5	2-4
Barley	48	84	4/5	2-67
Munin	189	282	3/15	1-21

From these networks, we performed these experiments with 200, 500, 1000 training cases each. For each network and sample size, we sampled 20 original datasets and recorded the average results by each algorithm. Let $k = 3$ in Fig.1 and MAX=5 in Fig.2.

6.1 Comparison of Runtime

A summary of the time results of the execution of all the 4 algorithms is in Table 2. We normalized the times reported by dividing by the corresponding running time of OMRMRG on the same datasets and reported the averages over sample sizes. Thus, a normalized running time of greater than 1 implies a slower algorithm than OMRMRG on the same learning task. A normalized running time of lower than 1 implies a faster algorithm than OMRMRG.

From the results, we can see that MRMRG has better efficiency than other 3 algorithms OK2, OR and TPDA. In particular, for smaller sample sizes (200, 500), OMRMRG runs several times faster than OK2, OR and TPDA. For larger sample sizes (1000), OMRMRG performs nearly one magnitude faster than OK2.

6.2 Comparison of Accuracy

We compared the accuracy of Bayesian networks learned by these 4 algorithms according to the BDeu score[4]. The BDeu scores of which Equivalent Sample Size (ESS) is 10 in our experiments were calculated on a separate test set sampled from the true Bayesian network containing 20000 samples. Table 3-5 reports the results.

Table 2. Normalized Runtime

Size	OMRMRG	OK2	OR	TPDA
200	1.0	3.93	2.24	7.78
500	1.0	6.46	2.18	6.91
1000	1.0	8.36	2.02	4.62

Table 3. Average BDeu(Alarm)

Size	OMRMRG	OK2	OR	TPDA
200	-14.503	-15.169	-15.832	-21.474
500	-13.882	-14.390	-15.100	-18.097
1000	-13.561	-13.653	-14.819	-15.429

Table 4. Average BDeu(Barley)

Size	OMRMRG	OK2	OR	TPDA
200	-82.047	-83.922	-85.153	-102.287
500	-80.836	-82.044	-82.345	-103.783
1000	-77.064	-77.865	-78.966	-111.069

Table 5. Average BDeu(Munin)

Size	OMRMRG	OK2	OR	TPDA
200	-64.560	-66.803	-90.121	-123.103
500	-63.819	-65.533	-88.604	-125.625
1000	-61.651	-62.247	-87.237	-140.476

From the results, we can see that OMRMRG can learn more accurately than OR and TPDA on limited datasets. In particular, OMRMRG has better accuracy than OK2 on small datasets (200, 500). The accuracy of OMRMRG is almost the same as OK2 on larger datasets relative to the true Bayesian network, such as Insur(1000), Alarm(1000).

7 Conclusions

Efficiency and accuracy are two main indices in evaluating algorithms for learning Bayesian network. OMRMRG algorithm greatly reduces the number of high dimensional computations and improves scalability of learning on limited datasets. The experimental results indicate that OMRMRG has better performance on efficiency and accuracy than most of existing algorithms on limited datasets.

Acknowledgments

This work was supported by NSFC (60503017).

References

1. Spirtes, P., Glymour, C., Scheines, R.: Causation, Prediction and Search. MIT Press, Massachusetts (2000)
2. Cheng, J., Greiner, R.: Learning Belief Networks form Data: An Information Theory Based Approach. Artificial Intelligence 137(1-2), 43–90 (2002)
3. Cooper, G., Herskovits, E.: A Bayesian method for constructing Bayesian belief networks from databases. In: UAI 1991, pp. 86–94. Morgan Kaufmann, San Francisco (1991)
4. Heckerman, D., Geiger, D.: Learning Bayesian Networks: the Combination of Knowledge and Statistical Data. Machine Learning 20(3), 197–243 (1995)
5. Wai, L., Fahiem, B.: Learning Bayesian Belief Networks An approach based on the MDL Principle. Computational Intelligence 10(4), 269–293 (1994)
6. HanChuan, P., Chris, D., Fuhui, L.: Minimum redundancy maximum relevance feature selection. IEEE Intelligent Systems 20(6), 70–71 (2005)
7. HanChuan, P., Fuhui, L., Chris, D.: Feature Selection Based on Mutual Information: Criteria of Max-Dependency, Max-Relevance,and Min-Redundancy. IEEE Transactions on PAMI 27(8), 1226–1238 (2005)
8. Teyssier, M., Koller, D.: Ordering-Based Search: A Simple and Effective Algorithm for Learning Bayesian Networks. In: UAI 2005, pp. 584–590. Morgan Kaufmann, San Francisco (2005)
9. Moore, D.S.: Goodness-of-Fit Techniques. Marcel Dekker, New York (1986)
10. Steck, H.: On the Use of Skeletons when Learning in Bayesian Networks. In: UAI 2000, pp. 558–565. Morgan Kaufmann, San Francisco (2000)
11. Moore, A.W., Wong, W.K.: Optimal Reinsertion: A New Search Operator for Accelerated and More Accurate Bayesian Network Structure Learning. In: ICML 2003, pp. 552–559. AAAI Press, Stanford (2003)
12. Friedman, N., Nachman, I., Peter, D.: Learning Bayesian Network Structure from Massive Datasets: The Sparse Candidate algorithm. In: UAI 1999, pp. 196–205. Morgan Kaufmann, San Francisco (1999)

New Options for Hoeffding Trees

Bernhard Pfahringer, Geoffrey Holmes, and Richard Kirkby

Department of Computer Science
University of Waikato
Hamilton, New Zealand
{bernhard, geoff, rkirkby}@cs.waikato.ac.nz

Abstract. Hoeffding trees are state-of-the-art for processing high-speed data streams. Their ingenuity stems from updating sufficient statistics, only addressing growth when decisions can be made that are guaranteed to be almost identical to those that would be made by conventional batch learning methods. Despite this guarantee, decisions are still subject to limited lookahead and stability issues. In this paper we explore Hoeffding Option Trees, a regular Hoeffding tree containing additional *option* nodes that allow several tests to be applied, leading to multiple Hoeffding trees as separate paths. We show how to control tree growth in order to generate a mixture of paths, and empirically determine a reasonable number of paths. We then empirically evaluate a spectrum of Hoeffding tree variations: single trees, option trees and bagged trees. Finally, we investigate pruning. We show that on some datasets a pruned option tree can be smaller and more accurate than a single tree.

1 Introduction

When training a model on a data stream it is important to be conservative with memory and to make a single scan of the data as quickly as possible. Hoeffding trees [6] achieve this by accumulating sufficient statistics from examples in a node to the point where they can be used to make a sensible split decision. A decision, in fact, that is guaranteed to be almost as reliable as the one that would be made by an algorithm that could see all of the data.

The sufficient statistics turn out to be beneficial for both tree growth and prediction as they can be used to build Naive Bayes models at the leaves of the tree that are more accurate than majority class estimates [7]. There remain situations where Naive Bayes leaves are not always superior to majority class, and [8] shows that adaptively deciding when their use is beneficial per leaf of the tree can further enhance classification performance.

Given this sequence of improvements to the basic algorithm, is it possible to improve predictive performance further? Contemporary machine learning would suggest that the next logical step is to employ ensembles of Hoeffding trees using a framework such as bagging or boosting.

In part, ensemble methods seek to overcome problems inherent in all greedy tree learners. Tree learners have limited lookahead ability (the best scoring single test based on the children that test would generate) and they are not stable. Stability [2] relates to the selection of a particular attribute split when the scores of other attribute splits are close. The examples processed just prior to the split decision could have an undue influence over this decision and it may not be the best decision in the longer term. Breiman [4] demonstrates just how unstable this influence can be, especially when marginal decisions are taken near the root of the tree. Even though decisions are made with more data in a Hoeffding tree, they are still prone to these two issues.

The use of classifier ensembles for data streams has been explored before, most notably in the work on online bagging and boosting by Oza and Russell [10]. Of these though, only bagging appears to work successfully in practice. In addition to the important issue of ensemble model size, ensemble methods produce models that are not interpretable.

In the context of batch learning, these issues led several authors to develop option trees [5,9]. In a sense, option trees represent a middle ground between single trees and ensembles. They are capable of producing useful, and interpretable, additional model structure without consuming too many resources.

Option trees consist of a single structure that efficiently represents multiple trees. A particular example can travel down multiple paths of the tree, contributing, in different ways, to different options. The class of a test example is determined by a committee made up of the predictions of all leaf nodes reached. In the context of a data stream, the idea is to grow multiple options in the same manner as a single Hoeffding tree. By doing so the effects of limited lookahead and instability are mitigated, leading to a more accurate model.

2 Option Trees

Figure 1 is an example of what the top few levels of an option tree can look like. The tree is a regular decision tree in form except for the presence of option nodes, depicted in the figure as rectangles. At these nodes multiple tests will be applied, implying that an example can travel down multiple paths of the decision tree, and arrive at multiple leaves.

Algorithm 1 describes the Hoeffding option tree induction process. This process is a modification of the most recent and more robust version of the original

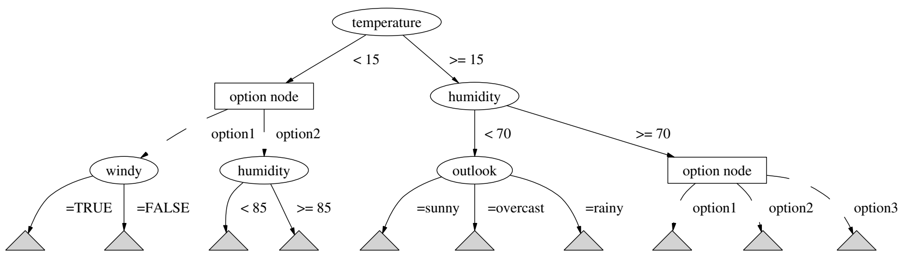


Fig. 1. An option tree

Hoeffding tree induction algorithm [8] to allow exploration of multiple options. Lines 19-34 enable introduction of options into the tree, removing these lines results in the standard Hoeffding tree algorithm, as will setting *maxOptions* to 1. The other differences between this algorithm and the standard one occur in lines 3 and 4, where an example updates multiple option nodes down multiple paths—in the original algorithm only a single leaf will be influenced by each particular example. As is common with standard Hoeffding trees, the evaluation criterion, $\overline{G}$, for all test nodes is information gain. The implementation of this algorithm is referred to as HOT in subsequent discussions (see Section 4).

Predictions by the Hoeffding option tree are made by combining the predictions of all leaves applying to an example. Kohavi and Kunz [9] used majority voting, but like [5] we used weighted voting—experiments suggested that a weighted vote, where the individual probability predictions of each class are summed, performs better than unweighted voting.

2.1 Managing Tree Growth

Option trees have a tendency to grow very rapidly if not controlled. In [9] the authors employ several strategies to alleviate this problem, including limiting the number of options allowed locally per node. Our primary strategy for controlling growth limits the number of options allowed *per example*, placing a global limit that is less trivial to enforce but more effective at regulating growth. Every option node in the tree has a counter ($optionCount$) that is initialized to 1. As the tree grows the counters are updated to reflect the number of leaves reachable by an example at each node. This is not trivial—the number of nodes reachable at a particular node is determined by options found in both its ancestors and descendants. Tree growth is controlled by only allowing an option to be introduced where the counter has not reached the maximum allowable value, which guarantees a maximum number of paths that an example can follow down the tree. Equally this limits the number of leaves that will be involved in an update or prediction.

A range of option limits were tested and averaged across all datasets to determine the optimal number of paths. Figure 2 shows the result. As can be seen, prior to five paths, significant gains in accuracy are apparent, but after that point the accuracy gains diminish. Interestingly, this is exactly the same number of options used by [9] although they arrived at this figure by chance.

2.2 Restricting Additional Splits

In order to generate new structure at an option node we only consider options for attributes that have not already been split at that node. This further cuts down the eagerness of the tree to grow, and overcomes the problem of the same split being repeatedly chosen, resulting in multiple redundant paths.

The final element of tree growth management involves deciding when additional options look sufficiently competitive with existing splits in the tree to

Algorithm 1 Hoeffding option tree induction algorithm, where n_{min} is the grace period, $\overline{G}$ is the split criterion function, R is the range of $\overline{G}$, τ is the tie-breaking threshold, δ is the confidence for the initial splits, δ' is the confidence for additional splits and *maxOptions* is the maximum number of options reachable by a single example

```
1: Let $HOT$ be an option tree with a single leaf (the root)
2: for all training examples do
3:    Sort example into option nodes $L$ using $HOT$
4:    for all option nodes $l$ of the set $L$ do
5:       Update sufficient statistics in $l$
6:       Increment $n_l$, the number of examples seen at $l$
7:       if $n_l \ mod \ n_{min} = 0$ and examples seen at $l$ not all of same class then
8:          if $l$ has no children then
9:             Compute $\overline{G}_l(X_i)$ for each attribute
10:            Let $X_a$ be attribute with highest $\overline{G}_l$
11:            Let $X_b$ be attribute with second-highest $\overline{G}_l$
12:            Compute Hoeffding bound $\epsilon = \sqrt{\frac{R^2 ln(1/\delta)}{2n_l}}$
13:            if $\overline{G}_l(X_a) - \overline{G}_l(X_b) > \epsilon$ or $\epsilon < \tau$ then
14:               Add a node below $l$ that splits on $X_a$
15:               for all branches of the split do
16:                  Add a new option leaf with initialized sufficient statistics
17:               end for
18:            end if
19:         else
20:            if $l.optionCount < maxOptions$ then
21:               Compute $\overline{G}_l(X_i)$ for existing splits and (non-used) attributes
22:               Let $S$ be existing child split with highest $\overline{G}_l$
23:               Let $X$ be (non-used) attribute with highest $\overline{G}_l$
24:               Compute Hoeffding bound $\epsilon = \sqrt{\frac{R^2 ln(1/\delta')}{2n_l}}$
25:               if $\overline{G}_l(X) - \overline{G}_l(S) > \epsilon$ then
26:                  Add an additional child option to $l$ that splits on $X$
27:                  for all branches of the split do
28:                     Add a new option leaf with initialized sufficient statistics
29:                  end for
30:               end if
31:            else
32:               Remove attribute statistics stored at $l$
33:            end if
34:         end if
35:      end if
36:   end for
37: end for
```

warrant their introduction. The δ parameter controls the allowable error in decisions that occur during Hoeffding tree induction. The smaller this value, the longer the tree will wait before committing to a decision to split.

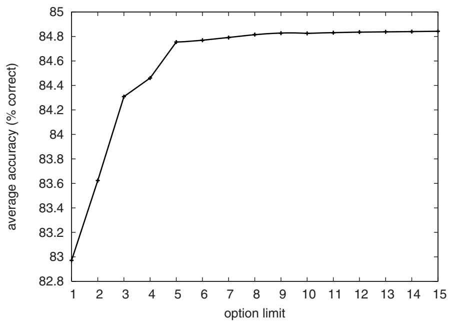


Fig. 2. The average accuracy achieved over all datasets by HOT varying the per-example option limit. Accuracies were estimated using an interleaved test-then-train evaluation over one million examples.

We introduce a second parameter, δ', for deciding when to add another split option beneath a node that already has been split. A new option can be introduced if the best unused attribute looks better than the current best existing option according to the $\overline{G}$ criterion and a Hoeffding bound with confidence δ'. Tie-breaking is not employed for secondary split decisions, so new options will only be possible when a positive difference is detected. δ' can be expressed in terms of a multiplication factor α, specifying a fraction of the original Hoeffding bound:

$$\delta' = e^{\alpha^2 ln(\delta)} \tag{1}$$

For example, with our default parameter settings of $\delta = 1E^{-8}$ and $\alpha = 0.05$ (that is, decisions are 20 times more eager), then from (1) $\delta' \approx 0.955$.

3 Pruning

Option trees improve accuracy at a cost: they consume more memory, as they can grow more rapidly than a standard Hoeffding tree. Not all that additional growth is necessarily beneficial, some subparts of the tree may be redundant, others can negatively impact accuracies. Pruning needs mechanisms for estimating the merit of subtrees and ways of selecting candidates for pruning. In a stream setting an additional issue is choosing an appropriate frequency for pruning as making decisions after every single example might be too time-consuming.

We implemented a pruning method modeled after reduced error pruning, where accuracies of subtrees were estimated by recording accuracy for every example seen at each node, and then periodically subtrees would be removed when their estimated accuracy appeared worse than their parents' accuracy. This method did not distinguish between standard splits and option nodes in the tree. It did not fare well, which we conjecture is due to its local nature, which cannot take into account the global effect averaging multiple options has:

on average a node may look worse than its parent, but it might be less correlated with other options and therefore help achieve the correct global prediction more often than the parent node.

Another approach tried to estimate the global contribution of each option. Every node in the tree with multiple options records accuracy estimates for each option for two settings: accuracy when including this option in global voting, and accuracy when excluding this option from global voting. Once estimated "exclusion" accuracy of an option exceeds the "inclusion" estimate, it (as well as all the substructure rooted at that option) becomes a candidate for pruning. The HOTP algorithm evaluated later in this paper implements this strategy. Every training example is used to update these accuracy estimates, and periodically (after every 100,000 examples) any option that appears to reduce accuracy is removed. Note that this pruning method is specialized and limited to options.

Unfortunately the results in Section 4 highlight that this way of pruning is not often very effective, either. Even though it is often able to reduce the size of the Hoeffding option trees, sometimes back to the size of the baseline Hoeffding tree, this reduction in size at times also causes a loss in accuracy. We have also, unsuccessfully, tried to base the pruning decision on a Hoeffding bound as well to make pruning more reliable.

We conclude that automatic pruning of Hoeffding option trees is very unlikely to produce satisfactory results, but also that pruning can improve performance on selected problems, given careful selection of the right method and parameter settings.

4 Empirical Evaluation

We use a variety of datasets for evaluation, most of which are artificially generated to make up for the lack of large publicly available real-world datasets. Table 1 lists the properties of the datasets.

Table 1. Dataset characteristics

name	nominal	numeric	classes	name	nominal	numeric	classes
GENF1-F10	6	3	2	WAVE21	0	21	3
RTS	10	10	2	WAVE40	0	40	3
RTC	50	50	2	RRBFS	0	10	2
LED	24	0	10	RRBFC	0	50	2
COVERTYPE	44	10	7				

The class functions and the generation processes for GENF1 through to GENF10 are described in [1]. RTS and RTC are simple and complex random tree concepts, based on the generation method described in [6], with 10% added noise. RTS has a maximum depth of 5, with leaves starting at level 3 and a 0.15 chance of leaves thereafter; the final tree had 741 nodes, 509 of which were leaves. RTC has a depth of 10 with leaves starting a level 5; the final tree had 127,837 nodes, 90,259 of which were leaves. Each nominal attribute has 5 possible values. LED, WAVE21 and WAVE40 (waveform without/with noise), and COVERTYPE(CTYPE) are from the UCI [3] collection of datasets. RRBFS refers to a simple random RBF (Radial Basis Function) dataset comprising 100 centers, 10 attributes and 2 classes. RRBFC is more a complex version comprising 1000 centers, 50 attributes and 2 classes.

Table 2. Comparison of algorithms. Accuracy is measured as the final percentage of examples correctly classified over the 10 million test/train interleaved evaluation. Size for HT is measured in terms of the number of tracking nodes, and sizes for HOT, HOTP and Bag10 are measured relative to this. Time is measured as the time per prediction of the final tree relative to HT. The best individual accuracies are indicated in boldface.

	HT		HOT			HOTS		HOTP			Bag10		
dataset	acc	size	acc	size	time	acc	time	acc	size	time	acc	size	time
GENF1	95.03	2031	95.03	1.73	1.10	**95.04**	1.06	95.03	1.09	0.99	95.03	9.57	5.76
GENF2	83.53	3205	93.06	2.73	1.84	93.07	1.61	93.18	0.91	0.91	**94.02**	8.0	6.36
GENF3	97.51	604	97.51	1.19	1.01	97.51	1.00	97.51	1.12	1.01	97.51	10.08	6.5
GENF4	94.37	2823	94.37	2.56	1.42	94.39	1.28	94.27	1.18	1.04	**94.49**	10.32	8.29
GENF5	89.73	3131	92.15	4.80	2.40	92.06	2.10	92.01	1.26	1.04	**92.56**	12.07	8.15
GENF6	92.31	3390	92.85	3.06	1.68	92.87	1.36	92.26	0.94	1.01	**93.13**	9.37	7.84
GENF7	96.62	2417	96.65	4.05	2.39	96.71	1.88	96.15	0.40	1.50	**96.77**	8.12	8.56
GENF8	99.39	563	99.39	2.90	1.61	99.40	1.25	99.38	1.61	1.06	99.40	5.41	6.15
GENF9	96.41	3270	96.46	3.81	2.35	96.50	1.79	95.47	0.55	0.92	**96.67**	6.18	8.46
GENF10	99.87	242	99.87	1.95	1.09	99.87	1.10	99.87	0.98	1.00	99.87	4.14	6.91
RTS	76.81	5013	77.09	2.05	1.14	77.09	1.14	76.91	1.16	1.03	**77.32**	10.31	3.47
RTC	58.84	3881	**61.51**	4.44	1.25	61.50	1.46	57.20	0.07	0.97	59.05	10.44	2.08
LED	73.99	302	73.99	4.71	2.97	73.98	1.75	73.99	0.38	2.04	**74.01**	10.03	6.16
WAVE21	84.90	1451	85.79	4.90	3.00	85.76	2.91	85.56	0.68	2.48	**85.86**	10.13	6.49
WAVE40	84.74	1463	85.68	4.88	2.55	85.69	2.73	85.44	0.53	1.98	**85.85**	9.98	4.95
RRBFS	90.50	2596	91.64	4.78	3.67	90.73	2.31	88.05	0.19	1.34	**92.89**	10.05	9.56
RRBFC	95.73	3101	97.84	4.86	3.21	97.13	2.37	93.64	0.08	3.11	**98.98**	9.68	7.0
CTYPE	70.92	41	71.53	3.39	1.83	71.11	1.92	71.50	3.24	1.83	**72.70**	9.63	4.69
average	87.84	2196	89.02	3.49	2.03	88.91	1.72	88.19	0.91	1.40	89.23	9.48	6.52

The evaluation methodology used every example for testing the model before using it to train. This interleaved test followed by train procedure was carried out on 10 million examples[1]. Table 2 reports the final accuracy, size and speed of the classification models induced. Additionally, the learning curves and model growth curves for selected datasets are plotted (Figure 3). For some datasets the differences in accuracy, as seen in Table 2, are marginal.

Sizes of the tree models are measured by counting the number of *tracker* nodes, i.e. nodes which store sufficient statistics. This measure is proportional to the actual size of the tree in memory, as the sizes of other data structures are insignificant compared to these nodes. In Table 2 the sizes of HOTS (see below)

[1] For COVERTYPE, evaluation was limited to the 581,012 examples available.

are omitted as they are equivalent, by definition, to the size of the corresponding single Hoeffding tree. Prediction speeds of the final models were measured by timing how long it took the model to classify an additional 1 million examples after training was complete. The prediction time figures in Table 2 are relative to the speed of HT[2].

The first, and baseline, algorithm (HT) is a single Hoeffding tree, enhanced with adaptive Naive Bayes leaf predictions. Parameter settings are $n_{min} = 1000$, $\delta = 1E^{-8}$ and $\tau = 0.05$. The second algorithm, HOT, is the Hoeffding option tree algorithm, restricted to a maximum of five option paths. Decisions to add more options to the tree were twenty times more eager than the initial decisions, so, as explained in Section 2.2, the parameter δ' was set to 0.955. HOTS is a modified version of HOT that halts growth once it reaches the same size as the final HT model on the same problem. The statistics in the leaves of the tree continue to be updated, but no additional nodes are added to the tree. This tests whether the accuracy advantages of option trees are solely due to higher memory consumption. HOTP tests one of the more successful attempts at pruning HOT. Every 100,000 examples, option paths are removed if it is estimated that they reduce the overall accuracy of the tree. BAG10 is online bagging [10] using ten Hoeffding trees.

Bagging is clearly the best method in terms of accuracy. This superior position is, however, achieved at high cost in terms of memory (almost ten times more memory is used on average) and prediction speed (six times slower on average). All option tree variants give superior accuracy performance over HT on average, with the options slowing down the prediction speed by at most a factor of two. HOT is more accurate than HT in all cases except LED[3], and sometimes the differences are substantial. This shows the potential of options to overcome issues seen in single trees due to instability and limited lookahead.

Options do not need additional memory to be useful: both HOTS and HOTP use essentially as much memory as a single tree, but still outperform it on average. Of the two, HOTS is the more impressive, performing at the level of the full option tree. It's prediction times are often close to the single tree and at most three times worse. Although HOTP returns good average performance, there are occasions where it does poorly. Its prediction times can be up to three times as high as HT, even though tree sizes are much smaller. This is caused by the presence of multiple options very close to the root of the HOTP tree.

Figure 3 shows the graphs where the option methods gain substantially over the single tree, except for HOTP on RTC, WAVE40, and RRBFS where vigorous pruning prevents the tree from reaching a reasonable size. On the RTC dataset both HOT and HOTS are superior to BAG10. HOTS uses only a little more memory than the single tree and approximately 10% of the ensemble memory.

[2] The actual speed for HT varied from around 4,000 predictions/second (for RTC) to 129,000 predictions/second (for GENF3).

[3] On LED, due to rounding this difference is not apparent in Table 2. All five algorithms are very close to the optimal Bayes error of 26%.

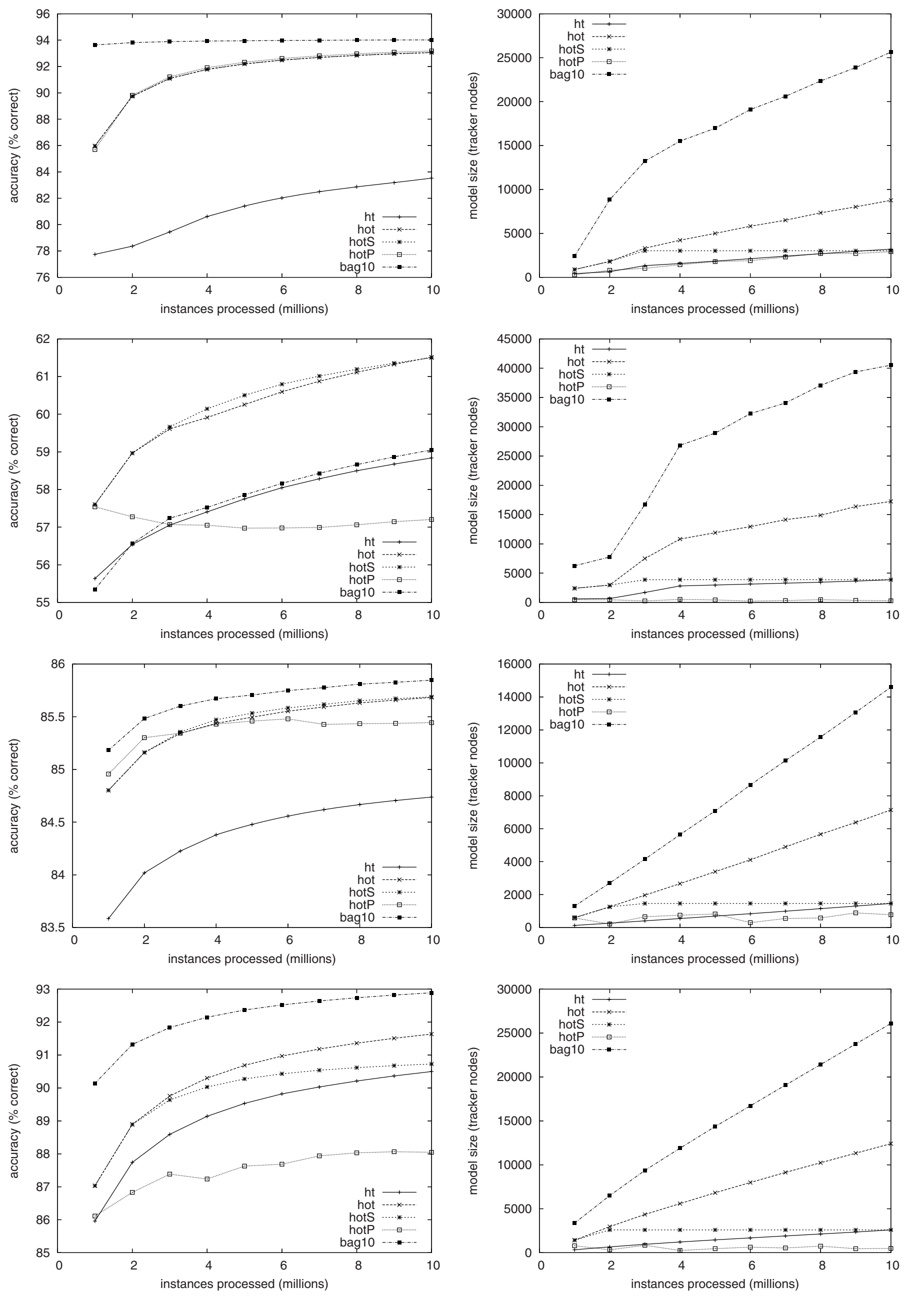


Fig. 3. Accuracy and tree size on (from top to bottom) GENF2, RTC, WAVE40 and RRBFS

5 Conclusions

We have demonstrated the efficacy of incorporating multiple paths via option nodes in Hoeffding trees. We described a method for controlling tree growth, and determined a reasonable number of options to explore. In all but one of our datasets the additional structure improved the performance of the classifier. Option trees represent a useful middle ground between single trees and ensembles. At a fraction of the memory cost an option tree can provide comparable accuracy performance and superior prediction speed which are important factors in data stream processing. The results for pruning unnecessary structure were mixed. Indeed pruning may not be a viable option in a stream setting. Processor time is consumed and the potential accuracy improvements may not outweigh the cost. The idea of simply capping options, to the size of a single tree or to the memory limit of the processor is an attractive and effective low-cost alternative solution.

References

1. Agrawal, R., Imielinski, T., Swami, A.: Database mining: A performance perspective. IEEE Transactions on Knowledge and Data Engineering 5(6), 914–925 (1993)
2. Ali, K.: Learning Probabilistic Relational Concept Descriptions. PhD thesis, University of California, Irvine (1996), http://www.isle.org/~ali/phd/thesis.ps.Z
3. Blake, C.L., Merz, C.J.: UCI repository of machine learning databases (1998)
4. Breiman, L.: Bagging predictors. Machine Learning 24(2), 123–140 (1996)
5. Buntine, W.: Learning classification trees. In: Hand, D.J. (ed.) Artificial Intelligence frontiers in statistics, pp. 182–201. Chapman & Hall, London (1993)
6. Domingos, P., Hulten, G.: Mining high-speed data streams. Knowledge Discovery and Data Mining, 71–80 (2000)
7. Gama, J., Rocha, R., Medas, P.: Accurate decision trees for mining high-speed data streams. In: KDD 2003. Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 523–528. ACM Press, New York (2003)
8. Holmes, G., Kirkby, R., Pfahringer, B.: Stress-testing hoeffding trees. In: Jorge, A.M., Torgo, L., Brazdil, P.B., Camacho, R., Gama, J. (eds.) PKDD 2005. LNCS (LNAI), vol. 3721, Springer, Heidelberg (2005)
9. Kohavi, R., Kunz, C.: Option decision trees with majority votes. In: Fisher, D. (ed.) Machine Learning. Proceedings of the Fourteenth International Conference, Morgan Kaufmann, San Francisco (1997)
10. Oza, N.C., Russell, S.: Online bagging and boosting. In: Artificial Intelligence and Statistics 2001, pp. 105–112. Morgan Kaufmann, San Francisco (2001)

Avoiding Local Minima in Feedforward Neural Networks by Simultaneous Learning

Akarachai Atakulreka[1] and Daricha Sutivong[2]

[1] Department of Computer Engineering,
[2] Department of Industrial Engineering,
Faculty of Engineering, Chulalongkorn University,
Pathumwan Road, Bangkok 10330 Thailand
akarachai.a@hotmail.com, daricha.s@chula.ac.th

Abstract. Feedforward neural networks are particularly useful in learning a training dataset without prior knowledge. However, weight adjusting with a gradient descent may result in the local minimum problem. Repeated training with random starting weights is among the popular methods to avoid this problem, but it requires extensive computational time. This paper proposes a simultaneous training method with removal criteria to eliminate less promising neural networks, which can decrease the probability of achieving a local minimum while efficiently utilizing resources. The experimental results demonstrate the effectiveness and efficiency of the proposed training method in comparison with conventional training.

Keywords: Local Minima, Simultaneous Learning, Removal Criteria, Feedforward Neural Networks.

1 Introduction

Artificial neural networks (ANNs) are widely used in various applications such as classification, approximation [1], signal processing, and pattern recognition [2]. One of the interesting properties of neural networks is the ability to learn from its environment in order to improve its performance. When learning takes place, the connection weights of a network are adjusted using a training dataset without any prior knowledge. The most common learning method used for supervised learning with feedforward neural networks (FNNs) is backpropagation (BP) algorithm. The BP algorithm calculates the gradient of the network's error with respect to the network's modifiable weights. However, the BP algorithm may result in a movement toward the local minimum.

To overcome the local minimum problems, many methods have been proposed. A widely used one is to train a neural network more than once, starting with a random set of weights [3,4]. An advantage of this approach lies in the simplicity of using and applying to other learning algorithms. Nevertheless, this approach requires more time to train the networks. Therefore, this paper proposes simultaneous training along with certain removal criteria. By randomizing starting weights of multiple neural networks, a local minimum is probabilistically avoided. Elimination of less promising neural networks then help save on computational resources. In the experiments, the proposed method is applied with Resilient Backpropagation (RPROP) [5], which is a fast and computationally economical variant of BP algorithm, for comparison of effectiveness and efficiency with conventional neural networks.

This paper is organized as follows: Section 2 describes the local minimum problem in neural networks. Section 3 explains our proposed algorithm, which consists of the training method and removal criteria. Section 4 discusses the experimental procedure. Section 5 compares and analyzes the results. Section 6 involves a comparison with evolving artificial neural networks (EANNs). Finally, section 7 concludes the paper.

2 Local Minima in Neural Networks

Supervised learning of multilayered neural networks with conventional learning algorithms faces the local minimum problems. Gradient descent-type learning algorithms including BP changes the connection weights of a network using a training set of input-output pairs without any prior knowledge. Using a gradient descent to adjust the weights involves following a local slope of the error surface which may lead toward some undesirable points, or the local minima.

In this situation, conventional training of neural networks often gets stuck in the local minima. There are several studies [6]-[11] that investigate this problem, by exploring the appearance of the architecture and the learning environment for the local minima-free condition. Different types of local minima are studied in [12] in order to understand the behavior of an error surface in the neighborhood of a local minimum and to explain the global behavior of the error surface. In fact, the local minima are mainly associated with two factors: the learning style and the network structure. The methods handling the problem can be based on a deterministic approach or a probabilistic approach.

In a deterministic approach, a new learning algorithm, Global descent [13], was proposed in place of primary gradient descent rules. Optimization algorithms are applied in the learning process to avoid local minima in [14] and [15]. These methods can assist in finding the global optimum; however, they are rather time-consuming.

Another alternative, a probabilistic approach, often focuses on a set of weights such as the weight initialization method [16], which can decrease the probability of achieving local minima. Neural networks learning, in which training occurs more than once by starting with a random set of weights is another interesting method. The best neural network is often selected as the one with the lowest error. For example, Park et al. [3] proposed a procedure to determine a structure of the neural network model for predicting sun spots. Their best network was selected from among ten networks of the same architecture, each of which was initiated with a different random set of weights. But the number of random starts can not be specified for certain. To avoid restarting too many times, Iyer and Rhinehart [4] took a probabilistic approach to determine the number of random starts needed in the neural network training. Although this approach is easy to implement, it requires extensive time for training.

Another interesting learning paradigm of ANNs which have been widely studied is the evolving artificial neural networks (EANNs). This model is a combination of

ANNs and evolutionary algorithms (EAs), which have been applied at three major levels: connection weights, architecture, and learning rules. For instance, EPNet [17] is an automatic system for EANNs that uses mutation operators to modify architecture and weights. A survey of EANNs can be found in Yao [18] and the comparison results are demonstrated in [19] and [20]. This method is effective in discovering a global minimum; however, it still requires substantial resources.

3 The Proposed Simultaneous Learning Method

The proposed method involves learning of multiple neural networks similar to the concept of repeated training with a random set of weights that help avoiding local minima. However, in this approach, neural networks learn simultaneously in parallel using multiple initial weights. The method also incorporates removal criteria in order to gradually stop running some neural networks to achieve efficient resource utilization. The following subsections describe the method in more details in two parts: simultaneous learning and removal criteria.

3.1 Simultaneous Learning

The proposed approach applies parallel computing to neural network training, which offers a different training order than the conventional way. The conventional training starts from creating a neural network with a set of initial weights and then trains by adjusting the weights. When the number of epochs reaches the max epoch size, in the case of repeated training, a new network will be recreated with a different set of initial weights and trained as before (Fig. 1(a) shows an example procedure).

In the proposed training method, neural networks are trained simultaneously. In order to train a number of neural networks in parallel, all networks will be created each with a random set of weights and trained simultaneously, all starting from the first epoch and continuing to the max epoch. When training with multiple core

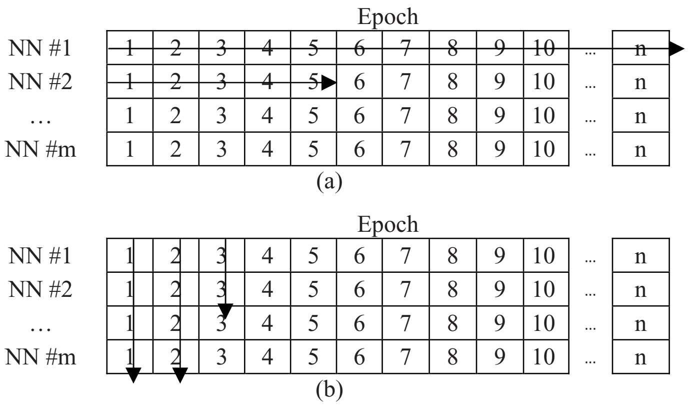


Fig. 1. (a) The conventional training procedure. (b) The simultaneous training procedure. Each slot represents a training epoch. The arrows show the order of training.

processors, all networks may be trained at the same time. Fig. 1(b) illustrates the procedure. The proposed training order allows us to perceive an error trend, as more networks provide better information of the error. Observing a set of errors can assist in determining the max epoch more appropriately. The error trend is then used for comparison among networks in order to stop training of a poor network. The removal criteria are introduced in the next subsection.

3.2 Removal Criteria

In order to reduce the resource consumption in learning, evaluation criteria for neural network removal are incorporated in the simultaneous training in order to stop the neural networks with poor tendency. Specifically, we utilize certain comparative parameter as a predictor of future error behavior. The network with the worst value will be stopped. The comparative parameters under consideration include the followings.

MinimumError	The smallest error from all previous training epochs
LastGenError	The error of the last training epoch
ErrorSlope	The slope computed by linear regression on the last 50 errors
MixedError	The approximate error estimated from the last generation error scaled down by its error tendency, as shown in equation (1) and Fig. 2

$$MixedError = LastGenError(1 - (\arctan(ErrorSlope)/(\pi/2))) \tag{1}$$

In the experiment, we set the maximum epoch to be the number of epochs that the error begins to converge, specifically the error decreases by less than ε or the error

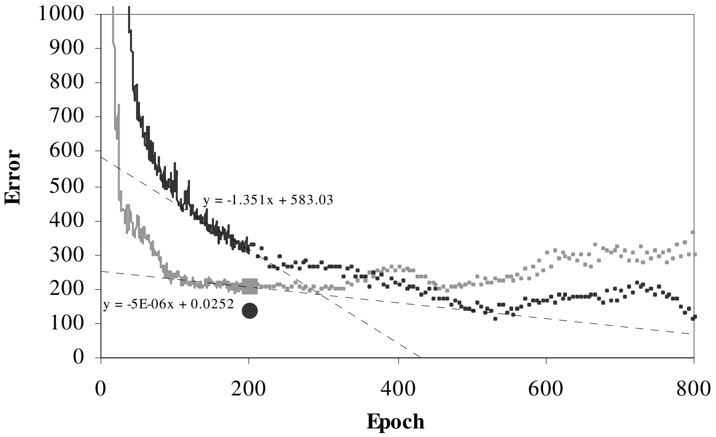


Fig. 2. An example of using the mixed error criterion to evaluate two networks at 200th epoch. Black line and gray line show the error of each network. Solid lines represent the past error, and dotted lines represent the future error. The error slope of each network is computed using linear regression (dashed line) as an input into equation (1). The mixed errors of the black line and the gray line are plotted with a circle dot and a square dot, respectively. Even though the error of the black network at the present epoch is higher than the error of the gray network, this mixed error criterion can predict the future error behavior correctly.

starts to increase indicating that overfitting occurs in the problem. We simultaneously train ten networks and begin choosing the network to stop when the number of training epochs reaches 10% of the max epoch and subsequently at the interval of 5% of the max epoch. Therefore, nine networks will be eliminated after 50% of the max epoch, and the last network is trained until the max epoch for complete convergence. Fig. 3 shows an example of applying the approach with ten networks.

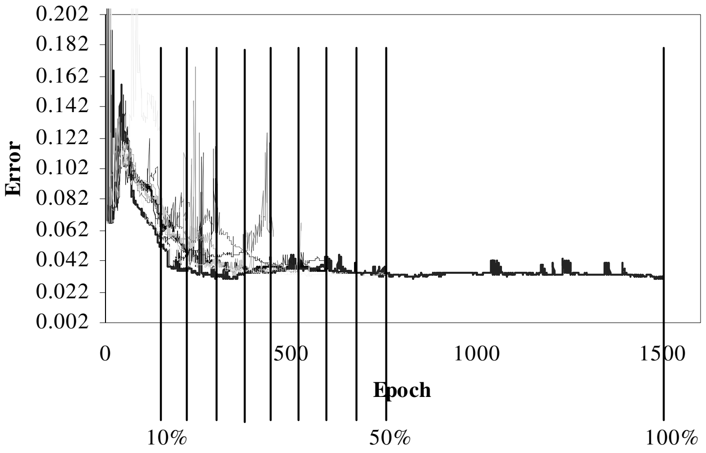


Fig. 3. An example of applying the method to ten neural networks that are trained simultaneously. The vertical lines indicate the epochs at which one network is stopped. The max epoch (100%) is set where error seems to converge.

4 Experimental Details

In our experiment, we apply the proposed method to eight problems. The datasets were obtained from Proben1 [21], which can be categorized into two types: approximation and classification problems. Approximation problems include three

Table 1. Neural network architecture and dataset for testing problems, which include data size, network architecture (input, neurons in hidden layers, output), and the max epoch size

Problem	Dataset size	Network Architecture			Max epoch size
		Input	Hidden	Output	
Approximation Problem					
Building	4208	14	16	3	800
Flare	1066	24	32	3	200
Heart	920	35	32	1	160
Classification Problem					
Card	690	51	32	2	200
Mushroom	8124	125	32	2	
Gene	3175	120	4+2	3	500
Soybean	683	82	16+8	19	1000
Thyroid	7200	21	16+8	3	1500

testing problems: Building, Flare, and Heart. Classification problems include five testing problems: Card, Mushroom, Gene, Soybean, and Thyroid. Numeric data used in the simulations are shown in Table 1.

5 Results and Discussions

The results from all testing problems are used for comparison between the conventional method and the proposed method using various removal criteria. The network architecture is adjusted for each problem. The performances of the two methods are compared using two measures: effectiveness and efficiency.

5.1 Effectiveness

The effectiveness measure compares the solution quality between the two methods using the validation errors from training after the same number of epochs. The number of epochs is equal to the sum of epochs used by all networks in the simultaneous training. This amount limits the number of epochs used in the conventional training, which repeatedly initializes weights when each network reaches the max epoch. The experimental results show an average minimum error after 20 simulations of each method in Table 2.

Table 2. The effectiveness (solution quality) comparison. The table compares conventional training with simultaneous training using various removal criteria. The numbers show a validation error (x 10-3) followed by a percentage improvement from conventional training as shown in parentheses. The best result for each test problem is highlighted in bold.

Problem	Conventional training	Simultaneous training			
		Minimum	Last Gen.	Slope	Mix
Approximation					
Building	8.154	7.809(4.2%)	**7.796(4.4%)**	8.094(0.7%)	**7.796(4.4%)**
Flare	3.897	**3.801(2.5%)**	3.816(2.1%)	3.807(2.3%)	3.821(2.0%)
Heart	43.47	**42.21(2.9%)**	42.24(2.8%)	42.33(2.6%)	42.44(2.4%)
Classification					
Card	410.1	**382.7(6.7%)**	**382.7(6.7%)**	404.6 (1.3%)	**382.7(6.7%)**
Mushroom	9.872	6.204(37.2%)	6.179(37.4%)	8.198(17.0%)	**6.155(37.7%)**
Gene	201.1	181.2 (9.9%)	**180.3(10.3%)**	197.1(2.0%)	181.3(9.8%)
Soybean	450.3	**434.5(3.5%)**	435.4(3.3%)	445.6(1.0%)	436.0(3.2%)
Thyroid	27.36	**25.75(5.9%)**	26.14(4.5%)	27.47 (-0.4%)	26.31(3.9%)

From Table 2, simultaneous training outperforms conventional training for all test problems. There is no single removal criterion that is best for all problems. Minimum error, last error, and mixed error are effective error predictors in different cases, while error slope is never the best choice. An average error over all removal criteria shows an improvement of the proposed approach over conventional training by 2.2-32.3%.

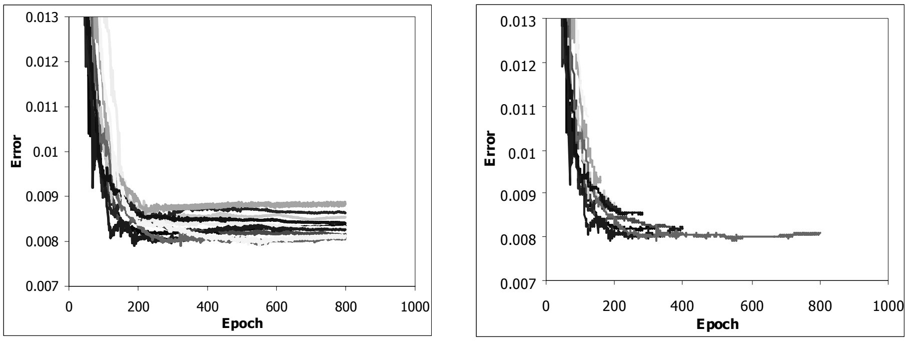
5.2 Efficiency

The efficiency measure compares the training time or the number of epochs used in order to obtain a specified error. Table 3 shows the results, which are the sum of the number of training epochs for each problem. For conventional training, the number of epochs is counted from the start of the first network until an error of the training is less than a specified error for a particular problem, but not more than ten networks. Each neural network training has a limitation on the number of epochs at the max epoch size. For simultaneous training, the number of epochs is calculated from the sum of ten networks' training using various removal criteria. The results are an average over 20 simulations.

From Table 3, the proposed approach outperforms the conventional approach in most cases and could lead to a decrease in computational time of up to 53.9%.

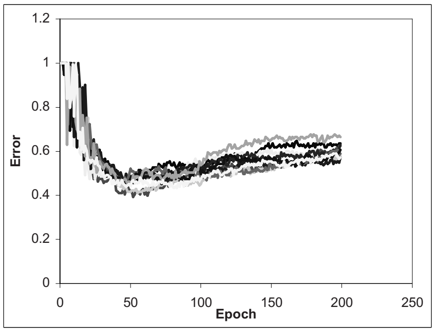
Table 3. The efficiency (computation time) comparison. The table presents an average number of epochs used followed by a percentage improvement from conventional training as shown in parentheses. The best result for each test problem is highlighted in bold.

Problem	Conventional	Simultaneous training			
	training	Minimum	Last Gen.	Slope	Mix
Approximation					
Building	3695.85	2386.3(35.4%)	**2361.6(36.1%)**	2723.6(26.3%)	**2361.6(36.1%)**
Flare	540.55	**402.5(25.6%)**	427.3(21.0%)	406.6(24.8%)	436.2(19.3%)
Heart	784.00	**361.7(53.9%)**	376.1(52.0%)	370.6(52.7%)	387.0(50.6%)
Classification					
Card	782.05	**417.1(46.7%)**	417.1(46.7%)	612.3(21.7%)	417.2(46.7%)
Mushroom	**2632.70**	3034.0(-15.2%)	3034.0(-15.2%)	3034.0(-15.2%)	3034.0(-15.2%)
Gene	2862.90	**1627.7(43.2%)**	**1627.7(43.2%)**	1682.2(41.2%)	1635.6(42.9%)
Soybean	4465.25	**2825.5(36.7%)**	2922.0(34.6%)	3176.6(28.9%)	2922.0(34.6%)
Thyroid	4677.00	**4578.3(2.1%)**	4598.0(1.7%)	5026.9(-7.5%)	4598.0(1.7%)

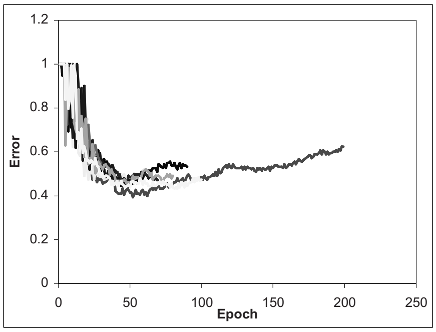


(a) (b)

Fig. 4. Comparison results of the learning process on the Building dataset. (a) The conventional training procedure. (b) The simultaneous training procedure with the mixed error removal criteria.



(a)



(b)

Fig. 5. Comparison results of the learning process on the Card dataset. (a) The conventional training procedure. (b) The simultaneous training procedure with the minimum error removal criteria.

Fig. 4 and 5 show the error behaviors of the two methods on Building and Card problems respectively. In Fig. 4(a), errors from different initial weights converge to different values in conventional training. In simultaneous training, a network with poor error tendency is cut off at each epoch as shown in Fig. 4(b). Fig. 5 displays error behavior when overfitting problem occurs but a good solution is finally found.

Note that among all removal criteria, the minimum error seems to be the best indicator, as it performs best five out of eight in the effectiveness comparison and six out of eight in the efficiency comparison. In most problems, the proposed method improves resource utilization and performance in neural network training.

6 Comparison with EANNs

EPNet is an automatic system for EANNs that uses mutation operators to modify architectures and weights [17]. The experimental results showed that this method could improve the generalization ability. The numerical result for the thyroid problem is shown in the last row of Table 4. We apply the conventional approach and our proposed approach to the thyroid problem for comparison with the EANNs approach, and the results are shown in Table 4.

Table 4. Comparison results on the number of epochs, error and standard deviation of various training methods on the thyroid problem. The total time used by each training method is the maximum number of epochs. The average testing error and standard deviation are also presented in the table.

Method	Number of epochs	Error	SD
Conventional training	2000	0.03033	0.00213
Simultaneous training	7400	0.02575	0.00312
EPNet	109000	0.02115	0.00220

From comparisons in Table 4, EPNet outperforms in achieving the lowest error and reduces the conventional training's error by nearly one third. However, this approach and its accuracy come at the cost of additional computation time in order to perform search. Our proposed simultaneous training uses about four times the number of epochs more than the conventional training but much fewer than EPNet to achieve a reasonable improvement in accuracy.

7 Conclusions

This paper proposes a simultaneous training method with removal criteria for neural network training to avoid local minima which is common in many problems. Experiments are carried out on multiple testing problems for comparison. Results demonstrate that the proposed method can decrease the probability of achieving local minima while utilizing resources efficiently. For time-critical application, the proposed method may help achieving better results using limited computation time, and it can be practicable in the parallel computing environment.

References

1. Ferrari, S., Stengel, R.F.: Smooth Function Approximation Using Neural Networks. IEEE Transactions on Neural Networks 16, 24–38 (2005)
2. Bishop, C.M.: Neural Networks for Pattern Recognition. Clarendon Press, Oxford (1995)
3. Park, Y.R., Murray, T.J., Chung, C.: Predicting Sun Spots Using a Layered Perceptron Neural Network. IEEE Transactions on Neural Networks 7, 501–505 (1996)
4. Iyer, M.S., Rhinehart, R.R.: A Method to Determine the Required Number of Neural-Network Training Repetitions. IEEE Transactions on Neural Networks 10, 427–432 (1999)
5. Riedmiller, M., Braun, H.: A Direct Adaptive Method for Faster Backpropagation Learning: The RPROP Algorithm. In: Proceedings of the IEEE International Conference on Neural Networks, pp. 586–591 (1993)
6. Poston, T., Lee, C.N., Choie, Y., Kwon, Y.: Local Minima and Back Propagation. In: IJCNN-91-Seattle International Joint Conference on Neural Networks, vol. 2, pp. 173–176 (1991)
7. Yu, X.H.: Can Backpropagation Error Surface Not Have Local Minima. IEEE Transactions on Neural Networks 3, 1019–1021 (1992)
8. Gori, M., Tesi, A.: On the Problem of Local Minima in Backpropagation. IEEE Transactions on Pattern Analysis and Machine Intelligence 14, 76–86 (1992)
9. Xiao-Hu, Y., Guo-An, C.: On the Local Minima Free Condition of Backpropagation Learning. IEEE Transactions on Neural Networks 6, 1300–1303 (1995)
10. Fukumizu, K., Amari, S.: Local Minima and Plateaus in Multilayer Neural Networks. In: 9th International Conference on Artificial Neural Networks, vol. 2, pp. 597–602 (1999)
11. De-Shuang, H.: The Local Minima-Free Condition of Feedforward Neural Networks for Outer-Supervised Learning. IEEE Transactions on Systems, Man and Cybernetics, part B 28, 477–480 (1998)
12. Sprinkhuizen-Kuyper, I.G., Boers, E.J.W.: A Local Minimum for the 2-3-1 XOR Network. IEEE Transactions on Neural Networks 10, 968–971 (1999)

13. Cetin, B.C., Burdick, J.W., Barhen, J.: Global Descent Replaces Gradient Descent to Avoid Local Minima Problem in Learning with Artificial Neural Networks. IEEE International Conference on Neural Networks 2, 836–842 (1993)
14. Toh, K.A.: Deterministic Global Optimization for FNN Training. IEEE Transactions on Systems, Man and Cybernetics, part B 33, 977–983 (2003)
15. Jordanov, I.N., Rafik, T.A.: Local Minima Free Neural Network Learning. In: 2nd International IEEE Conference on Intelligent Systems, vol. 1, pp. 34–39 (2004)
16. Wessels, L.F.A., Barnard, E.: Avoiding False Local Minima by Proper Initialization of Connections. IEEE Transactions on Neural Networks 3, 899–905 (1992)
17. Yao, X., Liu, Y.: A New Evolutionary System for Evolving Artificial Neural Networks. IEEE Transactions on Neural Networks 8, 694–713 (1997)
18. Yao, X.: Evolving Artificial Neural Networks. Proceedings of the IEEE 87, 1423–1447 (1999)
19. Sexton, R.S., Gupta, J.N.D.: Comparative Evaluation of Genetic Algorithm and Backpropagation for Training Neural Networks. Information Sciences 129, 45–59 (2000)
20. Cantu-Paz, E., Kamath, C.: An Empirical Comparison of Combinations of Evolutionary Algorithms and Neural Networks for Classification Problems. IEEE Transactions on Systems, Man and Cybernetics, part B 35, 915–927 (2005)
21. Prechelt, L.: Proben1: A Set of Neural Network Benchmark Problems and Benchmarking Rules. Technical report 21/94, Univ. Karlsruhe, Karlsruhe, Germany (1994)

Some Analysis on the Network of Bursting Neurons: Quantifying *Behavioral Synchronization*

Dragos Calitoiu[1], John B. Oommen[2], and Doron Nussbaum[3]

[1] Carleton University, Ottawa, K1S 5B6, Canada
dcalitoi@scs.carleton.ca
[2] *Chancellor's Professor*[⋆]; *Fellow: IEEE* and *Fellow: IAPR*, Carleton University, Ottawa, K1S 5B6, Canada
oommen@scs.carleton.ca
[3] Carleton University, Ottawa, K1S 5B6, Canada
nussbaum@scs.carleton.ca

Abstract. There are numerous families of Neural Networks (NN) used in the study and development of the field of Artificial Intelligence (AI). One of the more recent NNs involves the Bursting neuron, pioneered by Rulkov. The latter has the desirable property that it can also be used to model a system (for example, the "brain") which switches between modes in which the activity is excessive ("bursty"), to the alternate case when the system is "dormant". This paper, which we believe is a of pioneering sort, derives some of the analytic properties of the Bursting neuron, and the associated NN.

To be more specific, the model proposed by Rulkov [11] explains the so-called "bursting" phenomenon in the system (brain), in which a low frequency pulse output serves as an envelope of high frequency spikes. Although various models for bursting have been proposed, Rulkov's model seems to be the one that is both analytically tractable and experimentally meaningful. In this paper, we show that a "small" scale network consisting of Bursting neurons rapidly converges to a synchronized behavior implying that increasing the number of neurons does not contribute significantly to the synchronization of the individual Bursting neurons. The consequences of such a conclusion lead to a phenomenon that we call "*behavioral synchronization*".

1 Introduction

Although the field of Neural Networks (NN) in Artificial Intelligence (AI) has matured, the theoretical aspects of many of these NNs are now at varying levels of development. The primary results currently available involve the convergence, rate of convergence, asymptotic distributions and the structural interconnectivity of the neurons in the NN. As far as we know, the entire concept

[⋆] Prof. Oommen is also an Adjunct Professor with the Department of Information and Communication Technology, at the University of Agder, in Grimstad, Norway. Dr. Calitoiu is now with Health Policy Branch, Health Canada, in Ottawa, Canada.

of synchronization/desynchronization of the neurons has not been studied for most families because such a study not only involves the asymptotic behavior, but also an analysis of transient/periodic phenomena of the individual neurons. In this paper, we shall analyze the synchronization behavior of one such NN, namely the network with Bursting neurons. We believe that this is an open area that can generate theoretical and applied results in Artificial Intelligence (AI). By way of motivation, we suggest a real life rationale for such a study.

One of the fundamental tasks of perception is to extract sensory features from a perceived scene and to group them into coherent clusters. The process of extracting and grouping is essential for resolving segregation, object segmentation and pattern recognition. The sensory segmentation can be explained as the correlation of local qualities within a pattern. The processing of different features (like orientation or color) occurs in different cortical areas, but there are no explanations regarding how these features are integrated/grouped. One hypothesis for perceptual grouping could be the temporal correlations. Schneider and von der Malsburg [12] proposed a correlation theory where features are linked through temporal correlations in the firing patterns of different neurons. A biological implementation of temporal correlation consists of using networks of neural oscillators (for example Bursting neurons), where each neuron represents features of an object. In this architecture, each segment is represented by a cluster of Bursting neurons that are synchronized. To extrapolate this construction, different objects are represented by different clusters whose oscillations are desynchronized from each other. This is essentially the problem alluded to, namely the study of the synchronization/desynchronization in a network of bursting neurons. We study this problem, and an artificial NN which achieves this.

Bursting is a process in which a neuron alternates between a quiescent behavior (i.e., a salient non-spiking phase) and a rapidly varying spiking state where the latter is a fast repetitive spiking phase. A typical explanation for the bursting phenomenon is based on the dynamics of a system of nonlinear equations containing both fast and slow variables. The fast variables generate a firing activity and exhibit multistability between a resting state and the regular firing state. The slow variables, on the other hand, can be considered as control parameters for the fast variables.

Many mathematical complex models of bursting neurons, inspired from the behavior of biological systems, have been developed. But these models are analytically intractable when the individual neurons are connected in networks [1,4,5,8,10].

1.1 Contribution of This Paper

As we stated before, the study of synchronization/desynchronization of arbitrary NNs is a relatively open area for most NN families. The contribution of this paper pertains to novel results related to synchronization of one such NN, namely the Bursting neural network as motivated by Rulkov's model [11]. Such a model permits an explanation for the so-called bursting phenomenon in which a low frequency pulse output appears as an envelope of high frequency spikes.

Without being apologetic, we mention that the paper is fairly theoretical. But al over the results are supported by rigorous simulations. This paper shows that increasing the number of neurons in the network does not significantly increase the synchronization of the bursting. Indeed, we show that (with respect to the number of neurons) the network rapidly converges to a synchronized behavior. This is demonstrated by the inclusion and computation of two newly-proposed measures. These results are, to the best of our knowledge, novel, and of a pioneering sort.

2 Overview of the Field

The problem of describing the synchronization of nonlinear oscillators has been studied extensively. In particular, Winfree [13] and Kuramoto [7] have contributed with strong analytic results. Indeed, many kinds of synchronization (i.e., coordination with respect to time) were reported including those described as Amplitude synchronization or Phase synchronization [9].

In this paper, we propose a new category of synchronization analyzed in a network of Bursting neurons. Instead of describing this phenomenon in terms of two signals, we propose a synchronization that is based on behavior, which we call *"Behavioral Synchronization"*[1].

Consider two systems which are in their two functional modes called A and B, respectively. If both of them are in the same mode, either A or B, then they are said to be in a *synchronized behavioral* state, even though the outputs of the systems may be uncorrelated. In the model we proposed for investigation, the two output signals are chaotic and uncorrelated, and yet the systems can be behaviorally synchronized.

We investigate, in this paper, a network of neurons that can generate the so-called bursting (firing) behavior. The individual neuron displays characteristic firing patterns determined by the number and kind of ion channels in their membranes. The firing process of any neuronal circuit depends on the interaction between the intrinsic properties of the individual neurons and the synaptic interactions that connect them into functional ensembles. One of the related neuroscience problems is to explain how the system's dynamics depend on the properties of the individual neurons, the synaptic architecture by which they are connected, and the strength and time duration of the synaptic connections.

2.1 The Model of Bursting Neuron

In this Section, we present a particular model of a Bursting neuron proposed by Rulkov [11], which is formally described by two dimensional maps as:

$$x(n+1) = \frac{\alpha}{[1 + x(n)^2]} + y(n), \tag{1}$$

[1] We believe that the entire concept of synchronizing/desynchronizing the neurons in a NN is a relatively open area. Indeed, the available results concerning this are scanty.

$$y(n+1) = y(n) - \sigma x(n) - \beta, \tag{2}$$

where $x(n)$ and $y(n)$ are the fast and slow dynamical variables of the "oscillator" neuron, respectively. The slow evolution of $y(n)$ is due to the small values of the positive parameters β and σ, which are of the order of 0.001 [11]. The value of the parameter α is selected to be in the region $\alpha > 4$, where the map produces chaotic oscillations in $x(n)$. Depending on the parameter α, the neuron demonstrates two qualitatively different regimes of chaotic behavior, namely, continuous chaotic oscillations and chaotic bursts. This bursting dynamics was confirmed in experiments done with biological neurons [6].

The slow evolution of $y(n)$ for the next m steps [11] is given by: $y(n+m) = y(n) - m(\beta + \sigma y^*(n,m))$ where $y^*(n,m) = \frac{1}{N}\sum_{j=n+1}^{n+m} x(j)$ is the mean value of $x(n)$ computed for m consecutive iterations.

From the previous relation that describes $y(n+m)$, we see that the value $y(n)$ increases very slowly during the next m steps if $\sigma y^*(n,m) < -\beta$, and decreases very slowly if $\sigma y^*(n,m) > -\beta$. From this behavior, we can approximate $y(n)$ to be a constant of value γ. With this observation, Equation (1) becomes:

$$x(n+1) = \frac{\alpha}{[1 + x(n)^2]} + \gamma. \tag{3}$$

For a specific value of γ, the system leads to a bifurcation which corresponds to the beginning of the burst in the system. The end of the chaotic burst is due to the external crisis of the chaotic attractor in the system. The duration of the chaotic burst is determined by the time interval that is required for the slow variable $y(n)$ to move from value γ_1 to γ_2. The complete mathematical explanation of these two bifurcations can be found in [11], and is omitted here as it is not required for the rest of this paper.

3 The Network of Neurons

Having now introduced individual Bursting neurons, we consider a network built with N such "oscillating" neurons [11], which are coupled electrically to each other through the variable $x(n)$. In this case, the variables $X_i(n)$ and $Y_i(n)$ obey:

$$X_i(n+1) = \frac{\alpha}{[1 + X_i(n)^2]} + Y_i(n) + \frac{\epsilon}{N}\sum_{j=1}^{N} X_j(n), \tag{4}$$

with

$$Y_i(n+1) = Y_i(n) - \sigma X_i(n) - \beta \tag{5}$$

where $X_i(n)$ and $Y_i(n)$ are, respectively, the fast and slow dynamical variables of the i^{th} neuron and ϵ is the coefficient that describes the coupling between the neurons.

The coupling between neurons influences the fast dynamics of each neuron by adding the value $\Delta\gamma = \frac{\epsilon}{N}\sum_{j=1}^{N} X_j(n)$ to the almost constant term γ, in Equation (3). When the i^{th} neuron approaches the first bifurcation, its increased value of

$\gamma+\Delta\gamma$ pushes the neuron over the bifurcation value and triggers the burst. The greater the value of N, the number of neurons which are involved, the greater $\Delta\gamma$ will be, and the triggering impact that is experienced by the remaining cell will be correspondingly larger. Such a switching phenomenon is also observed at the end of each burst. The condition for this is formalized below. The proof is essentially taken from [11], but we have filled in the missing details and modified the result for the scenario of a *network*.

Lemma 1. Consider a system of N neurons described by Equations (4-5). When $\Delta\gamma$ is greater than a threshold, denoted $\Delta\gamma_{max}$, the system generates synchronized bursts.

The proof for this Lemma can be found in the unabridged version of this paper [3] and in [2].

During this process of generating bursts, we observe that the bursts themselves get synchronized and that the fast chaotic oscillations corresponding to each neuron tend to become asynchronous. Thus, the oscillations are asynchronous and only the behavior of the neurons is the same. We shall refer to this process as "behavioral synchronization" which is defined below:

Definition 1. Consider a Bursting neuron that can be in one of the two states: (i) A rapidly varying spiking state described by $min(X_1(t)) > Threshold_1$ for all $t \in [T, T+\theta_1]$ or (ii) A quiescent state described by $max(X_1(t)) < Threshold_2$ for all $t \in [T+\theta_1, T+\theta_2]$, where X_1 is the fast variable, and T is the beginning of the rapidly varying spiking state. These two states are characterized by the time delays θ_1 and θ_2, and two thresholds, $Threshold_1$, and $Threshold_2$.Two neurons in the network are *behaviorally synchronized* if there exist δ_{11}, δ_{12}, and δ_{22} which are small, for which the fast variable X_2 of the second Bursting neuron satisfies one of following conditions: (iii) $min(X_2(t)) > Threshold_1$ for all $t \in [T \pm \delta_{11}, T+\theta_1 \pm \delta_{12}]$ or (iv) $max(X_2(t)) < Threshold_2$ for $t \in [T+\theta_1 \pm \delta_{12}, T+\theta_2 \pm \delta_{22}]$.

This definition shows that if the first neuron has started to burst, the second neuron will burst too, after a short delay of δ_{11}. In such a case the neurons exhibit *behavioral synchronization*. The bursting process will disappear after a time, and we permit a small delay δ_{12} between the end of bursting of the first neuron and the end of the bursting of the second. The quiescent state which comes after the bursting states also permits a time difference of δ_{22}.

An example of such a "behavioral synchronization" for a network of two coupled neurons is presented in Figure 1. The reader should compare this process with the behavior of the uncoupled network, as presented in Figure 2.

3.1 The Problems

In order to characterize the "behavioral synchronization" phenomenon, we propose two new measures that resemble frequencies: (i) High Pseudo-frequency measure : This measure, denoted by $F1$, is computed as the ratio between the average number of zero crossings and the length of the chaotic part of the signal. and (ii) Low Pseudo-frequency measure : This measure, denoted by $F2$, is computed as the average of the inverse of the time between the ends of two consecutive bursts.

The problems to be studied and the respective *modus operandi* of the solutions we propose are: (i) We formally prove that the network of coupled bursting is unstable, and that it permits a chaotic behavior for a range of the control parameter, α. This problem is studied by resorting to the stability of the equilibrium points of the network. (ii) Finally, we explore the variation of these two new measures, namely $F1$ and $F2$, along with the cross correlation between each pair of neurons as a function of the size of the coupled network.

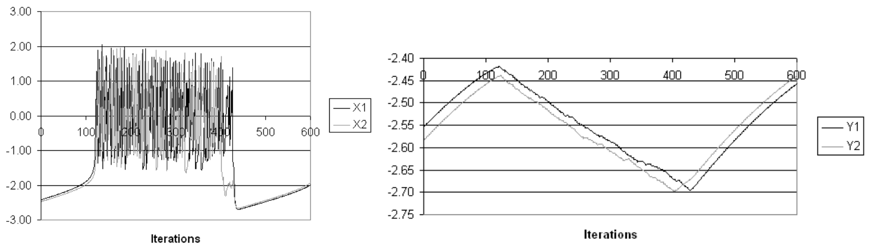


Fig. 1. The variations of $X_1(n)$ and $X_2(n)$ (on the left) and $Y_1(n)$ and $Y_2(n)$ (on the right) for a network of two neurons which are coupled

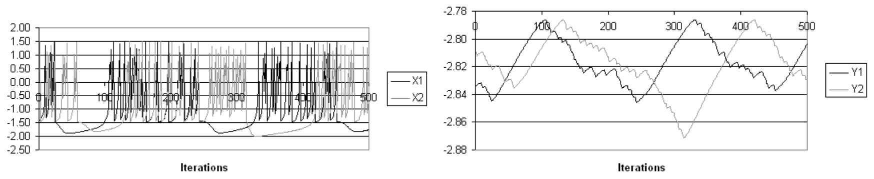


Fig. 2. The variations of $X_1(n)$ and $X_2(n)$ (on the left) and $Y_1(n)$ and $Y_2(n)$ (on the right) for a network of two neurons which are not coupled

4 Simulations Results for Computing New Measures

In Section 3, we showed the chaotic properties of an individual Bursting neuron, and of a network of such neurons. We now address the question whether a coupled network results in synchronization. In other words, if we are given a network with thousands of neurons would it be possible to describe the synchronization by computing the earlier mentioned measures, namely the pseudo-frequencies $F1$ and $F2$, and the Cross Correlation Coefficient CC.

The problem we face is of a computational sort. Indeed, computing these measures when the number of neurons is large is almost prohibitively expensive. But it turns out that the $F1$, $F2$ and CC tend to converge (and that, rather quickly) with the number of neurons. In other words, it happens that the synchronization behavior for relatively small N (say 4 or 5) actually approximates the behavior for large values of N with a negligible error. To demonstrate this we numerically compute these indices for networks of various sizes.

We numerically analyze the Bursting model of the neuron for a few cases, namely for networks in which $N = 2$ (when the pair is coupled/uncoupled) and when $N = 3, 4$. Of course, these experiments were done for an ensemble of parameter settings[2], but in the interest of brevity we merely mention a single case when the parameters were set as is typical in a real-life setting. Thus, for all the experiments reported here, we have set the parameters to be as realistic as possible: $\alpha = 4.3$, $\sigma = 0.001$, $\beta = 0.001$. The coupling coefficient for the experiments with networks involving coupled neurons was $\epsilon = 0.1$.

The notation we use is: (i) E_p = The iteration index of the termination of the previous burst (i.e, the "Previous End"), (ii) S = The iteration index of the start of the Bursting period, (iii) E = The iteration index of the end of the Bursting period, (iv) Z = The number of zero crossings during the length of the burst. Note that as a result of these terms, the length of the burst is $E - S$, and the length of the "periodic signal" is the difference $E - E_p$.

4.1 A Network with Two Neurons

Consider the case of a network with two neurons (characterized by their fast and slow variable, namely $(X_1(n), Y_1(n))$ and $(X_2(n), Y_2(n))$. The behavior of the coupled neurons is presented in Figure 1. The behavior of the uncoupled neurons is presented in Figure 2.

For the case of the coupled neurons, we compute the Largest Lyapunov Exponent[3] (LLE) for each signal, by evaluating it for the firsts 5 bursts of each signals. We find that: (i) For the first neuron, the LLEs are $\{0.43011, 0.3612, 0.2298, 0.2251, 0.3053\}$ and the average LLE is 0.3103. (ii) For the second neuron, the LLEs are $\{0.3242, 0.3484, 0.2089, 0.2860, 0.2962\}$ and the average LLE is 0.2927. In all the situations, the positive values for the LLE correspond to a chaotic behavior.

Observe also the appearance of behavioral synchronization. From Table 1- top part, we see that for the network without coupling, the S iteration indices[4] for the X_1 and X_2 values are: $X_1 = 3201$ vs. $X_2 = 3211$ for a specific burst, and $X_1 = 3606$ vs. $X_2 = 3633$ for the following burst.

As opposed to this, from Table 1 - bottom part, the difference between the corresponding S indices *after* coupling is: $X_1 = 1631$ vs. $X_2 = 1632$ for a specific burst and $X_1 = 2136$ vs. $X_2 = 2137$ for the following burst, which represents

[2] These parameters will be the same as the number of neurons is increased.

[3] The Largest Lyapunov Exponent is a quantitative measure that describes the sensitive dependence of the dynamical system on the initial conditions. It defines the average rate of divergence of two neighboring trajectories. The existence of a positive LLE is a widely-used definition of deterministic chaos.

[4] All the interpretations are made using only the phenomenon displayed by the fast variable. The behavior of the slow variable mimics the former.

Table 1. The values of the pertinent indices E_p, S, E, and Z etc. (see Section 4.1) for a network involving two *uncoupled* neurons (top table) and a network involving two *coupled* neurons (bottom table). The top table shows the corresponding indices and computed frequencies for the *first* signal (left) and for the *second* signal (right). Correspondingly, the bottom table shows the corresponding indices and computed frequencies for the *first* signal (left) and for the *second* signal (right).

1st	E_p	S	E	Z	$Z/(E-S)$	2nd	E_p	S	E	Z	$Z/(E-S)$
sig.	2849	2932	3094	62	0.3827	sig.	2734	2821	2957	34	0.25
	3094	3201	3325	41	0.3306		2957	3033	3124	38	0.4175
	3325	3404	3523	47	0.3949		3124	3211	3371	46	0.2875
	3523	3606	3742	41	0.3014		3371	3460	3557	32	0.3298
	3742	3832	3951	35	0.2941		3557	3633	3813	60	0.3333
			Avg.	45.2	0.34074				Avg.	42	0.32362

1st	E_p	S	E	Z	$Z/(E-S)$	2nd	E_p	S	E	Z	$Z/(E-S)$
sig.	1478	1632	1915	145	0.5123	sig.	1447	1631	1927	150	0.5067
	1915	2136	2472	160	0.4567		1927	2137	2453	155	0.4905
	2472	2704	2940	98	0.4025		2453	2701	2945	127	0.5204
	2940	3112	3331	115	0.5159		2945	3112	3347	122	0.5191
	3331	3522	3826	143	0.4703		3347	3524	3801	130	0.4693
			Avg.	132.2	0.4801				Avg.	136.8	0.5012

a difference of only a single iteration unit. It is thus clear that, in the case of coupled neurons, the beginning of the bursting behavior is almost synchronized.

The reader should observe from Table 1 - top part that the values of $F1$ are approximately equal for both the neurons. The same observation is also true for $F2$. When a coupling occurs, $F1$, the high pseudo-frequency that describes the chaotic oscillations, is larger, and $F2$, the low pseudo-frequency that describes the overall signal, is smaller (see Table 1 - bottom part). The variation of $F2$ can be seen as an envelope of signals X_1 and X_2.

To further illustrate the synchronic behavior, we also compute the cross correlation between the two signals. In Table 2, we present the values for the CC for the case of the uncoupled and coupled signals, respectively. The average values for CC are 0.2528 and 0.2015 for these cases. It is thus clear that although the behavior is the *same*, namely that, both are of a bursting nature, the signals themselves are *marginally correlated*, and that the coupling doesn't increase the relationship between them.

We did also simulations for networks of 3 and 4 coupled neurons. The corresponding results can be found in [2] and [3].

4.2 The Variation of $F1$, $F2$ and CC

We explore in this section the variation of these two new measures, namely $F1$ and $F2$, along with the cross correlation between each pair of neurons as a function of the size of the coupled network.

Table 2. The Cross Correlations between the pairs of signals: *without coupling* (left table) and *with coupling* (right table)

$S(1)$	$E(1)$	$S(2)$	$E(2)$	CC
2932	3068	2821	2957	0.2432
3201	3292	3033	3124	0.2481
3404	3523	3211	3430	0.2849
3606	3703	3460	3557	0.2093
3832	3951	3633	3752	0.2789
			Averages	0.2528

$S(1)$	$E(1)$	$S(2)$	$E(2)$	CC
1632	1915	1632	1915	0.2384
2137	2453	2137	2453	0.1557
2704	2940	2704	2940	0.2504
3112	3331	3112	3331	0.1725
3524	3801	3524	3801	0.1906
			Averages	0.2015

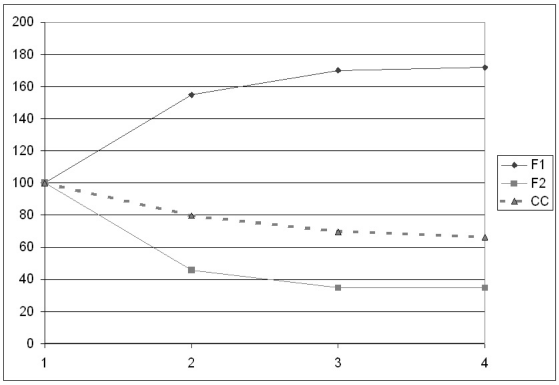


Fig. 3. The relative variations (in percents) for F1, F2 and CC, for the case of 2 coupled neurons (point 2), 3 coupled neurons (point 3) and 4 coupled neurons (point 4) compared with the case of two uncoupled neurons (point 1)

The pseudo-frequencies $F2$ have a value of 0.0045 and 0.0046 in the case of a network of two uncoupled neurons, a smaller value of 0.0021 when we encounter a network of two *coupled* neurons, and a much smaller value of 0.0016 for the case of a network of three coupled neurons.

We present in Figure 3 the evolutions of $F1$, $F2$ and CC. The graph represents the relative variations (in percentages) of the measures for the case of having 2, 3 and 4 coupled neurons, as opposed to the scenarios of having uncoupled neurons. The computed relative variation is compared to the measures obtained for two uncoupled neurons.

To conclude this section we can state that if one increases the size of the network N, the measures $F1$, $F2$ and CC will have an asymptotic behavior as depicted in Figure 3.

5 Conclusions

The Bursting neuron (modelled with Rulkov's settings) permits an explanation for the so-called *bursting* phenomenon in which a low frequency pulse output serves as an envelope of high frequency spikes. In this paper, we showed that the network rapidly converges to a synchronized behavior implying that increasing the number of neurons does not contribute significantly to the synchronization of the individual Bursting neurons.

References

1. AvRon, E., Parnas, H., Segel, L.A.: A basic biophysical model for bursting neurons. Biological Cybernetics 69, 87–95 (1993)
2. Calitoiu, D.: Chaotic System of Pattern Recognition and Brain Modeling. Ph.D. Thesis, Carleton University, Ottawa (2006)
3. Calitoiu, D., Oommen, B.J., Nussbaum, D.: Controlling behavioral synchronization in a network of bursting neurons. Unabridged version of this paper (submitted for publication), available at `www.scs.carleton.ca/~dcalitoi`
4. de Vries, G.: Multiples bifurcations in a polynomial model of bursting oscillations. Journal of Nonlinear Science 8, 281–316 (1998)
5. Doiron, B., Laing, C.R., Longtin, A., Maler, L.: Ghostbursting: a novel bursting mechanism in pyramidal cells. Journal of Computational Neuroscience 12, 5–25 (2002)
6. Elson, R.C., Selverston, A.I., Huerta, R., Rulkov, N.F., Rabinovich, M.I., Abarbanel, H.D.I.: Synchronous behavior of two coupled biological neurons. Physical Review Letters 81, 5692–5695 (1998)
7. Kuramoto, Y.: Chemical oscillations, waves and turbulence. Springer, New York (1984)
8. Laing, C.R., Logtin, A.: A two-variable model of somatic-dentritic interactions in a bursting neuron. Bulletin of Mathematical Biology, 1–32 (2002)
9. Pikovsky, A.S., Rosemblum, M., Kurths, J.: Synchronization: A Universal Concept in Nonlinear Sciences. Cambridge University Press, Cambridge (2001)
10. Rinzel, J., Ermentrout, G.B.: Analysis of neural excitability and oscillations. In: Koch, C., Segev, I. (eds.) Methods in Neuronal Modeling, MIT Press, Cambridge (1998)
11. Rulkov, N.F.: Regularization of synchronized bursts. Physical Review Letters 86, 183–186 (2001)
12. Schneider, W., von der Malsburg, C.: A neural cocktail-party processor. Biological Cybernetics 54, 29–40 (1986)
13. Winfree, A.: The geometry of biological time. Springer, New York (1982)

Comparative Analysis of Multiple Neural Networks for Online Identification of a UAV

Vishwas Puttige, Sreenatha Anavatti, and Tapabrata Ray

School of ACME, UNSW@ADFA, Canberra, Australia
{v.puttige,agsrenat,t.ray}@adfa.edu.au

Abstract. This paper sumarises a comparative study of multiple neural networks as applied for the identification of the dynamics of an Unmanned Aerial Vehicle (UAV). Each of the networks are based on non-linear autoregressive technique and are trained online. Variations in the architecture, batch size and the initial weights of the multi-network are analysed. A dynamic selection mechanism optimally chooses the most suitable output from the host of networks based on a selection criteria.

1 Introduction

UAVs have gained importance due to their reduced operating costs, and repeated operations without human intervention. The interdisciplinary nature of the UAV control systems is attracting researchers from all the fields of engineering. Currently, there are a few universities working towards the development and implementation of different algorithms on UAVs in real-time [11,5,2]. One such fixed-wing platform is under development at the School of Aerospace, Civil and Mechanical Engineering in UNSW@ADFA.

The UAV can be modeled as a Multi-Input Multi-Output (MIMO) system. The conventional physical model of aircrafts may not be applicable for UAVs because of the difference in aerodynamics, low altitude and low speed flights of the UAVs. As an alternative, identification of the UAV using flight data can be utilized. The Flight Control System (FCS) for the UAV performs tasks similar to a pilot for a manned aircraft and is considered as the brain of the UAV. Robust control techniques, capable of adapting themselves to the changes in dynamics of the platform are necessary for the autonomous flight. Such controllers can be developed with the aid of suitable system identification (ID) techniques. Realistic system identification developed using flight data is required for understanding the system behavior and dynamics of the UAV and to facilitate the controller design.

Offline trained neural network models are robust to environmental noise but the application is restricted to a range of flight. When the dynamics of the UAV changes beyond this range the prediction error is significant. As an alternative simpler neural networks can be trained online during the UAV's flight. Online networks adapt to the changes in dynamics of the nonlinear system by training when necessary. The accuracy of prediction depends on various factors including the initial weights of the network. Even though most neural network training techniques have the ability to converge to the target response, with the change in initial random weights the quality of convergence changes. Since the behaviour of the UAV varies with flight condition, having a single online network with random weights for modeling the UAV for all the flight conditions may not be possible. By having multiple networks with different initial weights and architectures and a dynamic switching technique to choose outputs from the different networks, robust prediction can be achieved.

The concept of multiple neural network models for adaptive control was introduced by Narendra in [8]. In the past, multi-networks have been used for linear systems or offline modeling that are mainly based on numerical simulations [12,3,1]. In this work a comparative study of multi-networks with variations in the network architecture and training parameters is performed. The switching between the online models is carried out based upon suitable error criterion. Each of the networks use a Multi-Input Multi-Output architecture. The models are based on Autoregressive technique with Exogenous inputs (ARX) introduced by Ljung in [6]. A novel training method is adapted for the online models where the network is trained with small batches of data and the weights from the previous training are retained in memory [9]. The retraining of the network is carried out only when the error in prediction is beyond a certain threshold.

The online training scheme for the ARX networks is explained in section 2. Different architectures for multi-networks are explained and compared in section 3. The results from the simulation of different networks are presented in section 4. The results to show the superiority of the proposed architecture are presented for a particular subset of the identification and the results are compared using the Mean Square Error (MSE). Some concluding remarks are presented in the section 5.

2 System Identification Based on Neural Networks

Neural networks for system identification is a well explored area in the study of control systems. Various types of networks have been developed and adapted for different applications. It has been shown in the past that recursive network structures outperform the non-recursive structures in modeling and control of non-linear systems [6]. In the autoregressive neural network model the network retains information about the previous outputs and inputs to predict the next output. This provides equivalent retention capabilities of the dynamics of the UAV by the network. The predicted output of a Nonlinear ARX (NARX) model can be obtained as [6,9]

$$\begin{aligned}\widehat{y}(t|\theta) = g(a_1 y(t-1) + a_2 y(t-2) + \\ + a_{na} y(t-na) + b_1 u(t-1) + .. + b_{nb} u(t-nb))\end{aligned} \tag{1}$$

where θ is the coefficient matrix which gives the influence of past outputs $(a_1, .., a_{na})$ and influence of past inputs $(b_1, .., b_{nb})$ on each of the subsequent outputs. The nonlinear function is defined by g, the y and u terms correspond to past outputs and past inputs respectively.

The above equation can be simplified as

$$\widehat{y}(t) = g(\theta, \phi(t)) \tag{2}$$

where,

$$\theta = (a_1, a_2, ...a_{na}, b_1, b_2, ..., b_{nb})$$
$$\phi(t) = (y(t-1), ...y(t-na), u(t-nk), ..., u(t-nk-nb+1))$$

Here ϕ is the matrix of past inputs and outputs called the regressor and is available from memory. To obtain the coefficients θ, many assumptions and detailed knowledge of the plant are necessary [4]. Hence for a dynamic nonlinear system such as the UAV it may not be feasible. This can be avoided by using black-box methods such as the neural networks. The output of a two layered neural network is given as

$$\widehat{y_i}(t) = F_i(\sum_{j=0}^{l1} W_{2ij} G_j(\sum_{k=1}^{l2} W_{1jk} x_k(t) + W_{1j0}) + W_{2i0}) \tag{3}$$

In the above equation F and G are the activation functions, $l1$ and $l2$ are the number of neurons in the two layers, W_{1j0} and W_{2i0} are the bias to the two layers and x_k is the network input. In most of the cases the nonlinearities are best represented by the hyperbolic tangent function as the activation function G and a linear relation F. W_1 and W_2 are the weights from the hidden layer and the output layer respectively. These weights correspond to the θ matrix in equation 2. Hence the problem of obtaining the best prediction depends on the adapted network structure and the training method.

Iterative training is performed to minimize an error function using the Levenberg Marquardt (LM) technique [7]. The LM training is a gradient based technique allowing fast error minimisation. The goal of the training is to obtain the most suitable values of the weights for closest possible prediction through repetitive iterations. Conventionally, LM method works on the principle of minimizing the mean squared error between the actual output of the system and the predicted output of the network given by [6]

$$V_N(\theta) = \frac{1}{N} \sum_{t=1}^{N} (y(t) - \widehat{y}(t))^2 \tag{4}$$

The updated value for θ after each iteration is given by

$$\theta_{i+1} = \theta_i - \mu_i R_i^{-1} G_i \tag{5}$$

where μ is the step length, R is a matrix responsible for the search direction and G is the gradient given by

$$G_i = \acute{V_N}(\theta_i) = -\frac{1}{N} \sum_{t=1}^{N} (y(t) - \widehat{y}(t|\theta_i))\psi(t, \theta_i) \tag{6}$$

where ψ is the derivative of the predicted output with respect to co-efficient matrix θ. The search direction is given by $R_i^{-1}G_i$ which is used to update the weights. For the LM method R_i is given as

$$R_i = \frac{1}{N}\sum_{t=1}^{N}(\psi(t,\widehat{\theta}_i), \psi^T(t,\widehat{\theta}_i)) + \mu I \tag{7}$$

In this equation the first part is the Hessian matrix which is a square matrix with second order partial derivatives of a function. μ is the step size and I is the identity matrix. For zero value of the step size the LM algorithm changes to the Gauss-Newton method.

The network is trained online with small batches of data formed by retaining the past values. The batch is formed by sliding the oldest data set out and allowing the present data set in. The weights from the trained network is retained in memory and is used for future predictions. The network is retrained only when the error in prediction exceeds a predefined bound. The performance of an online network largely depends on the hidden layer size, the batch length and the initial weights chosen for training. The effect of any one of these parameters can be observed by varying it and maintaining the others constant. For a given flight condition the network displays enhanced performance for a range of values in each of these parameters. To combine the best of each of these parameters separately a multi-network architecture is adapted.

3 Multiple Networks Architecture

The design of a reasonable offline model involves the availability of suitable flight data encompassing all possible regimes. This requires a large network with the ability to learn the changes in dynamics of the system at all the flight conditions [10]. Since this is not practical, smaller networks can be trained online to predict the behaviour of the UAV. It may not be possible to train a single online network to capture the UAV dynamics for the entire course of the flight. In a multi-network architecture, networks with different structures and training parameters are used to obtain accurate prediction. Another reason posing a necessity for using multiple online networks for prediction is related to the effect of initial weights on the response of the system. The online networks start with a random set of weights and the weights are updated iteratively for convergence using LM method explained in the previous section. Even though for all sets of random weights the network converges to the desired response the transients observed during the initial training phase is different for each set of weights. Such networks used to aid the control process may lead to instability in the system. The peak performance of a network for a given set of initial weights is restricted to a bound of input output data. Multiple networks with different initial random weights and a dynamic switching technique provide better prediction than any of the individual networks tested for different flight conditions.

The online models are compared and dynamically selected based on a particular selection criterion. This ensures a better performance compared to any of the

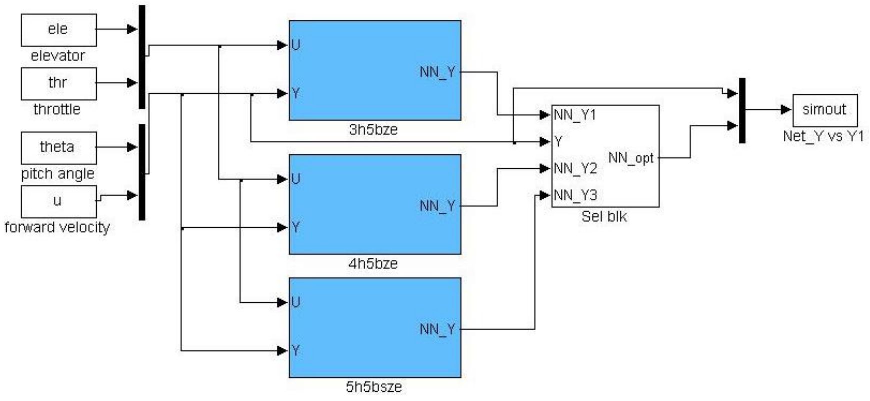


Fig. 1. Multi-network architecture with 3 online networks of different number of hidden neurons

individual networks. Two different criteria are tested for the dynamic selection between networks. The first type compared the instantaneous error between the present system output and the predicted network output. The output from the network with the smallest instantaneous error is selected. With this selection criterion the steady state error is small but the switching between networks is very high. This may lead to large switching transients when used in combination with a controller. In the second type of selection, not only the present but also the past performances of the networks are considered. In this method a weighted average of the instantaneous error and the past mean square error is used as the criterion as shown in equation 8. This provides more consistency in the output leading to lesser switching between them.

$$Err = w_1 * MSE + w_2 * E_{inst} \tag{8}$$

Here Err is the weighted sum of the errors used for the dynamic selection, w_1 and w_2 are the weights assigned to the MSE and the instantaneous error (E_{inst}). The values of the weights are decided based on the noise level in the network inputs and experimental conclusion. The network output with a smaller value of Err is used as the output of the multi-network. The errors are checked at every instant of time and the outputs are switched suitably.

Table 1. Mean square error values of the Multi-network model and individual networks with different hidden neurons (h)

Outputs	**NN1 (h=3)**	**NN2 (h=4)**	**NN3 (h=5)**	**Multi-net**
θ	0.1991	0.1895	0.1392	0.0476
u	0.0142	0.0228	0.0105	0.0048

Table 2. Mean square error values of the Multi-network model and individual networks with different batch sizes (b)

Outputs	NN1 (b=3)	NN2 (b=4)	NN3 (b=5)	Multi-net
θ	0.1267	0.3304	0.6622	0.0819
u	0.0392	0.0814	0.2379	0.0363

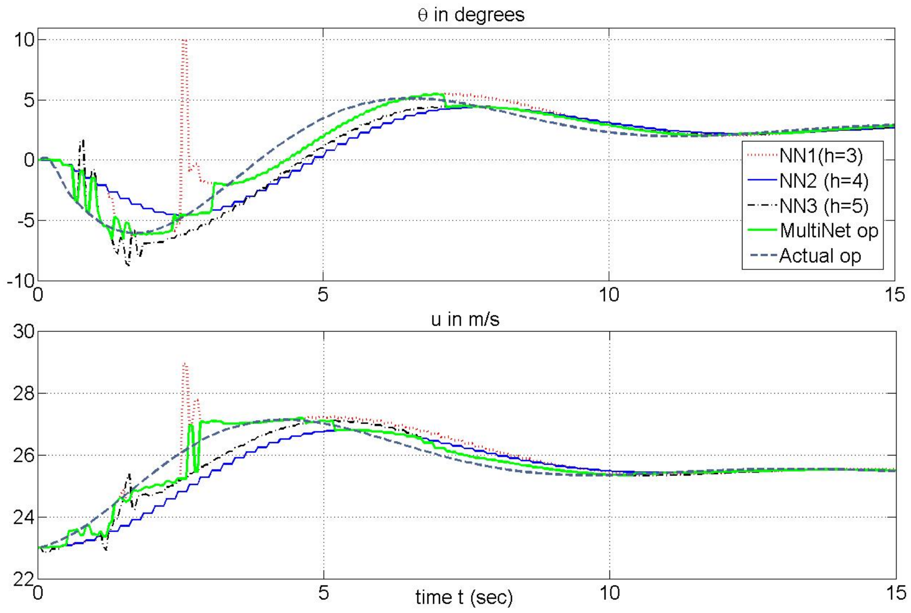


Fig. 2. Comparison of Multi-network output with fixed batch size 5 against the system output

As a comparative case study, different network and training parameters are altered to compare the response. It is observed that variations in number of neurons in the hidden layer and variations in the batch size has significant effect on the response. In the first case networks with different number of hidden neurons (h) are taken as a part of the multi-net architecture. Three networks with 3, 4 and 5 hidden neurons are considered as shown in figure 1. All the three networks are trained with a constant batch size of 5. Figure 2 shows the multi-network output plotted against outputs of the individual networks. In the second case the hidden neurons in three networks are maintained constant as 4. The batch sizes of data for training are different in each of the three networks. Batch sizes greater than 10 seemed to take large time to train and are not capable of predicting the system response in real-time. Batch sizes of 3, 4 and 5 are considered for analysis. The outputs from the individual networks and the dynamic selection block of the multi-net is shown in figure 3.

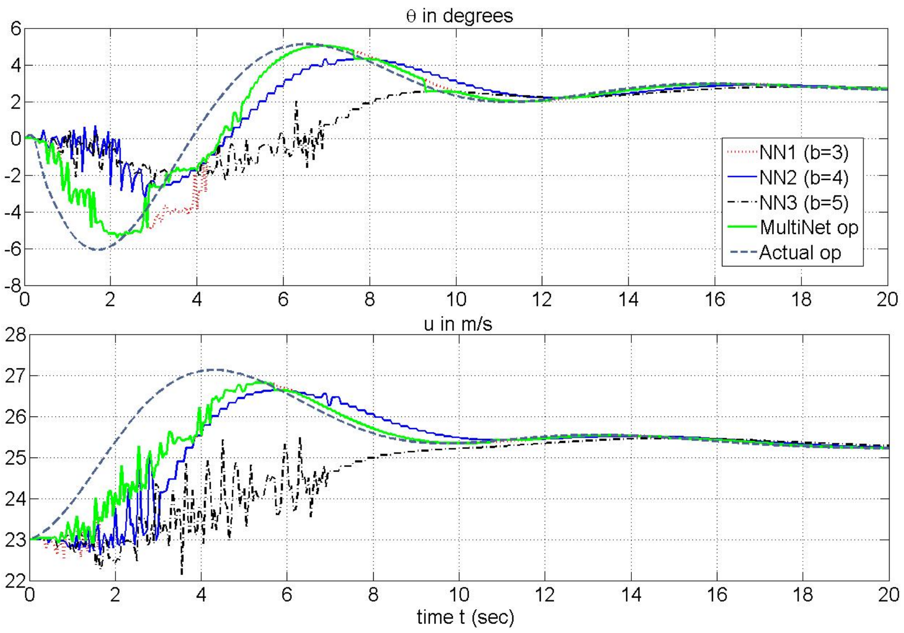


Fig. 3. Comparison of Multi-network output with 4 hidden neurons against the system output

4 Experiments and Results

The online training is performed using small fixed batches of input and output data. Larger batch sizes tend to take longer time to train and hence seize the purpose of online implementation. The available training period is less than or equal to the sampling period due to one step prediction provided by the network. In this study a MIMO longitudinal model of the UAV is considered. The forward velocity (u) and pitch angle (θ) are considered as outputs and elevator and throttle as inputs.

The MSE values for two different types of multi-network models are presented in tables 1 and 2. It can be seen in both the cases the multi-network models have lower MSE values as compared to any of the individual networks. The results from individual networks plotted with the multi-nets are shown in figures 2 and 3. The outputs from the ARX model considered depends on a single past input and output. The variation in the dynamics is measured in terms of the error between the previous output and predicted output obtained from the trained network. As long as the dynamics does not vary beyond a threshold the weights of this trained network are retained in memory. The network is retrained only when the error from the new batch is greater than the predefined threshold.

As a typical case study 20 different sets of initial random weights are considered for a single network with 4 hidden neurons and a batch size 5. The standard deviations, variance and mean square errors in the prediction of the

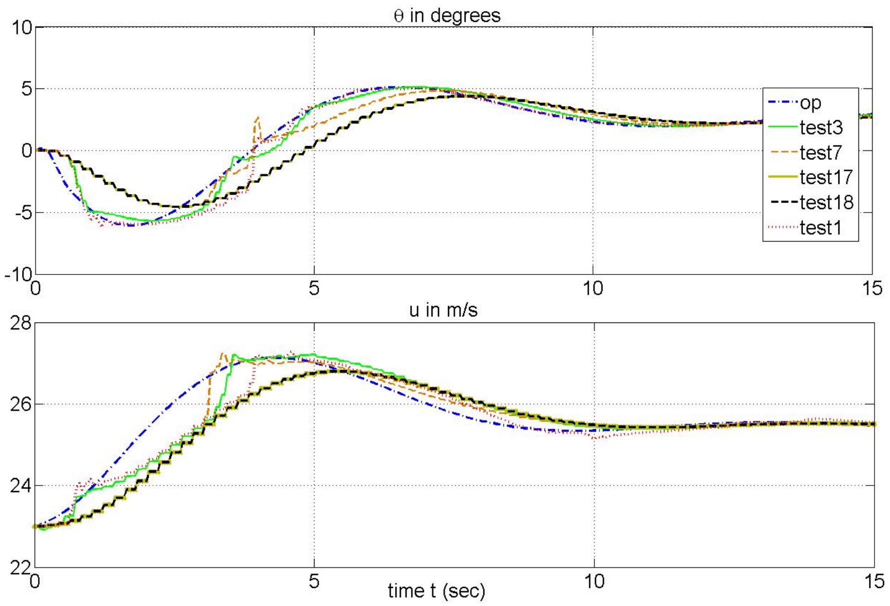


Fig. 4. Comparison of outputs obtained from different random initial weights

Table 3. Comparison of performance of a single network with different initial weights

Test	Std Dev u	Std Dev θ	Variance u	Variance θ	MSE u	MSE θ
1	0.1502	0.4351	0.0226	0.1893	0.0228	0.1895
2	0.2409	0.2349	0.058	0.0552	0.0584	0.0552
3	0.1120	0.1569	0.0126	0.0246	0.0127	0.0246
4	0.564	0.3129	0.3181	0.0979	0.3201	0.0982
5	0.4132	1.0525	0.1707	1.1078	0.1718	1.1087
6	0.0638	0.0938	0.0041	0.0088	0.0042	0.0088
7	0.1243	0.3026	0.0155	0.0915	0.0162	0.0918
8	0.1463	0.4348	0.0214	0.189	0.022	0.1892
9	0.1501	0.4353	0.0225	0.1895	0.0227	0.1897
10	0.1502	0.4351	0.0226	0.1894	0.0227	0.1895
11	0.3169	0.8069	0.1004	0.6511	0.1012	0.6451
12	0.1267	0.4344	0.0161	0.1887	0.0162	0.1892
13	0.1277	0.2695	0.0163	0.0726	0.0173	0.0731
14	0.2926	0.6241	0.0856	0.3896	0.087	0.3903
15	0.3259	0.8011	1.7581	0.6418	1.7696	0.6464
16	0.4847	1.5965	0.235	2.549	0.2355	2.5755
17	0.1504	0.4351	0.0226	0.1894	0.0228	0.1895
18	0.1503	0.4353	0.0226	0.1895	0.0228	0.1896
19	0.1496	0.4352	0.0224	0.1894	0.0226	0.1895
20	0.1504	0.4351	0.0226	0.1893	0.0228	0.1895

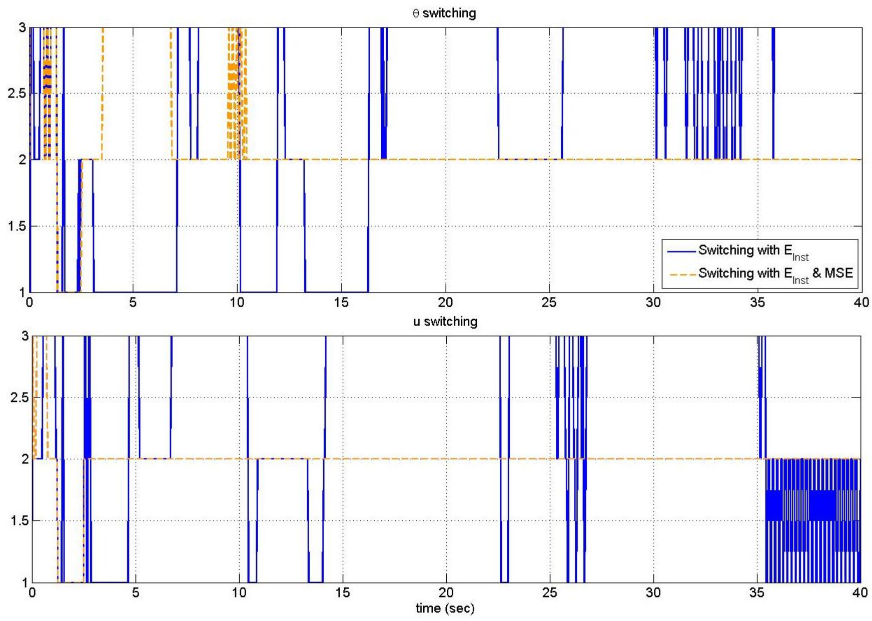


Fig. 5. Comparison of switching for two selection criteria

forward velocity and the pitch angle for each of the simulation is tabulated in table 3. Figure 4 shows the response of this network for some of these weights superimposed on each other. It can be seen that the output response is different each time the initial weights are changed. The test run 6 shows least errors as compared to the other tests.

Two different selection criteria are tested in the dynamic selection block of the multi-network with fixed batch size. In the first case the instantaneous error between the present output and the predicted output is used for switching. The resultant output from the multi-network continuously switches between the three networks. As an improvement in the second case a weighted average between the instantaneous error and past MSE for each of the networks is considered as the selection criteria for the switching. This technique allows the selection of network with the most consistent past and present performance. A comparison of the two switching is shown in figure 5. The figure indicates one of the three networks active for each of the outputs. It can be seen that the switching is reduced to a large extent when the past performance is considered. The past performance of each of the networks can be monitored and the training of networks with consistent poor performance can be suspended and hence providing scope for better training of the other networks.

5 Conclusions

The identification of the MIMO UAV is presented using a 'Black-box' technique based on autoregressive models and neural networks. An adaptive multi-neural network architecture with online training and dynamic selection was proposed. The results from variations in the hidden layer size and the batch length were compared. It was observed that in each case the multi-network outputs are more accurate compared to any of the individual networks. The effect of changes in the initial weights on the performance was also compared. It was seen that even though the networks with different initial weights had similar steady state response, their response during the initial training phase differed. At present, research to design and implement robust controllers for autonomous flight of the UAV using the multi-net architecture is being undertaken.

References

1. Giovanini, L., Benosman, M., Ordys, A.: Adaptive control using multiple models switching and tuning (2005)
2. Johnson Anthony, M.D., Calise, J., Johnson, E.N., Corban, J.E.: Applications of adaptive neural-network control to unmanned aerial vehicles. In: AIAA/ICAS International Air and Space Symposium and Exposition: The Next 100 Years, Dayton, OH, AIAA (2003)
3. Fujinaka, S., Omatu, T.: A switching scheme for adaptive control using multiple models (1999)
4. Hu, C., Huang, X., Hu, J., Zhu, J.: System identification of a small uav's speeding up process before take-off 21-07-2006 (2004)
5. Janardhan, V., Schmitz, D., Balakrishnan, S.N.: Development and implementation of new nonlinear control concepts for a ua. In: 23rd Digital Avionics Systems Conference, vol. 2, pp. 12.E.5– 121–10 (2004)
6. Ljung, L.: System Identification - Theory for the User, 2nd edn. Prentice-Hall, Englewood Cliffs (1999)
7. Marquardt, D.W.: An algorithm for least-squares estimation of nonlinear parameters. SIAM Journal on Applied Mathematics 11(2), 431–441 (1963)
8. Narendra, J., Balakrishnan, K.S.: Adaptive control using multiple models. IEEE Transactions on Automatic Control 42(2) (1997)
9. Puttige, V., Anavatti, S.: Real-time neural network based online identification technique for a uav platform. In: CIMCA 2006. International Conference on Computational Intelligence for Modelling, Control and Automation, Sydney, IEEE Computer Society Press, Los Alamitos (2006)
10. Puttige, V., Anavatti, S.: Comparison of real-time online and offline neural network models for a uav. In: International Joint Conference on Neural Networks, Orlando, U.S.A., IEEE Computer Society Press, Los Alamitos (2007)
11. White, A., Howitt, S., Richards, D.: Supervisory control of autonomous uavs in networked environments. In: AIAA InfotechAerospace, Arlington, Virginia, AIAA (2005)
12. Selmic, F.L., Lewis, R.R.: Multimodel neural networks identification and failure detection of nonlinear systems (2001)

Prediction of Polysomnographic Measurements

S.I. Rathnayake and Udantha R. Abeyratne

School of Information Technology and Electrical Engineering,St Lucia, Brisbane, Queensland 4072, Australia
suren@itee.uq.edu.au, udantha@itee.uq.edu.au

Abstract. During polysomnography, multivariate physiological measurements are recorded, and analysed to identify episodes of breathing disorders occur during patients sleep for the diagnosis of sleep disordered breathing disorders. Measurement distortions, such as signal losses that may occur due to loosening of a sensor, are often present in these measurements. Reliability and accuracy of automated diagnostic procedures using polysomnographic data can be improved through automated identification and recovery of such measurement distortions. In this study is an attempt towards that focusing on the respiratory measurements. Respiratory measurements are a main criterion in assessing sleep disordered breathing episodes. Treating respiratory system as a deterministic dynamic system, functional mapping that exists between two state space embeddings are approximated using artificial neural networks. Performance of the trained neural networks in identification of measurement distortions and measurement recovery are reported.

1 Introduction

Sleep Disordered Breathing Diseases (SDBD) are highly prevalent through out the world. Study conducted in the US reported 9% of the female and 24% of the male adult population aged between 30 to 60 show at least mild sleep disordered breathing (SDB) conditions [1]. Another study conducted in India showed 19.5% of asymptomatic males coming for routine health checks suffer from mild SDB conditions [2]. Untreated SDBD can cause serious health complications including cardiovascular disease such as Hypertension, cognitive dysfunctions [3]. SDBD patients often suffers from un-refreshed sleep and fatigue, which can lead to accidents [4].

The gold standard test for diagnosing SDBD is Polysomnography (PSG). During a typical PSG session over 15 physiological measurements are recorded during period of 6 to 8 hours of patients sleep. Measurements recorded during PSG are analysed to determine performance of breathing, brain, cardiovascular system, and muscle movements during sleep. PSG is a high resource consuming test which requires both specialised technicians and set up. Screening methods [5,6] and automatic SDBD diagnosis methods [9,5] have been investigated to reduce both the demand for PSG and the human labour associated with PSG. However, PSG can over estimate SDB event as shown in an study by Villa et al [9].

Presences of measurement distortions in physiological measurements can lead incorrect classifications of SDB episodes. Signal loss, which can occur due to loosening of the sensor, and a sleep apnea episode, which causes complete cessation of air flow over 10 seconds, show identical recording on a respiratory measurement channel. During manual analysis of SDB events, measurement distortion in one channel can be identified using simultaneously observed measurements of undistorted measurements. For accurate automated analysis of PSG data, such distortions needs to be detected and recovered.

Our main focus in this study is on distortions in respiratory measurements. Measurements on respiratory function observed during PSG are used in diagnosis [7] of the highest prevalent SDBD, named Obstructive Sleep Apnea-Hypopnea Syndrome. The physiological measurements commonly used in assessing SDB events are respiratory flow measurements; usually the Nasal Airflow measurement or Nasal Pressure measurement [7,8]. Thus we study predictions between Nasal Pressure measurement (NPM) and Nasal Airflow measurement (NAM) in this study.

Regulation of breathing is mainly done by the Central Nervous System and chemoreceptors. During inspiration Central Nervous System sends a rhythmic signal to the diaphragm and the intercostal muscles to contract. This causes negative pressure inside the lungs forcing air in. As the signal sent from the Central Nervous System ceases, contracted mucels relaxes causing air to exit from the lungs. Peripheral chemoreceptors regulates breathing based on concentration of CO_2 and O_2 in the blood, and mechanical events in heart, lungs or anywhere in the body [10]. Central chemoreceptors, reside in the brain stem, respond to changes in the Hydrogen ions or CO_2 concentrations of cerebral spinal fluid or blood. Chemoreceptors also modulates the heart rate depending on the CO_2 concentration [10]. When the environmental aspects that can affect breathing does not change during the period of PSG recordings, we can treat that the respiratory dynamics are coming from an autonomous system. We further assume, considering the dynamics of breathing, breathing is governed by a deterministic dynamic system.

Takens [11] embedding theorem states that, under certain smoothing conditions, it is possible to embed the state space of a deterministic dynamic system using a scalar measurement time series. Thus when multivariate measurements are obtained from a dynamic system, each time series separately can be used to embed state space. The functional mapping between thus obtained state spaces exists, which can be used in predicting one measurement from another. Thus, in this study we investigate on the prediction of one respiratory flow measurement from another as a time series prediction task in order to identify measurement redundancies.

2 Theory

2.1 Embedding and Re-embedding Respiratory Dynamics

In modeling the human biological sub-system that generates breathing as continuous deterministic dynamic system, we assume that the state space of the of the dynamics is in a sooth manifod M of d dimensional space. Transition from state s(t_1) at time t_1 to its state at time t_2 of a dynamic system is governed by a the flow function φ such that s$(t_2) = \varphi(t_2 - t_1, s(t_1))$. Flow function is differentiable with respect to time and obeys properties of group and identity. Thus knowledge of the state at one time and the flow will give information to determine the state of the system at anytime.

An arbitrary PSG measurement $x_l(t_i)$ recorded at time t_i of the state variable s_l is related by,$x_l(t_i) = h_l(s(t_i))$. With $\{t_i | i = 1, 2, ...N\}$ for some integer N, we can obtain a time series $\{x_l\}$. Provided we know h_l and φ_l, we can reconstruct the state space of the respiratory system. In this case, we are not aware of φ and h_l, thus our intention is to obtain a state space reconstruction for the purpose of predicting future measurements.

Using time delayed samples of measurements obtained, we construct delay vector $\underline{x}_l$(t) as,$\underline{x}_l(t) = (x_l(t), x_l(t - \tau), \ldots, x_l(t - \tau(m_l - 1)))$. The delay vector, $\underline{x}_l$(t) is a mapping of d dimensional dynamic system into m_l dimension. It contains information of both present and the past of the system. This mapping is equivalent to $\phi_l(s)$ where,

$$\phi_l(s) = (h_l(s(t)), h_l(\varphi_l(\tau, s(t))), \ldots, h_l(\varphi_l(-\tau m_l, (s(t))))). \qquad (1)$$

Takens [11,12] showed that ϕ_l is an embedding when $m_l \geq 2d + 1$, under general conditions on the dynamic system and the h_l be a C^2 differentiable observation function. Embedding is a smooth one to one and onto mapping with smooth inverse. Smooth dynamics $\mathcal{F}$ is induced on the space of reconstructed vectors, as (from Equation 1),

$$\mathcal{F}(\tau, \underline{x}_l) = \phi_l(\varphi_l(\tau, \phi_l^{-1}(\underline{x}_l)). \qquad (2)$$

Thus we can predict the time evolution of the measurements, as $x_l(t + \tau) = \mathcal{F}_l(\tau, \underline{x}_l)$.

If Ω_l is the reconstructed state space from $\{$x$_l\}$, then the mapping ϕ_l:$M \rightarrow \Omega_l$ is a diffeomorphism with properties of both one to one and onto mapping. Using another measurement function h_k, measurements time series $\{$x$_k\}$ can be obtained by measuring the same or another state variable of the respiratory dynamic system. With the time series $\{$x$_k\}$ we can embed the respiratory dynamics in Ω_k induced by mapping a ϕ_k up to diffeomorphism.

When both Ω_l and Ω_k are diffeomorphic to M, thus there exists mappings $\phi_{l,k}$ which maps from Ω_l to Ω_k, and its inverse. Thus the measurements of both $\{$x$_l\}$ and $\{$x$_k\}$ can be predicted from each other, such that

$$\mathcal{F}_{lk}(\tau, \underline{x}_l) = \phi_k(\varphi_k(\tau, \phi_l^{-1}(\underline{x}_l)) \quad for \quad l, k = 1, 2. \qquad (3)$$

Thus the prediction of measurement $x_k(t+\tau) = \mathcal{F}_{lk}(\tau,\underline{x}_l$(t)).

The ability to obtain one step (step size τ) prediction implies that the possibility of obtaining multi-step predictions. In summery our aim is to obtain the model of functional relationship $\mathcal{G}_{ij}$ which can be defined as,

$$\hat{x}_k(n + \lambda) = \mathcal{G}_{lk}(x_l(n), x_l(n - \tau_l), \ldots, x_l(n - (m_l - 1)\tau_l)) \qquad (4)$$

The model $\mathcal{G}$ is used here to estimate $\mathrm{x}_k(\lambda)$ as $\hat{\mathrm{x}}_k(\lambda)$ where λ is an integer multiple of τ_l. If $l = k$, we expect to reconstruct the state space of dynamic system using a measurement for the purpose predictions of future measurements. For the case where $l \neq k$, $\mathcal{G}$ is a mapping function from the reconstructed state spaces by $\{\mathrm{x}_l\}$ to the reconstructed state space from $\{\mathrm{x}_k\}$. The non-zero positive integers m_k and τ_k are termed as the embedding dimension, and delay time of measurement time series $\{\mathrm{x}_k\}$. If $l = k$ then $\lambda > 0$ or $\lambda < -(m_j\text{-}1)\tau_j$ is necessary for practical modelling.

2.2 Embedding Parameters *m* and *τ*

Diffeomorphism implies preservation of topological properties with one to one mapping between two spaces. If the state space reconstruction is done with $m \geq m_0$, where m_0 is the minimum embedding dimension required for diffeomorphism, then neighbouring points in the original state space will map to neighbouring points in the reconstructed state space. Thus if $\underline{x}(t_r,m)$ and $\underline{x}(_p,m)$ where $\underline{x}(t_r,m) = (x(t_r), x(t_r\text{-}\tau), \ldots, x(t_r\text{-}\tau(m\text{-}1)))$ are neighbours when embedded in the dimension m and $m \geq m_0$, both $\underline{x}(t_r,m+1)$ and $\underline{x}(_p,m+1)$ will also be neighbours in $m+1$ embedded dimension. However, if $m < m_0$, $\underline{x}(t_r,m+1)$ and $\underline{x}(_p,m+1)$ will not necessary be neighbours in $m+1$. This method known as false nearest neighbour method proposed by Kennel et al [16] is used to estimate embedding dimension.

In order to identify the distiance between two neighbour when embedded in high dimensio, the ratio *(NRD)* is computed as,

$$NRD = \left| \frac{\underline{x}(t_r, m+1) - \underline{x}(t_p, m+1)}{\underline{x}(t_r, m) - \underline{x}(t_s, m)} \right| \tag{5}$$

The neighbors whose *NRD* exceeds a given heuristic threshold are taken as a false nearest neighbors. The minimum m for which fraction of false nearest neighbors are close to zero is taken as the minimum embedding dimension of the time series.

Takens theorem does not provide nor specifies conditions of τ. If the measurements are noiseless, any value of τ is suitable for the reconstruction of the state space. However, due to presence of noise a number of methods are used to obtain a optimal time delay. If a small delay time is used, the reconstructed state space will be densely populated close to each other. If a higher τ, over a certain limit may again can cause successive elements in the delay vector to get correlated [15]. Thus aiming to de-correlate the delay vector, for selection of τ is Autocovariance Function (ACF) and Mutual Information Criteria (MIC) can be used. ACF is zero when there is no correlation is present, thus the duration for the first zero crossing is taken as the delay time when ACF is used. MIC, unlike ACF it takes non-liner correlations also into account. When MIC is used, the delay time is taken as the duration for first significant minimum where the information correlations is negligible.

ACF of a time series {x} with $\bar{x}$ and var(x) as the mean and the variance can be given as,

$$ACF(\omega) = \frac{\sum_{n=0}^{N-\omega}(x(n)-\bar{x})(x(n-\omega)-\bar{x})}{(N-\omega)(var(x))}, \tag{6}$$

MIC is calculated from the relation,

$$MIC(\omega) = \sum_{i}^{W}\sum_{j}^{W}(p_{ij}(\omega)log\frac{p_{ij}(\omega)}{p_i p_j}) \tag{7}$$

where for some W partition on the real numbers p_i is probability that a time series value falls into the i^{th} partition, and the probability that two observation made ω apart falls in to both i^{th} partition and j^{th} partition is given as $p_{ij}(\omega)$.

2.3 Modeling $\mathcal{G}_{lk}$

For the prediction, the continuous mapping function $\mathcal{G}_{lk}$ needs to be identified from the relationship between $x_k(n+\lambda)$ and $(x_l(n), x_l(n-\tau_l), \ldots, x_l(n-(m_l-1)\tau_l))$. Hornik et al [13] has showed that in approximating an arbitrary mapping from one finite dimensional space to another, single hidden layer with arbitrary sigmoidal hidden layer activation functions is sufficient when a feedforward neural network are used. In this study, multi-layer fully connected Neural Networks (NNs) with hidden units having hyperbolic tangent sigmoidal transfer functions are used to model $\mathcal{G}_{lk}$, the mapping function that predicts a measurement $\{x_l(n)\}$ from $\{x_k(n)\}$.

Long-term prediction of deterministic dynamic systems are discussed using neural networks with time delays in studies such as [14]. In this study, long term predictions are of no immediate interest. Thus easy to implement, fully connected Multilayer Perceptron neural networks are selected. During training, Levenberg-Marquardt backpropagation technique was used to update the bias terms and weights of the neurons. This algorithm converges rapidly with properties of Taylor series, and can reach to solutions outside of the initial conditions with properties of gradient decent algorithm.

However, in order for an NNs to be able to approximate a functional relationship, the structure of the NN should be capable of handling such relationship, and also the data provided for training the NN should contain the relationship. If the architecture of the NN is very large, it may learn high dimensional noise components present in the data.

We have used the sliding window method shown in Fig.1 to provide the input vector $\underline{x}_l(n)$ to the neural network training for the target x_k(n+τ). Output layer of the ANN consisted of a single neuron, while the number of neuron in the input layer, first hidden layer and second hidden layers are $\mathcal{H}0$, $\mathcal{H}1$ and $\mathcal{H}2$. All input layer neurons and output layer neuron are assigned bias of unity. The activation functions for the output layer is a linear function and for the hidden layers hyperbolic tangent sigmoid transfer functions was used.

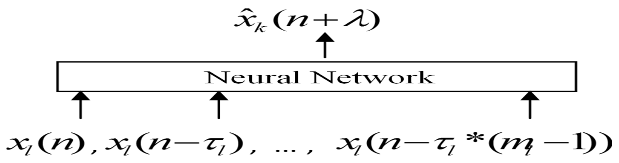


Fig. 1. Neural network for τ-step ahead prediction of {x2} from {x1}. The sliding windows are used to present inputs to the neural network to obtain an estimate x2(n+k) as x̂2(n+k).

2.4 Performance of Predictions

Performance of prediction errors were calculating using the Normalised Root Mean Squared Error (NRMSE) on *N+1* data samples.

$$NRMSE = \sqrt{\frac{\sum_{n=1}^{N}\{x(n+1) - \hat{x}(n+1)\}^2}{\sum_{n=1}^{N}\{x(n+1) - \overline{x}\}^2}} \tag{8}$$

where, $\hat{x}$ is the signal predicted and $\overline{x} = \sum_{n=1}^{N} x(n+1)$ is the mean of test sample x. Very low NRMSE implies that neural network has modeled the signal both in time and space. Higher value of NRMSE does not necessarily imply that the predictions are erroneous. It could be time or space shift in the predicted signal. Prediction error at peaks can also increase NRMSE.

3 Data Acquisition

The data used for this study was obtained from tests conducted for SDB using PSG. These tests were conducted at The Department of Respiratory and Sleep Medicine at the Princess Alexandra hospital, Brisbane, Australia. The routine PSG data were obtained from a clinical system (Model Siesta, Compumedic-sTM, Sydney, Australia). Signals acquired included EEG, EOG, ECG, EMG, leg movements, nasal airflow, nasal pressure, respiratory movements, blood oxygen saturation, breathing sounds and the body position. The routine PSG data are accompanied by an expert-edited annotation file with an event-by-event description of SDB and sleep stages. For this study, 5 PSG recordings of patients undergone for SDBD diagnosis are used. The Nasal Airflow and Nasal Pressure measurements are recorded 64 samples per second.

Nasal Pressure Measurement (NPM) is obtained by placing nasal prongs near the nasal openings of the patients to guide the airflow to a pressure transducer. The measurement provides the pressure variation at the nostrils with respect to the atmospheric pressure. Nasal Airflow Measurement (NAM) is obtained using a thermal sensor placed between nasal and mouth openings, such that respiratory flow contacts the sensor. Expired breathing flow of a human is at the body temperature while the temperature of the inspired airflow is at that of the room. Sleep laboratories record both the measurements in SDB diagnostic studies, which can be used to obtain accurate classifications of events.

4 Results and Discussion

4.1 Calculation of Embedding Parameters

For each patient, 30 second data segment prior to sleep on set is used for calculation of embedding parameters. Both ACF and MIC are used to calculate the delay time. Fig. 2a) shows ACF curves and Fig. 2b) shows MIC curves obtained for both NAM and NPM measurements for a one recording. We observed that for most of the patients τ is approximately 50 for both NAP and NPM measurements. Embedding dimensions m for the signals were calculated using FNNR routine implemented in the TISEAN [15] package. Fig. 2c) shows fraction of false nearest neighbours Vs embedding dimension for both NPM and NAM of one patient for minimum FNNR = 3. The curves are close to zero when embedding dimension used is greater than 5. In presence of noise, we find use of higher embedding dimension can introduce noise terms. Thus we select for both NAM and NPM m as 5.

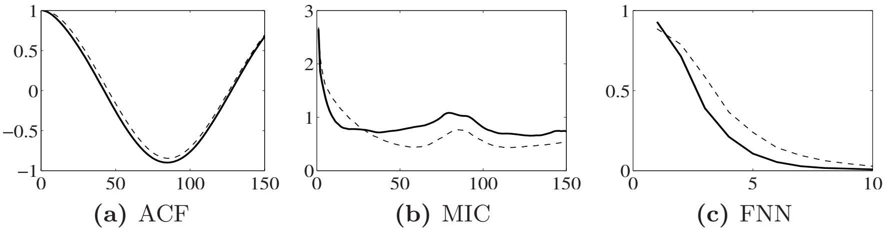


Fig. 2. Autocorrelation function, Mutual Information Criteria, function Fraction of false nearest neighbors for the PSG recording C on both Nasal Airflow and Nasal Pressure Measurements are shown in the figure from left to right respectively.

4.2 Training NNs

We have used m of 5 and tau 50 for both NAM and NPM cross prediction tasks. All $\mathcal{H}0$, $\mathcal{H}1$ and $\mathcal{H}2$ selected to be equal to the embedding dimension of the input time series (measurement) of each configuration of the NN. A data segment of 30 seconds duration recorded just before the sleep onset was selected from each PSG recordings to train the NNs. Fraction 3/4 of this data used to train the ANN, and the rest is used to measure the generalisation of the NNs while training. Though the fraction used in measuring the generalisaion of NNs are lower, we expect considerable amount of data is used. Mean Squared error is used to obtain training both training error and generalisation errors after each epoch of training. Once these error measures are calculated, a copy of the ANN is again trained.

NNs were trained for Maximum epoch number of 100, with early stopping criteria of mean squared training error is less than 1e-10 or gradient in the errors is less than 1e-5. However, we have observed that after around 10 epochs of training, both validation errors and training errors converge. In order to avoid over training, we have selected an NN trained to 10 epochs. None of the early stopping criteria we have set were not met before this.

4.3 Predictions

The prediction error *NRMSE* calculated for 30 second epoch during all 5 sleep stages (Stage 1 to 4 and REM) and a wakefulness is shown in the Fig. 3 in predicting NAM.

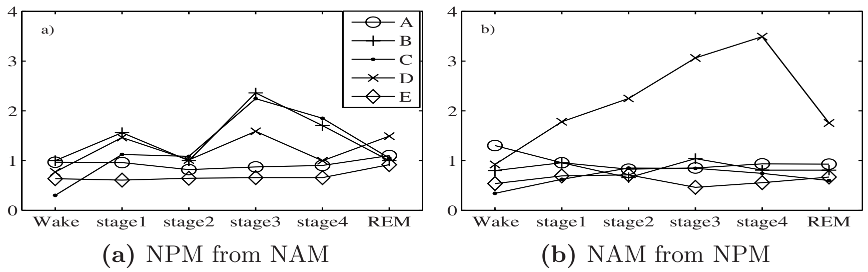


Fig. 3. The frame a) shows performance error in predicting Nasal Pressure measurement from the Nasal Airflow Measurement, vise verse in the frame b) for 5 PSG recordings. The recordings used for prediction in both figures are each patient were done simultaneously.

Fig. 4 shows an episodes of actual and predicted NAM with sudden large drop in measurement. In the actual NPM a sudden drop in is visible around 20 seconds, which we assume due to an artifact. Identification of the presence of artifacts can be done using the high NRMSE of prediction with low numebr of data point in calculating the NRMSE. Once identified, the contaminations can be eliminated by using the predicted measurement as a substitute of the contaminated segment.

When at least one measurement is distorted we expect high NRMSE values at prediction. However, variance of low amplitude measurements are small, thus prediction errors (NMSE) of low amplitude measurements get amplified. We believe its the reason for high NMSE during the prediction of PR from AF at stage 4 sleep.

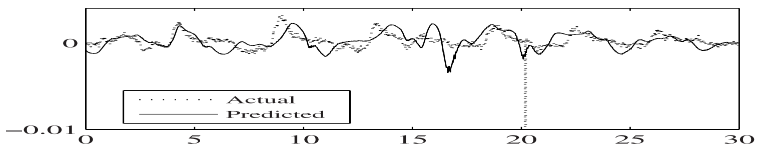


Fig. 4. Prediction of Nasal Pressure measurements from the Nasal Airflow measurement

Fig. 5 show that the signal loss in one channel can be detected using high prediction errors present in predictions. Once identified, lost measurement can be reconstructed from the un-distorted measurement. When a signal loss is present in a measurement say in x_l, the variance of that segment is small when compared with undistorted simultaneous measurement of another channel, unless an Sleep Apnea event is experienced during this time. Recording at a signal loss is close to the mean value of an measurement. Thus when a measurement loss is present, such as in Fig. 5, and measurements are predicted from each other, high NRMSE and a low NRMSE (around 1) will be obtained. This can be used to automatically identify measurement distortions such as signal loss. The high NRMSE (in predicting x_l from x_k) implies that the predicted measurement (x_l) may contain a signal loss.

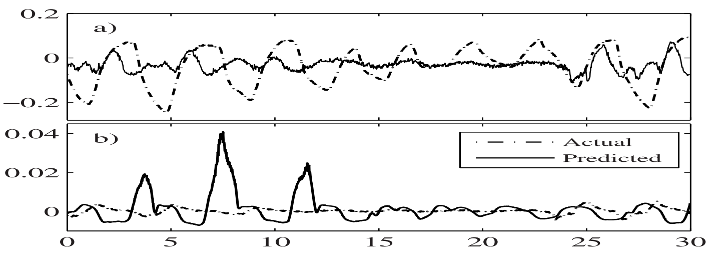


Fig. 5. The frame a) shows actual Nasal Airflow Measurement and the measurement predicted from Nasal Pressure measurement, and vise verse in the frame b). Prediction error NRMSE for Nasal Pressure measurement is 1.0 while for the Nasal Pressure measurement is 5.1.

5 Conclusion and Future Work

By treating the respiratory system to an autonomous dynamic system, we have modeled the functional relationship between the Nasal Pressure Measurement and Nasal Airflow Measurements. Multilayer Peceptron type neural networks are used to approximate the mapping function between them. We have used 5 polysomnography recordings of patients undergone for sleep disordered breathing diseases. We find results are very much encouraging toward automated classification of polysomnography data. In future studies, we expect to investigate on optimum neural network architecture and selections of training data for reliable respiratory signal predictions.

References

1. Young, T., Palta, M., Dempsey, J., Skatrud, J., Weber, S., Badr, S.: The Occurrence of Sleep-Disordered Breathing among Middle-Aged Adults. N. Engl. J. Med. 328, 1230–1235 (1993)

2. Udwadia, Z.F., Doshi, A.V., Lonkar, S.G.: Prevalence of sleep-disordered breathing and sleep apnea in middle-aged urban Indian men. Am. J. Respir. Crit. Care Med. 169, 168–173 (2004)
3. Babar, S.I., Quan, S.F.: Through the looking glass, the unrecognized tragedies of obstructive sleep apnea. Sleep Med., 299–300 (2002)
4. Barbé, F., Pericás, J., Muñoz, A., Findley, L., Antó, J.M., Agustí, A.G.N., de Lluc, J.M.: Automobile Accidents in Patients with Sleep Apnea Syndrome. An Epidemiological and Mechanistic Study. Am. J. Respir. Crit. Care Med. 158, 18–22 (1999)
5. Steltner, H., Staats, R., Timmer, J., Vogel, M., Guttmann, J., Matthys, H., Christian, J.: Diagnosis of Sleep Apnea by Automatic Analysis of Nasal Pressure and Forced Oscillation Impedance. Am. J. Respir. Crit. Care Med. 165, 940–944 (2002)
6. Epstein, L.J., Dorlac, G.R.: Cost-effectiveness analysis of nocturnal oximetry as a method of screening for sleep apnea-hypopnea syndrome. Chest 113, 97–103 (1998)
7. American Academy of Sleep Medicine Task Force: Sleep related breathing disorders in adults: recommendations for syndrome definition and measurement techniques in clinical research. Sleep 22, 667–689 (1999)
8. Farre, R., Montserrat, J.M., Navajas, D.: Noninvasive monitoring of respiratory mechanics during sleep. Eur. Respir. J. 24, 1052–1060 (2004)
9. Villa, M.P., Piro, S., Dotta, A., Bonci, E., Scola, P., Paggi, B., Paglietti, M.G., Midulla, F., Ronchetti, R.: Validation of automated sleep analysis in normal children. Eur. Respir. J. 11, 458–461 (1998)
10. Levitzky, M.G.: Pulmonary physiology. McGraw-Hill Medical, New York (2007)
11. Ruelle, D., Takens, F.: On the nature of turbulence. Commun. Math. Phys., 20,23, 167–192, 343-344 (1971)
12. Casdagli, M., Eubank, S., Farmer, D., Gibson, J.: State-space reconstruction in the presence of noise. Physica D 51, 52–98 (1991)
13. Hornik, K., Stinchombe, M., White, H.: Multilayer feedforward networks are universal approximators. Neural Networks 2, 359–366 (1989)
14. Han, M., Xi, J., Xu, S., Yin, F.-L.: Prediction of chaotic time series based on the recurrent predictor neural network. IEEE Trans. Signal Processing 52, 3409–3416 (2004)
15. Hegger, R., Kantz, H., Schreiber, T.: Practical implementation of nonlinear time series methods: The TISEAN package. In: CHAOS, vol. 9, p. 413 (1999)
16. Kennel, M.B., Brown, R., Abarbanel, H.D.I.: Determining embedding dimension for phase-space reconstruction using a geometrical construction. Phys. Rev. A. 45, 3403 (1992)

An Adaptive Approach for QoS-Aware Web Service Composition Using Cultural Algorithms

Ziad Kobti and Wang Zhiyang

Department of Computer Science, University of Windsor,
Windsor, Ontario, N9B-3P4, Canada
{kobti, wang11r}@uwindsor.ca

Abstract. Web service composition is the process of integrating existing web services. It is a prospective method to build an application system. The current approaches, however, only take service function aspect into consideration. With the rapid growth of web service applications and the abundance of service providers, the consumer is facing the inevitability of selecting the "maximum satisfied" service providers due to the dynamic nature of web services. This requirement brings us some research challenges including web service quality model, the design of web service framework monitoring service real time quality. The further challenge is to find the algorithm which can handle customized service quality parameters and has good performance to solve NP-hard web services global selection problem. In this paper, we propose an adaptive web service framework using an extensible service quality model. Evolutionary algorithms are adopted to accelerate service global selection. We report on the comparison between Cultural Algorithms with Genetic Algorithms and random service selection.

Keywords: cultural algorithms, genetic algorithms, web service composition, quality of service (QoS).

1 Introduction

A web service pulls the web functionality from document oriented to application oriented and forms a new standard of distributed computing known as Service Oriented Architecture (SOA). Business service developers are just to assemble a set of appropriate web services to implement the business tasks. Business applications are no longer written manually. For example [1], a client requirement can be described as a workflow like a sequence diagram in UML. The workflow is composed of several sub-functional modules or abstract services. Each abstract service is associated with a web service community which contains several concrete web services with the same functionality. The process of selecting a concrete service from a web service community for an abstract service by QoS attributes is called local selection. Obviously, a task presented by the service composition can be solved by a significant number of combinations. The process of selection from the numerous combinations according to the non-functional requirement is called global selection which is a NP-hard problem. Consequently, client requirements can be presented as web services global selection problem. In order to give a general approach for web service composition, we need a general quality model to present the quality of each available service. We also require the web service framework that has the functionality of real-time monitoring of the quality parameters. Moreover, we should have an efficient algorithm to globally select web services. In our work we employ evolutionary algorithms, namely Genetic Algorithms (GA) and Cultural Algorithms (CA), where the latter is used in population evolution models [2].

This paper is organized as follow. Section 2 reviews major challenges and mainstreams in QoS-aware web service composition. Section 3 presents our proposed approach. Section 4 gives the comparison results among different algorithms. The last section summarizes conclusions and identifies directions for future work.

2 Related Work: Quality Models and Related Algorithms

With the growing number of service providers and consumers, there are different interests and concerns with respect to different service quality attributes. It is common practice to use a simple approach to model and represent QoS, such as availability, accessibility, integrity, performance, reliability and price [3, 4, 5]. Another practice suggests mapping the objective system quality to the users' subjective perception of quality [6]. This introduces a new web service quality attribute termed objectively and proposes an approach to quantify it. The extensibility is endorsed as more important than the parameters used in the given quality model [7].

Many approaches have been investigated to implement service composition, such as, integer programming [8], Pinsinger's Algorithm, and Reduction Algorithm [9]. However, they can only deal with limited number of generic quality parameters. Studies [1] have adapted a Genetic Algorithm (GA) to globally select web services and analyze web service quality parameters, but, unfortunately a slow process. Therefore, based on that work, a GA alone is not suitable for use as an all purpose approach to compose web services.

Cultural Algorithms (CA) are population based evolutionary algorithms that use a global belief space with an influence function to accelerate the convergence of the population solution [2].

3 Proposed QoS Model

3.1 Proposed QoS Model for Component Services

Since the meaning of QoS attributes is different by a variety of end-user factors, contextual circumstances as well as the perspective of interest, each provider must unambiguously define its QoS model before delivering its QoS aware service. In this paper, we use four typical quality attributes execution cost, response time, reliability and availability to model the extendable quality of web services. The brief explanation of each attribute is as follows:

Execution cost: The execution cost of an operation of a service is the amount of money that a service requester has to pay for executing the operation. Web service providers either directly advertise the execution cost of their operations, or they provide means to enquire about it.

Execution duration: The execution duration measures the maximum delay in seconds between the moment when a request is sent and the moment when the results are received by client-side view. The execution duration wraps the complex transmission factor into account. It includes service response time and network transmission latency. Service response time is the maximum seconds that elapses from the moment that a web service receives a legitimate SOAP request until it produces the corresponding legitimate SOAP reply. The execution duration expression is $q_{duration} = t_{rep}(s,op) + t_{trans}(s,op)$, meaning that the execution duration is the sum of the response time $t_{rep}(s,op)$ and the transmission time $t_{trans}(s,op)$. Services advertise their response time or provide methods to enquire about it. The transmission time is estimated based on past executions of the service operations, e.g. $t_{trans}(s,op) = \sum_{i=1}^{n} t_i(s,op)/n$, where $t_i(s,op)$ the past observation of the transmission time, and n is the number of execution times observed in the past.

Reliability: the reliability of a service is the probability that a request is correctly responded within a maximum expected time frame (which is published in the web service description). In the thesis, reliability measures the degree of compliance between providers claimed value with the actual value. Reliability is a technical measure related to hardware and/or software configuration of web services and network connections between the service consumers and providers. The value of reliability is computed from historical data about past invocations using the expression $q_{rel}(s) = N_c(s)/k$, where $N_c(s)$ is the number of times that the service s has been successfully delivered within the maximum expected time frame, and k is the total number of invocations.

Availability: Availability is the quality aspect of whether the web service is present or ready for immediate use. The availability $q_{av}(s)$ of a service is the probability that the service is accessible. In the thesis, the value of the availability of a service is computed using the following expression $q_{av}(s) = t_a(s)/k$, where $t_a(s)$ is the total amount of time (in seconds) in which service is available during the last k seconds (k is a constant). The value of k may vary depending on a particular application. In applications where services are more frequently accessed, a small value of k gives a more accurate approximation for the availability of services. If the service is less frequently accessed, using a larger k value is more appropriate. In the thesis, we give a set of constants corresponding to the invocation times in a month. E.g. invocation times < 1000 k=1000; invocation times >= 1000 k=800; etc.

In conclusion, the quality vector $Q(s) = (q_{\cos t}(s), q_{duration}(s), q_{rel}(s), q_{av}(s))$ is used to represent the quality of a service in the thesis. This proposed QoS model also supports extended custom QoS attribute as long as customers give its unambiguous definition and computing approach.

3.2 Proposed QoS Model for Composite Service

The same quality attributes described in 3.1 are also applied to evaluate the quality of composite services. In order to simplify the computation, a composite service, first of all, will be unfolded and only composed of a set of sequential component services, no loops and conditions. Namely, we can consider a composite web service containing several concretized abstract services with sequential structure. For example, a composite service can be defined as *S = {AS1, AS2, ASi …}* and each component *ASi* must be concretized. For brevity, the explanation of the approach to unfold a complex composition workflow is omitted from this paper, but the reader may refer to [10] and [8] for further details.

3.3 Computation of the QoS of Composite Service

Table 1 provides the aggregation functions of computing the QoS of a composite service. A brief explanation is given as follows: Execution cost - the execution cost of an unfolded composite service is the sum of every component service. For instance, for a workflow containing a conditional branch, with cost *C1* and *C2* and probabilities *p* and *1-p*, the overall cost is *p*C1 + (1-p)*C2*. For a workflow containing a Loop, with cost *C1* and n times iteration, the overall cost is *n*C1*. Execution duration - the execution duration of an unfolded composite service is the sum of every component

Table 1. QoS aggregation function of different workflow structure

QoS Attributes	Sequential	Conditional	Parallel	Loop
Execution Cost (*C*)	$\sum_{i=1}^{n} C(t_i)$	$\sum_{i=1}^{n} p_i * C(t_i)$	$\sum_{i=1}^{n} C(t_i)$	$k * C(t)$
Execution Duration (*D*)	$\sum_{i=1}^{n} D(t_i)$	$\sum_{i=1}^{n} p_i * D(t_i)$	$Max\{T(t_i)_{i\in\{1..n\}}\}$	$k * D(t)$
Reliability (*R*)	$\prod_{i=1}^{n} R(t_i)$	$\sum_{i=1}^{n} p_i * R(t_i)$	$\prod_{i=1}^{n} R(t_i)$	$R(t)^k$
Availability (*A*)	$\prod_{i=1}^{n} A(t_i)$	$\sum_{i=1}^{n} p_i * A(t_i)$	$\prod_{i=1}^{n} A(t_i)$	$A(t)^k$
Custom Attributes (*F*)	$f_S(F(t_i)_{i\in\{1..m\}})$	$f_C(p_i, F(t_i)_{i\in\{1..m\}})$	$f_P(F(t_i)_{i\in\{1..m\}})$	$f_L(k, F(t_i)_{i\in\{1..m\}})$

service. Reliability - the reliability of a composite service is the multiplication of the reliability of each component service. Availability - the availability of a composite service is the multiplication of the availability of each component service.

4 Experiment Setup and Results

The goal of our experiments is to search for Web Service Composition (WSC) solutions which maximize availability and reliability and also minimize execution Cost and execution Duration by Random service selection, GA and CA approaches.

4.1 Simulation Process and Initialization

The initial parameters of the experiments include the number of available web services, the size of abstract services, the quality properties of web services, population size and the number of generations. A large population size of 700 is used so that we can rapidly reach to the convergent point. The fixed number of generations is 40. The initial service quality parameters are compliant with better execution duration and availability offers corresponding higher price. In this experiment, we consider a simple workflow including 5 distinct abstract services which include 3 to 15 available services respectively. Tables 2 and 3 show the samples of service QoS and abstract service (i.e. service community) description in the experiment. In the experiment, we assume a complex service workflow has been unfolded and stored as Table 4.

Table 2. Web service description samples

Service						
serviceName	**rspMax**	**rspAvg**	**price**	**Reliability**	**availability**	**comName**
S1	100	63	100	0.627	0.544	AS1
S2	500	451	30	0.598	0.532	AS1
S3	400	318	30	0.653	0.584	AS1
S4	30	13	300	0.955	0.956	AS2
S5	80	54	160	0.592	0.527	AS1
S6	140	68	70	0.937	0.929	AS2
S7	130	92	65	0.595	0.529	AS3

Table 3. Abstract service description samples

Community	
communityName	**Members**
AS1	S1,S3,S2,S5,S15,S36,S37,S38,S39,S40
AS2	S10,S6,S4,S16,S33,S34,S35
AS3	S7,S9,S17,S18,S28,S29,S30,S31,S32
AS4	S8,S11,S19
AS5	S14,S13,S12,S20,S21,S22,S23,S24,S25,S26,S27

Table 4. Unfolded service workflow

Task	
taskName	**process**
TASK1	AS1,AS5,AS2,AS3,AS4
TASK2	AS3,AS4,AS6,AS7

According to the purpose of the experiment, the solutions must satisfy the quality requirements, maximum *availability* and *reliability*, minimum *execution cost* and *execution Duration*. Therefore, our problem can now be modeled by means of a fitness function which maximizes *reliability* and *availability*, while minimizing *execution cost* and *execution duration*. The fitness function is defined for a genome *g* as follows:

$$F(g) = \frac{w_1\ Availability + w_2\ Reliability}{w_3\ ExecutionCost + w_4\ ExecutionDuration}$$

where $w1, \ldots, w4$ are real, positive weighting factors. Many different approaches have been proposed in literatures to deal with this kind of fitness function. In general, calibrating weights is guided by observing the fitness function's landscape, as well as from the analysis of the evolution of the different factors. In our experiment, in order to simplify the calculation, we define the fitness function as follows:

$$F(g) = \frac{Availability + Reliability}{ExecutionCost + ExecutionDuration} * 1000$$

In the above fitness function, the four quality parameters are considered having the same importance, so they have the same weight factor. The calibrating coefficient 1000 is to make the fitness values easily analyze. In our problem, clearly, the convergent point of fitness function is inconstant because the service quality properties are unpredictable. However, we can identify the convergent point at which a number of generations with no apparent improvement in the solution set.

4.2 Random Service Selection

Mainly serving as an experimental control, a random service generator is devised to build a workflow each generation from a randomly selected service from each service community.

4.3 Genetic Algorithm

To use GA search for WSC solutions, we first need to have a suitable genome to present the problem. In our case, the genome is represented by an ArrayList with a number of items equals to the number of distinct *abstract services* composing our service. Each item contains a HashMap of the *concrete services* matching that *abstract service*. Figure 1 gives the idea of how the genome is made.

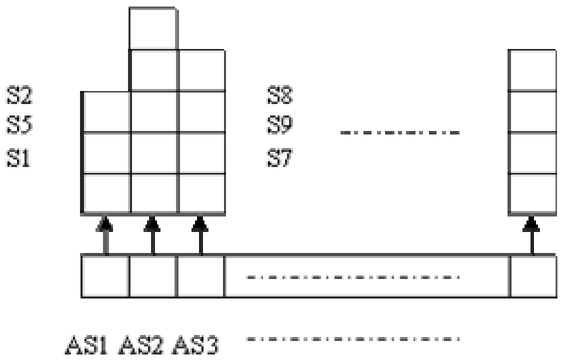


Fig. 1. Genome model for the GA

The crossover operator is the standard two-point crossover. The crossover point is randomly selected according to the length of the genome. The mutation operator randomly selects an *abstract service* (a position in the genome) and randomly replaces the corresponding *concrete service* with another one among those available. Service selection operator picks up the individuals which have best fitness values in current generation as the parents of next generations. The stop criterion is a certain number of generations reached.

4.4 Cultural Algorithm

In order to use a CA into our experiment, we select the same genome model as the above GA to present our problem. The same method in GA is also used to evaluate the individuals in the population space. In our experiment, the Belief Space is maintained by two knowledge sources including situational and historical knowledge. Situational knowledge selects the best individuals, or exemplars, into the Belief Space. Historical knowledge picks up the most recently used individuals into the Belief Space. The individuals in the Belief Space are associated as parents in the Population Space. In the population space, parents are first selected from the Belief Space to generate and evaluate the individuals for the next generation. The individuals in the population space have two paths to be accepted into Belief Space: if the fitness is better than the individuals in Belief Space, or if the most recent selected individuals. The Belief Space is updated by selecting the certain number of best individuals.

4.5 Experimental Results

Figure 2 shows the evolution of average quality parameters and fitness among 40 generations. Apparently, the convergent point of the average fitness is very low by random service selection approach. In other words, random selection approach is not ideal approach to search for “maximum satisfied” WSC solution.

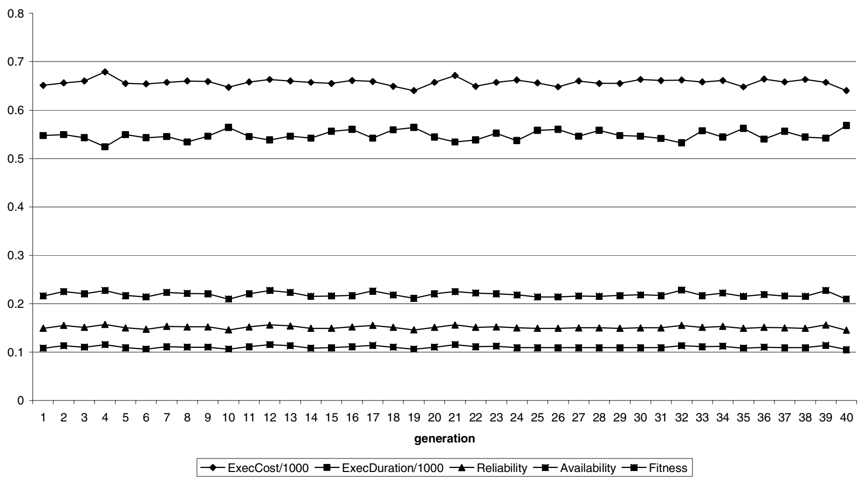


Fig. 2. Evolution of quality and fitness parameters by Random service selection approach

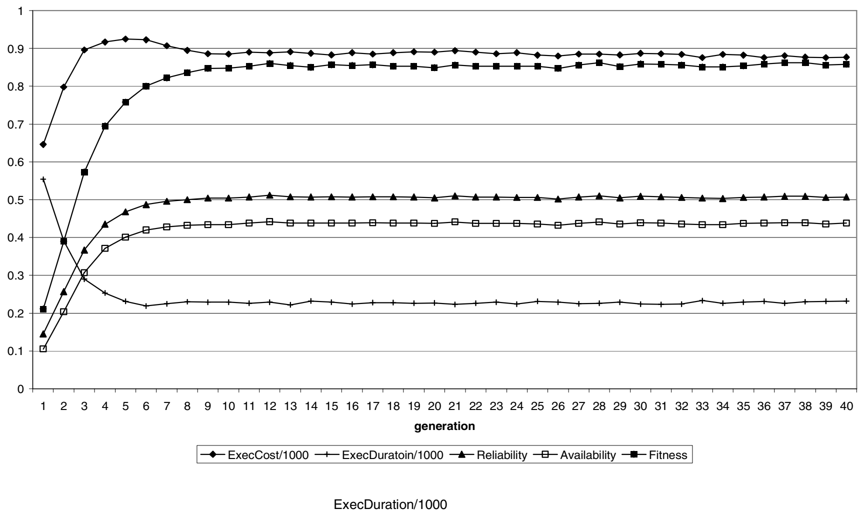


Fig. 3. Evolution of quality parameters and fitness by GA service composition approach

Figure 3 plots the *Execution Cost, Execution Duration, Reliability, Availability* and *Fitness* evolution track across GA generations. From this figure, the average fitness is converged at the central value 0.85 ± 0.01 from the 10th generation. In other words, it takes 10 generations to reach to the convergent point 0.85 ± 0.01 in GA WSC approach. The results of the fitness of various quality parameters using the CA are shown in Figure 4 while Figure 5 compares the overall fitness between Random, CA and GA revealing the improved performance of the CA. This comparison showing the fitness value achieved and its increase is summarized in Table 5.

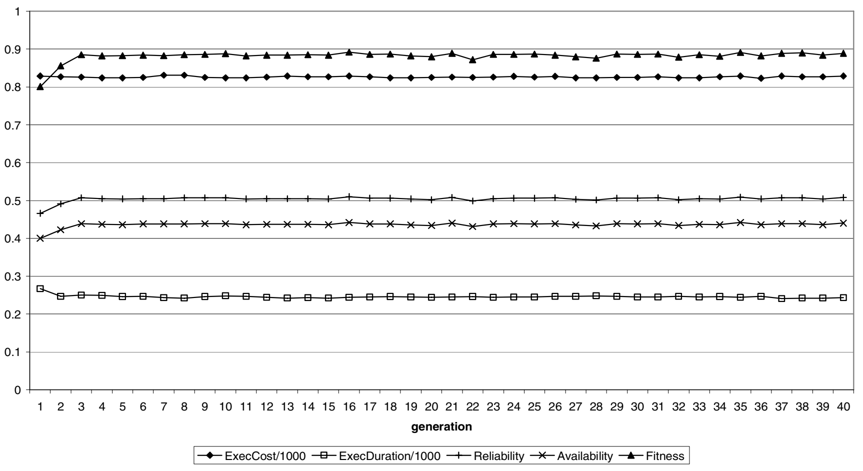


Fig. 4. Evolution of quality parameters and Fitness by CA service composition approach

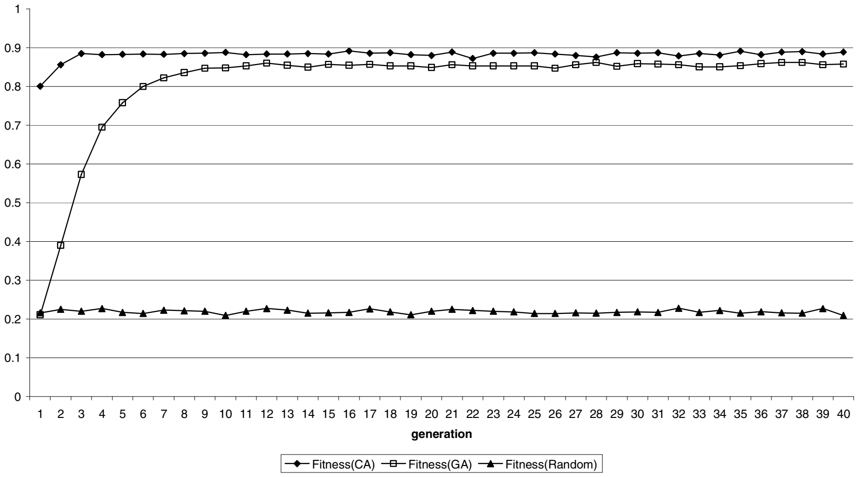


Fig. 5. Comparison of fitness evolution of Random, GA and CA

Table 5. Comparison of fitness among Random service selection, GA and CA WSC approaches

Approaches	#of Runs	# of Generations	Fitness	Fitness Increase
Random	5	N/A	0.2 ± 0.02	N/A
GA	5	11	0.85±0.01	76.5%
CA	5	3	0.885±0.007	77.4%

5 Conclusion and Future Work

This paper proposes a framework using evolvable agents to monitor QoS in web service composition. According to the results reporting on a limited set of quality parameters a near optimal solution was reached in a reasonable amount of generations using a Cultural Algorithm, slightly surpassing the performance of a GA.

Future work will focus on additional knowledge structures in the Belief Space of the CA and investigate further quality parameters in the evaluation process.

References

1. Canfora, G., Penta, M.D., Esposito, R., Vilanni, M.L.: An Approach for QoS-aware Service Composition based on Genetic Algorithms; Genetic And Evolutionary Computation Conference. In: Proceedings of the 2005 conference on Genetic and evolutionary computation, pp. 1069–1075. ACM Press, New York (2006)
2. Reynolds, R.G., Peng, B.: Cultural Algorithms: Modeling of how cultures learn to solve problems. In: Proceedings of the 16th IEEE International Conference on Tools with Artificial Intelligence, IEEE Press, Los Alamitos (2004)
3. Lau, T.C.: QoS for B2B Commerce in the New Web Services Economy. In: Kou, W., Yesha, Y., Tan, C.J.K. (eds.) ISEC 2001. LNCS, vol. 2040, Springer, Heidelberg (2001)
4. Jin, L.: Analysis on Service Level Agreement of Web Services; Software Technology Laboratory, HP Laboratories Palo Alto, Technical Paper HPL-2002-180 (June 2002)
5. Tian, M., Gramm, A., Naumowicz, T., Ritter, H., Schiller, J.: A concept for QoS integration in web services. In: Proceedings of the First Web Services Quality Workshop at WISE, Rome, Italy, December
6. Kalepu, S., Krishnaswamy, S., Loke, S.W.: Verity: a QoS metric for selecting web service and providers. In: Web information systems engineering workshops, Proceedings 4th international conference, pp. 131–139 (2004)
7. Gouscos, D., Kalikakis, M., Georgiadis, P.: An Approach to Modeling Web Service QoS and Provision Price. In: the 4th international conference on web information systems engineering workshops, pp. 121–130. IEEE Press, Los Alamitos (2004)
8. Zeng, L.Z., Benatallah, B., Dumas, M., Kalagnanam, J., Seng, Q.Z.: Quality driven web services composition; International World Wide Web Conference. In: Proceedings of 12th international conference on World Wide Web Session: Web engineering, pp. 411–421. ACM Press, New York (2003)
9. Cardoso, J., Sheth, A., Miller, J., Arnold, J., Kochut, K.: Quality of service for workflows and web service processes. Journal of Web Semantics (2004)
10. Canfora, G., Penta, M.D., Esposito, R., Vilanni, M.L.: A lightweight approach for QoS-aware service composition. In: Proceedings of the 2nd International Conference on Service Oriented Computing, pp. 36–47 (2004)

A Genetic Programming Approach to Extraction of Glycan Motifs Using Tree Structured Patterns

Masatoshi Nagamine[1], Tetsuhiro Miyahara[1], Tetsuji Kuboyama[2], Hiroaki Ueda[1], and Kenichi Takahashi[1]

[1] Graduate School of Information Sciences,
Hiroshima City University, Hiroshima 731-3194, Japan
{nagamine@rea.its, miyahara@its, ueda@its, takahasi@its}.hiroshima-cu.ac.jp
[2] Center for Collaborative Research, The University of Tokyo, Tokyo 153-8505, Japan
kuboyama@ccr.u-tokyo.ac.jp

Abstract. We propose a genetic programming approach to extraction of glycan motifs by using tag tree patterns, which are tree structured patterns with structured variables. A structured variable in a tag tree pattern can be substituted by an arbitrary tree. Our experiments show that we have obtained tag tree patterns as the best individuals including similar substructures of glycan motifs obtained by the previous works.

1 Introduction

Glycans are said to be the third major class of biomolecules next to DNA and proteins. They are known to be crucial for the development and function of multi-cellular organisms. They are polysaccharide structures, or carbohydrate structures, often forming tree structures, as opposed to the linear structure of DNA and proteins.

As the glycan data g in Fig. 1 shows, the structure of a glycan is abstractly represented as a tree by representing single sugars as vertices and covalent bonds between sugars as edges. A vertex label and an edge label denote a name of a sugar and a name of a bond, respectively.

In glycan structure analysis, extraction of common characteristic structures is important [5,8]. By a *glycan motif* we mean a common characteristic glycan structure specific to a phenomenon. To formulate a glycan motif, we use an *ordered tag tree pattern* [9], or simply a *tag tree pattern*, as a rooted tree pattern with ordered children and structured variables. A variable in a tag tree pattern can be substituted by an arbitrary tree. By a *motif tag tree pattern* we mean a tag tree pattern which represents a glycan motif. Since the formulation of tag tree patterns and genetic operations on them is based on edge labels, we regard each vertex label as the prefix of the label assigned to the edge adjacent to the vertex. The tree T in Fig. 1 shows the tree structure corresponding to the glycan data g.

Genetic Programming (GP) [1,7] is Evolutionary Computation [3,11] and widely used as a search method for evolving solutions from structured data. GP is shown to be useful for evolving highly structured knowledge [12,6].

So learning of the structural features, called glycan motifs in Bioinformatics, of glycan data with respect to a specific phenomenon is very suited for genetic programming approach. Two previous works [5,8] on learning of glycan motifs are known. They use subtree structures as representations of glycan motifs and kernel methods for learning. A tag tree pattern is a whole tree structured pattern which matches a whole structure of an example tree structure, and has rich expressiveness of a structured variable representing any subtree structure.

By the *support* of a tag tree pattern we mean the ratio of examples explained by the pattern over all examples. The purpose of our previous works using GP [12,6] is to find characteristic tag tree patterns with support of 1.0 and specific structural features from Web documents. The fitness and GP operators of this work are different from those of our previous works, and are suited to extraction of motif tag tree patterns from glycan data.

Our experiments show that we have obtained tag tree patterns as the best individuals including similar substructures of glycan motifs obtained by the previous works. Our method successfully obtains motif tag tree patterns of leukemia.

This paper is organized as follows. In Section 2, we introduce tag tree patterns as tree structured patterns. In Section 3, we define our data mining problem of extraction of glycan motifs and present our method using Genetic Programming for solving the problem. In Section 4, we report experimental results on glycan data.

2 Tag Tree Patterns as Tree Structured Patterns

In this section, we briefly review tag tree patterns [9] which are used for representing characteristic tree structured patterns.

Let $T = (V_T, E_T)$ be a rooted tree with ordered children (or simply a **tree**) which has a set V_T of vertices and a set E_T of edges. Let E_g and H_g be a partition of E_T, i.e., $E_g \cup H_g = E_T$ and $E_g \cap H_g = \emptyset$. And let $V_g = V_T$. A triplet $g = (V_g, E_g, H_g)$ is called a **term tree**, and elements in V_g, E_g and H_g are called a *vertex*, an *edge* and a *variable*, respectively. A variable in a term tree can be substituted by an arbitrary tree. We assume that every edge and variable of a term tree is labeled with some words from specified languages. Λ and X denote a set of edge labels and a set of variable labels, respectively, where $\Lambda \cap X = \phi$. By regarding a variable as a special type of an edge, we can introduce the parent-child and the sibling relations in the vertices of a term tree. We use a notation $[v, v']$ to represent a variable $\{v, v'\} \in H_g$ such that v is the parent of v'. We assume that all variables in H_g have mutually distinct variable labels in X. A term tree with no variable is considered to be a tree.

A *substitution* θ on a term tree t is an operation which identifies the vertices of a variable x_i with the vertices of a substituted tree g_i, and replaces the variables

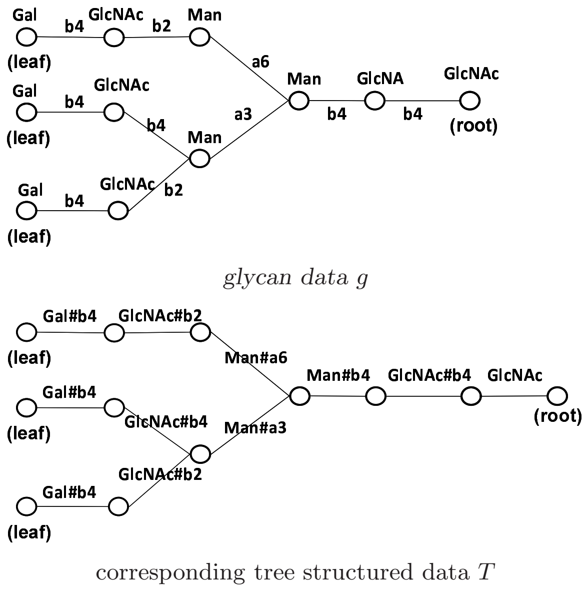


Fig. 1. Glycan data and corresponding tree structured data

x_i with the trees g_i, simultaneously. We assume that the parent of a variable is identified with the root of a substituted tree, and the child of a variable is identified with a leaf of a substituted tree. By $t\theta$ we denote the term tree which is obtained by applying a substitution θ to a term tree t.

For example, Fig. 2 shows a term tree t and trees t_1, t_2. In figures, a variable is represented by a box with lines to its parent and child vertices. The label inside a box is the variable label of a variable. The term tree t is defined as follows. $V_t = \{u_1, u_2, u_3, u_4, u_5, u_6, u_7\}$, $E_t = \{\{u_1, u_2\}, \{u_3, u_4\}, \{u_4, u_5\}, \{u_5, u_6\}\}$, $H_t = \{h_1, h_2\}$ with variables $h_1 = [u_2, u_3]$, $h_2 = [u_3, u_7]$, $\Lambda = \{GlcNac$, $Man\#a6$, $GlcNac\#b2$, $Gal\#b4\}$, $X = \{x, y\}$. Consider the substitution θ which identifies the vertices u_2, u_3 with the vertices v_1, v_2 and identifies the vertices u_3, u_7 with the vertices v_3, v_4 and then replaces the variables h_1 and h_2 with the trees t_1 and t_2 respectively. The term tree $t\theta$ is obtained by applying the substitution θ to the term tree t.

A **tag tree pattern** is a term tree such that each edge label on it is a keyword or a special symbol "?", which is a wildcard for any edge label. A tag tree pattern with no variable is called a *ground tag tree pattern*. Consider an edge e with an edge label L of a tag tree pattern and an edge e' with an edge label L' of a tree. We say that the edge e *matches* the edge e' if (1) L and L' are the same keyword, or (2) L is "?" and L' is any keyword. A ground tag tree pattern π *matches* a tree T if π and T have the same tree structure and every edge of π matches its corresponding edge of T. A tag tree pattern π **matches** a tree T if there exists a substitution θ such that $\pi\theta$ is a ground tag tree pattern and $\pi\theta$

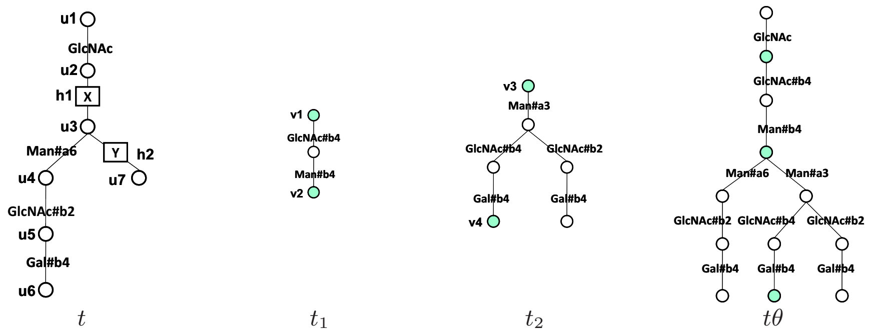


Fig. 2. A term tree t, trees t_1, t_2, and a term tree $t\theta$ obtained by a substitution θ

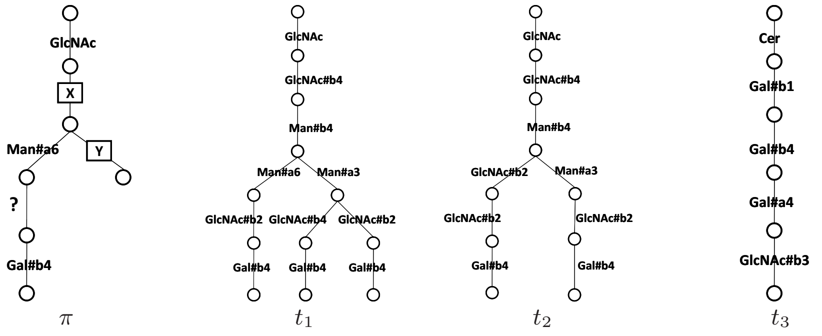


Fig. 3. A tag tree pattern π and trees t_1, t_2, t_3

matches T. For example, in Fig. 3, the tag tree pattern π matches the tree t_1 but does not match the trees t_2 and t_3.

3 A Genetic Programming Approach to Extraction of Glycan Motifs Using Tree Structured Patterns

The purpose of our previous works using GP [12,6] is to find characteristic tag tree patterns with support of 1.0 and specific structural features from Web documents. But the purpose of this work is to find motif tag tree patterns, which have lower support and stronger structural features than desired patterns of the previous works. So the fitness and GP operators of this work are different from those of our previous works, and are suited to extraction of motif tag tree patterns from glycan data.

3.1 Our Data Mining Problem

In this paper, we consider the following data mining problem.

Problem of Extraction of Glycan Motifs:
Inputs: A finite set $\mathcal{D}$ of positive and negative glycan data (or trees) with respect to a specific phenomenon.
Problem: Find a tag tree pattern π which represents a glycan motif of the specific phenomenon. Such a tag tree pattern π is called a *motif tag tree pattern* and has a specific structural feature w.r.t. the specific phenomenon.

We give the following GP-based method for the Problem of Extraction of Glycan Motifs by using tag tree patterns. (1) Determine a finite set of keywords KW, which is used in tag tree patterns, based on the set $\mathcal{D}$. (2) Generate the initial population of tag tree patterns in a random way from KW. (3) Evaluate the fitness of each individual (or tag tree pattern) in the population. (4) Select individuals with probability in proportion to fitness. (5) Perform genetic operations such as crossover, mutation and reproduction on selected individuals, and generates individuals of the next generation. (6) If the termination criterion is fulfilled, then terminate the whole process. Otherwise return to (3).

3.2 Fitness

By using background knowledge on structures of glycan data and preliminary experiments, we adopt the following definition of the fitness of a tag tree pattern.

Let π be a tag tree pattern and $\mathcal{D} = \{T_1, T_2, \cdots, T_m\}$ a finite set of positive and negative tree structured data w.r.t. a specific phenomenon. num_P denotes the total number of positive examples T_i in $\mathcal{D}$, and num_N denotes the total number of negative examples T_i in $\mathcal{D}$. $match_P(\pi)$ denotes the total number of positive examples T_i which π matches in $\mathcal{D}$, and $no\_match_N(\pi)$ denotes the total number of negative examples T_i which π does not match in $\mathcal{D}$. The **support**, which is the ratio of explained examples over all examples, $supp(\pi)$ of π w.r.t. $\mathcal{D}$ is defined as follows [12]:

$$supp(\pi) = \frac{1}{2} \times (\frac{match_P(\pi)}{num_P} + \frac{no\_match_N(\pi)}{num_N}).$$

The degree of edges, $edge\_deg(\pi)$, of π is defined as the sum of the assigned values of each edge or variable. The assigned value of an edge with a keyword is 1.0, the assigned value of an edge with "?" (a wildcard for edge labels) is 0.5, and the assigned value of a variable is 0.2. $size(\pi)$ denotes the total number of edges or variables in π. The ratio of specific edges, $edge\_ratio(\pi)$, of a tag tree pattern π is defined as $edge\_ratio(\pi) = edge\_deg(\pi)/size(\pi)$.

The **specificness** of π, denoted by $spec(\pi)$, and the **fitness** of π w.r.t. $\mathcal{D}$, denoted by $fitness(\pi)$, are defined as follows

$$size\_adjust(\pi) = \begin{cases} size(\pi)/10 & (0 < size(\pi) < 10) \\ max\{2 - size(\pi)/10, 0\} & (size(\pi) \geq 10) \end{cases}$$

$$spec(\pi) = \frac{e^{edge\_ratio(\pi)} - 1}{e - 1} \times size\_adjust(\pi)$$

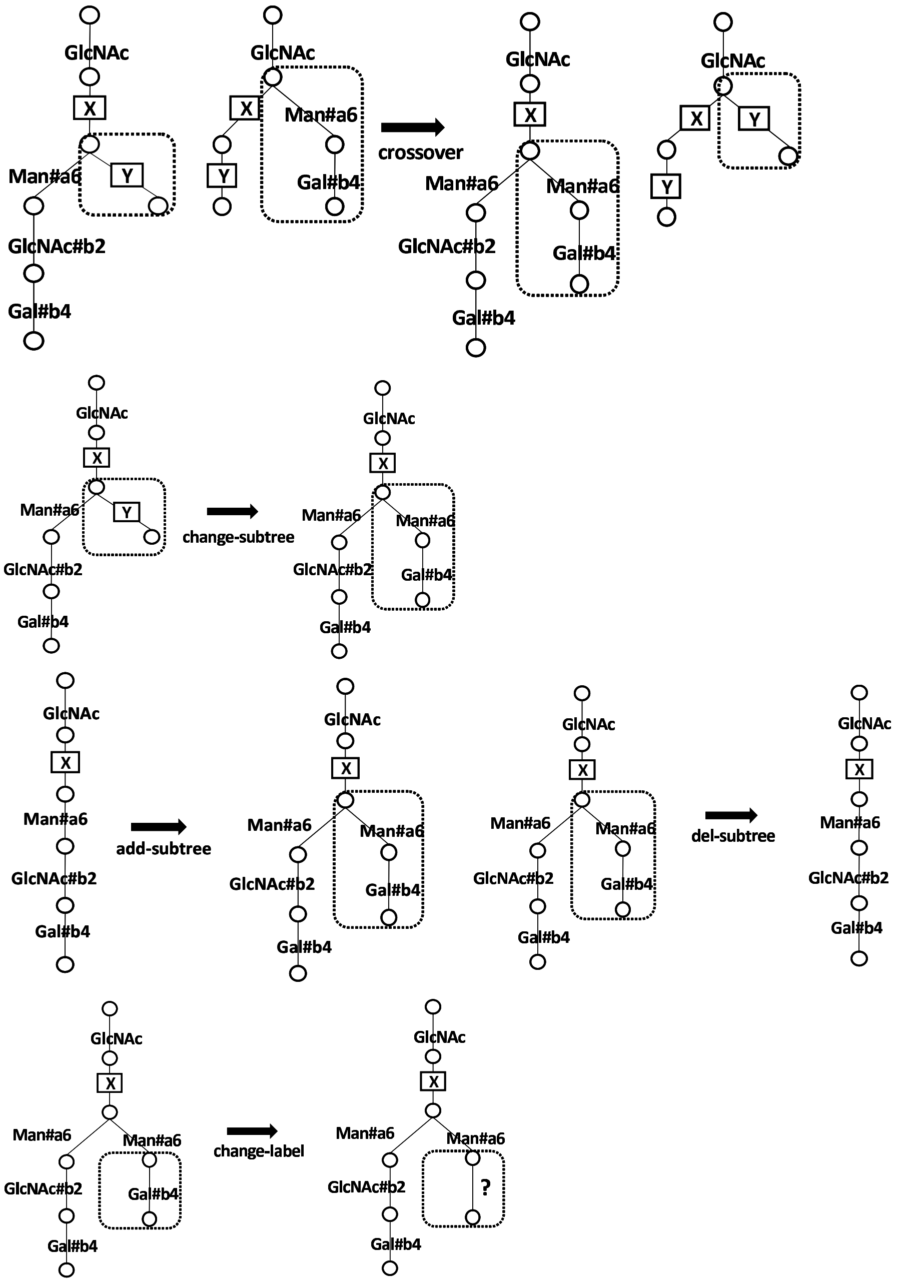


Fig. 4. Genetic operators on tag tree patterns

$add\_prob(\pi)$ denotes $\log_e((e-1) \times supp(\pi)+1)$. We add $spec(\pi)$, at a probability of $add\_prob(\pi)$, to $supp(\pi)$, then we have $fitness(\pi)$.

For example, consider the the tag tree pattern π and the set of trees $\mathcal{D} = \{t_1, t_2, t_3\}$ in Fig. 3. In the setting that t_1 and t_2 are positive examples, and t_3 is a negative example, we have $supp(\pi) = 0.75$. Also we have $edge\_deg(\pi) = 3.9$, $size(\pi) = 6$, $edge\_ratio(\pi) = 0.65$, $size\_adjust(\pi) = 0.6$, $spec(\pi) = 0.32$, $add\_prob(\pi) = 0.83$. If we add $spec(\pi)$ to $supp(\pi)$, then we have $fitness(\pi) = 1.07$, otherwise we have $fitness(\pi) = 0.75$.

3.3 Generation of Tag Tree Patterns

Glycan data w.r.t. a specific phenomenon have some common tree structure consisting of keywords. If we adopt a general algorithm for generating tag tree patterns as individuals, the algorithm generates tag tree patterns with keywords of inappropriate parent-child relation. Such a tag tree pattern does not match positive examples w.r.t. the specific phenomenon. In order to avoid generating such inappropriate tag tree patterns, we use the following generation algorithm [12]. (1) Extract a tree structure consisting of keywords from positive tree structured data in $\mathcal{D}$. (2) Generate only tag tree patterns conforming with the extracted tree structure in the processes of generating initial individuals.

We probabilistically apply the following 5 genetic operators. Since glycan data have a relatively small number of children per vertex, we do not use an inversion operator. In order to improve local search, we use 4 mutation operators (2)-(5). Fig. 4 illustrates these genetic operators. (1) **crossover:** Choose two individuals as parents, select a subtree in each parent, and change the selected subtrees between the two parents. (2) **change-subtree:** Replace a subtree with a newly generated subtree in an individual. (3) **add-subtree:** Add a newly generated subtree at a node in an individual. (4) **del-subtree:** Delete a subtree at a node in an individual. (5) **change-label**: Replace a specific edge label or "?" or a variable label to another arbitrary label.

4 Experiments

4.1 Experimental Setting

We have implemented our GP-based method for extracting of glycan motifs using tag tree patterns. The implementation is by Java on a PC with clock 3.0GHz on Windows XP. Our program uses the algorithm [10] for determining whether a tag tree pattern matches a tree or not.

In our experiments, we used glycan structure data comparable to the previous works [5,8]. We retrieved glycan structure data from the KEGG/GLYCAN data base [4] and their annotations from CarbBank/CCSD database [2]. We used the data of four blood components, *leukemic cells, erythrocyte, serum* and *plasma.*

Table 1. Parameters of GP

population size	50	selection	elite tournament
probability of reproduction	0.1		with size 4
probability of crossover	0.5	termination criterion	none
probability of mutation	0.4	maximum number of generations	200

Table 2. The fitness, support and specificness of experiments

	Leukemia			Erythrocyte		
	best	average	worst	best	average	worst
fitness	1.412	1.253	1.198	1.403	1.242	1.120
support	0.834	0.826	0.820	0.503	0.504	0.505
specificness	0.578	0.426	0.378	0.900	0.739	0.613
consumed time (sec)	6395	7645	8956	7241	7685	8280

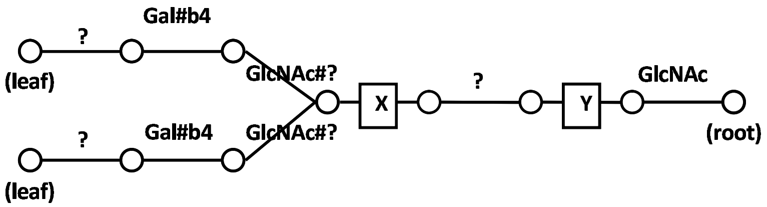
In the first experiment, we regarded the data related to leukemia as positive data and the data not related to leukemia as negative data. Then we have 176 positive data and 304 negative data w.r.t. leukemia. Similarly in the second experiment, we have 164 positive data and 316 negative data w.r.t. erythrocyte. This experimental setting is same as the previous works.

4.2 Experimental Results

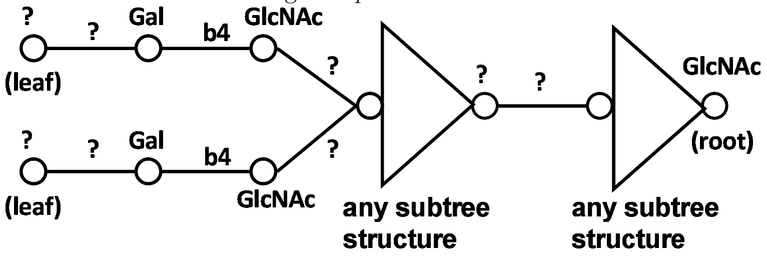
We have 10 GP runs for each experiment. Table 1 shows the parameters of our GP setting. The *best* (resp. *worst*) *run* means the run with best (resp. worst) fitness at the final generation (200th generation) out of 10 runs. Table 2 shows the fitness, support, specificness and overall consumed time (sec) of the best and worst runs at the final generation. "average" means the average value over 10 runs at the final generation.

The graph in Fig. 5 shows development of the fitness, support and specificness of the best run over generations. "average" means the average value at each generation of the fitness, support and specificness over 10 runs.

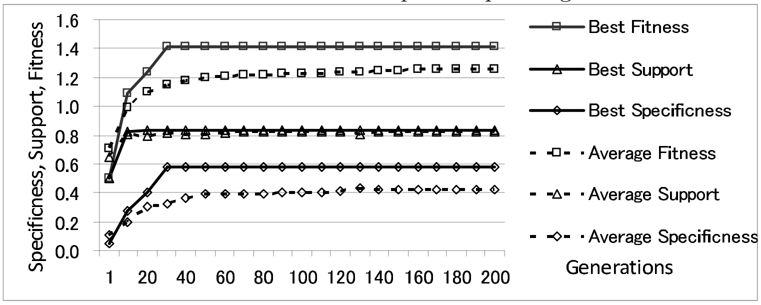
The tag tree pattern π in Fig. 5 shows the best individual at the final stage of the best run in the experiment w.r.t. leukemia. In the figures a special symbol "?" means a wildcard for any edge or vertex label. The whole tree structure p in Fig. 5 corresponding to π includes substructures similar to glycan motifs obtained by the previous works[5,8]. So the tree structured pattern p and the tag tree pattern π are considered a glycan motif and a motif tag tree pattern of leukemia. Since a tag tree pattern is a whole tree structured pattern which matches a whole structure of an example tree structure, we can say that our method successfully obtains a frequent global common tree structure, which is considered a glycan motif of leukemia.



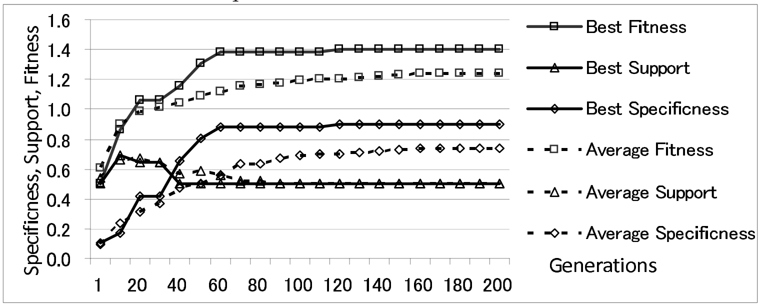
a motif tag tree pattern π of leukemia



the whole tree structure p corresponding to π



experimental result on leukemia



experimental result on erythrocyte

Fig. 5. The fitness, support, specificness of experiments and a motif tag tree pattern

5 Conclusions

In this paper, we have proposed a genetic programming approach to extraction of glycan motifs by using tag tree patterns, which are tree structured patterns with structured variables. Our experiments have shown that our method successfully obtains motif tag tree patterns of leukemia. Since we did not utilize specific knowledge about leukemia in our experiments, our method might be applicable to other glycan motifs. This work is partly supported by Grant-in-Aid for Scientific Research (C) No.19500129 from Japan Society for the Promotion of Science and Grant for Special Academic Research No.7116 from Hiroshima City University.

References

1. 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)
2. Doubet, S., Albersheim, P.: Clycobiology. Data Mining and Knowledge Discovery 2(6), 505 (1992)
3. Yao, X. (ed.): Evolutionary Computation: Theory and Applications. World Scientific, Singapore (1999)
4. Hashimoto, K., Goto, S., Kawano, S., Aoki-Kinoshita, K.F., Ueda, N., Hamajima, M., Kawasaki, T., Kanehisa, M.: KEGG as a glycome informatics resource. Glycobiology 16, 63R–70R (2006)
5. Hizukuri, Y., Yamanishi, Y., Nakamura, O., Yagi, F., Goto, S., Kanehisa, M.: Extraction of leukemia specific glycan motifs in humans by computational glycomics. Carbohydrate Research 340, 2270–2278 (2005)
6. Inata, K., Miyahara, T., Ueda, H., Takahashi, K.: Evolution of characteristic tree structured patterns from semistructured documents. In: Sattar, A., Kang, B.H. (eds.) AI 2006. LNCS (LNAI), vol. 4304, pp. 1201–1207. Springer, Heidelberg (2006)
7. Koza, J.R.: Genetic Programming: On the Programming of Computers by Means of Natural Selection. MIT Press, Cambridge (1992)
8. Kuboyama, T., Hirata, K., Aoki-Kinoshita, K.F., Kashima, H., Yasuda, H.: A gram distribution kernel applied to glycan classification and motifextraction. Genome Informatics 17(2), 25–34 (2006)
9. Miyahara, T., Suzuki, Y., Shoudai, T., Uchida, T., Takahashi, K., Ueda, H.: Discovery of frequent tag tree patterns in semistructured web documents. In: Chen, M.-S., Yu, P.S., Liu, B. (eds.) PAKDD 2002. LNCS (LNAI), vol. 2336, pp. 341–355. Springer, Heidelberg (2002)
10. Suzuki, Y., Akanuma, R., Shoudai, T., Miyahara, T., Uchida, T.: Polynomial time inductive inference of ordered tree patterns with internal structured variables from positive data. In: Kivinen, J., Sloan, R.H. (eds.) COLT 2002. LNCS (LNAI), vol. 2375, pp. 169–184. Springer, Heidelberg (2002)
11. Tan, K.C., Lim, M.H., Yao, X., L. (eds.): Recent Advances in Simulated Evolution and Learning. World Scientific, Singapore (2004)
12. Watanabe, A., Miyahara, T., Takahashi, K., Ueda, H.: Application of genetic programming to discovery of characteristic tree structured patterns (in Japanese). IEICE Technical Report 101(502), 41–48 (2001)

Feature Construction and Dimension Reduction Using Genetic Programming

Kourosh Neshatian, Mengjie Zhang, and Mark Johnston

School of Mathematics, Statistics and Computer Science
Victoria University of Wellington, P.O. Box 600, Wellington, New Zealand
{kourosh.neshatian,mengjie.zhang,mark.johnston}@mcs.vuw.ac.nz

Abstract. This paper describes a new approach to the use of genetic programming (GP) for feature construction in classification problems. Rather than wrapping a particular classifier for single feature construction as in most of the existing methods, this approach uses GP to construct multiple (high-level) features from the original features. These constructed features are then used by decision trees for classification. As feature construction is independent of classification, the fitness function is designed based on the class dispersion and entropy. This approach is examined and compared with the standard decision tree method, using the original features, and using a combination of the original features and constructed features, on 12 benchmark classification problems. The results show that the new approach outperforms the standard way of using decision trees on these problems in terms of the classification performance, dimension reduction and the learned decision tree size.

1 Introduction

The quality of representation has a significant impact on the level of attainment in machine learning problems. Issues like feature selection, feature construction, dimension reduction and space transformation have always been big challenges. In some systems these are intrinsic parts of the learning process. For example, neural networks implicitly build new features based on the raw input data. Decision trees use feature selection in every step of the tree construction process.

The capability of genetic programming (GP) in dynamically building programs and expressions, based on an objective function, has made it an excellent choice for automatic construction of new features for those learners that can not do it by themselves. Recently there has been a new research trend in using GP for feature construction for decision trees. GP has been used in different classification scenarios such as a complementary tool in learning process of a decision tree [1], as part of problem solving [2,3,4], or as a pre-processing phase [5].

In terms of the interaction with the final learner, previous research on GP for feature construction and selection can be divided into two major categories: the wrapper approach and non-wrapper (or preprocessing) approach. In the wrapper approach [6], the final learner is used as an indicator for the appropriateness of the constructed features. In each step, the constructed features are fed into the classifier, then the classifier accuracy is used as a guide to rank features [7,8]. Some approaches use a structure consisting of several program trees in each chromosome [7] while others perform a cooperative coevolution for this purpose [8]. The number of constructed features is usually the same as the number of original features, which means the complexity of feature dimensions cannot be reduced. However, [2] tries to compensate for this deficiency by implicitly putting pressure to select the classifiers that use a smaller number of features. Because there is no guide to measure the performance of individual features, and every fitness evaluation involves a complete run of the classifier, the search process is very expensive. Moreover, as both the construction and the learning process are performed at the same time, the solutions tend to be specific to a particular type of classifier.

In the non-wrapper approach, the process of feature construction is performed as a preprocessing phase. Since no particular classifier is involved in evaluating the constructed features, the process is expected to be more efficient and the results are expected to be more general. However, because they need a rather problem independent indicator to determine the suitability of features, the design of a fitness function is more challenging. Past researchers have used information gain (IG) and information gain ratio (IGR) as their fitness function [5,9] for constructing features.

While existing non-wrapper methods enjoy efficiency and generality, they have a serious drawback. As functions like IG and IGR can only tell us which feature is better in splitting up the data instances, the process of evolution comes up with the best individual that can provide the best split point(s). This means that these methods can create only one single feature for every problem. Since using a single feature usually is not enough for appropriate classification, researchers feed the classifier with an augmented feature set which includes both the new constructed feature and all the original ones [5,9]. However, this leads to another drawback, that is, increasing the dimensions of the problem.

This paper aims to develop an approach to the use of GP for feature construction in classification with the goal of improving classification performance, reducing the feature dimensions of a classifier, and learning a small classifier. Instead of wrapping a particular classifier for single feature construction as in most of the existing methods, this approach uses GP to construct multiple (high-level) features from the original features, then these constructed features are used by decision trees for classification. Consequently, rather than using the classification performance as the fitness measure, the GP fitness function is designed based on the class dispersion and entropy. This approach will be examined and compared with the standard decision tree method using the original features and using a combination of the original features and constructed features on 12 benchmark problems. Specifically, we wish to address the following research issues:

- how feature construction can be properly measured;
- how a new fitness function can be constructed based on the class dispersion and entropy;

– whether this approach outperforms the decision tree method using the original features in terms of the classification performance;
– whether the dimension of the features can be effectively reduced and whether a smaller decision tree can be learned using the constructed features by the new approach while retaining the classification performance; and
– whether a combination of the original features and newly constructed features can improve the classification performance.

2 Using GP for Feature Construction

As shown in Figure 1, this approach first uses a GP system to construct high-level features from the original features. Based on the constructed features, the training set is then transformed into a new training set, from which the decision tree (DT) method is used for learning a DT classifier. The DT classifier is then applied to the transformed test set to obtain the final classification results of this approach.

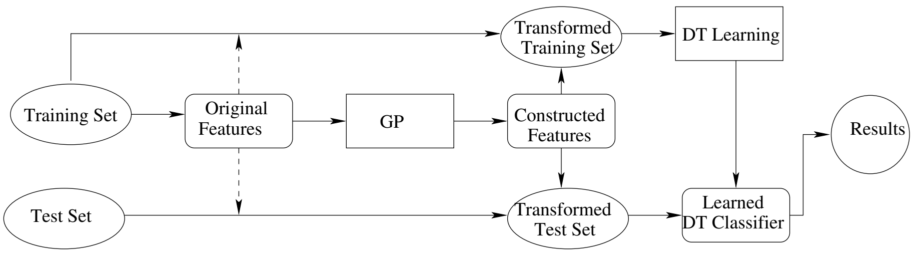


Fig. 1. Overview of this approach

In the rest of this section, we will first describe the rationale of feature construction, then describe two components we have used for this purpose. The GP method for feature construction will be described in section 3. The DT part uses the standard C4.5 decision tree and will be briefly described in section 4.

2.1 Rationale of Multiple Feature Construction

In many classification tasks, the data is so dispersed that class boundaries cannot be easily distinguished by simply using one or more original features. One reason is that the instances of different classes make hyper-ellipsoids whose axes are not orthogonal to the original features. This phenomenon causes a severe difficulty for classifiers like decision trees, which cannot perform any representation transformation by themselves [10].

Figure 2 illustrates this difficulty, where there are three classes plotted along two features ($feature_1$ and $feature_2$) with class labels "$\circ$", "$+$" and "$*$" respectively. The discrimination border of class $\circ$ and $+$ is oblique to the coordinate axes

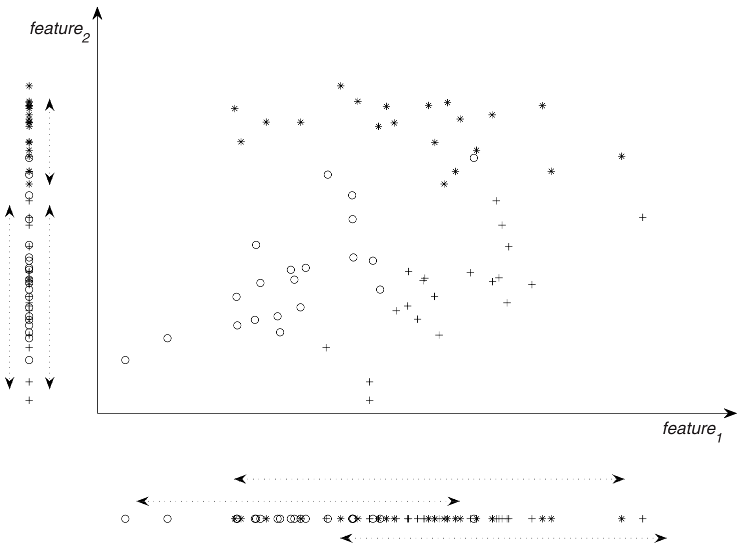


Fig. 2. Entropy of classes over their intervals

and neither $feature_1$ nor $feature_2$ is directly suitable for constructing a decision stub to separate these class instances (even if they contain latent useful information). However, the situation is different for the $*$ class. The boundary of this class is almost perpendicular to the $feature_2$ axis. Thus this feature can provide good enough information for separating instances of this class from the others.

For feature construction, we would like to find what makes a feature a good discriminator for a particular class. In figure 2, we have projected all class instances to the axes. Double arrow lines show the width of class intervals along each axis. A class interval along an axis is a rough area with the most occurrence of class instances. When the class boundaries are oblique, class intervals overlap. If we look at the projection line, we find that when the class intervals overlap, the projection area beneath them contains many different class labels. By contrast, when a feature provides good enough information for separating a particular class (e.g. $feature_2$ for class $*$), the class interval contains fewer instances of other classes. The occurrence of the other class instances in the interval of one class shows a kind of uncertainty. In this view, a proper feature for a class is one that the class interval along it has the minimum amount of uncertainty. Accordingly, this approach will construct the same number of such good features as the number of classes for classification, each of which can minimise the level of uncertainty for a single class. To do this, we model two components of the system: class interval and a metric for measuring uncertainty of a particular interval.

2.2 Feature Construction Component 1: Class Intervals

As the abstract concept of class interval shows the dispersion of the class instances along the feature axis, it is analogous to the concept of class distribution.

As the aim is to find an area where the instances of one class are more probable than those of the others, the mean and standard deviation of a class seem good approximations for locating the interval. We use the mean of the class instances as the centre of the interval and use the standard deviation of the instances to determine the width of the interval. Assuming that the class data approximately follow normal distributions, we can use the following formula to determine the boundaries of the interval:

$$\mu - 3\sigma \leq x_c \leq \mu + 3\sigma \tag{1}$$

where x_c is the value of the feature for an instance of class c and μ and σ are mean and standard deviation of the class along the feature respectively.

The area formulated by equation (1) can cover 99% of the class instances [11]. As the GP evolutionary learning process aims to construct a good feature for separating a class from others, it will have relatively a low level of uncertainty in any distance from the mean of class. So this setting is expected to meet the requirement in most cases.

2.3 Feature Construction Component 2: Entropy of a Class Interval

To measure the level of uncertainty of a class interval, we used the concept of entropy from information theory. Once the boundary of a class interval has been determined, we regard the interval as an information channel and different class labels over the interval as different symbols in the channel. Accordingly, we used the entropy of a channel to measure the level of uncertainty [12]:

$$Entropy(I) = \sum_{c \in C} -P_I(c) \log_2 P_I(c) \tag{2}$$

where I is the interval, C is the set of all classes, c is the class index, and $P_I(c)$ is the probability of class c in interval I, which can be easily calculated by counting the number of occurrences of the class on the interval. Note that the base of logarithm is not important. Using entropy as the measure of uncertainty has a benefit of efficiency in terms of the computation time. The smaller the entropy, the smaller the uncertainty and the better the constructed feature.

3 Algorithm and Design Issues

The feature construction metrics described in the last section are used for constructing the fitness measure to evaluate how well the individual programs are in a population. At the end of the evolutionary process, the best evolved program will be used as the constructed feature.

We used the tree-based structure to represent genetic programs [13], each of which produces a single floating number output. In the rest of this section, we will describe the terminal set, the function set, and the fitness function for constructing optimal features.

3.1 Terminal Set and Function Set

In the GP system, the original features in a classification data set were used to form the terminal set. A number of randomly generated constants are also added to the terminal set.

The four standard arithmetic operators were used to form the function set: $FuncSet = \{+, -, \times, \div\}$. The $+$, $-$, and $\times$ operators have their usual meanings — addition, subtraction and multiplication, while $\div$ represents "protected" division which is the usual division operator except that a divide by zero gives a result of zero. Each of these functions takes two arguments.

3.2 Fitness Function

For a training set S and a target class c, the fitness of an individual program p is calculated as follows.

1. Use program p to convert each example in the training set S to a new transformed set S_{new}. The program uses the original feature values of an instance as inputs and produces a single floating point value for each instance.
2. Make a subset Sub_c from all instances of the target class c in S_{new}.
3. Calculate the mean μ and standard deviation σ over all examples of Sub_c.
4. Determine an $Interval_c$ according to equation (1), which is $[\mu - 3\sigma, \mu + 3\sigma]$.
5. Collect all instances from S_{new} which fall in the interval and calculate the entropy over $Interval_c$ using equation 2. The entropy is used as the fitness of the program p.

If most of the instances falling into an interval belong to a single class, the entropy will be quite low. According to this design, the smaller the fitness, the better the program, and consequently the better the constructed feature.

4 Experiment Design and Configurations

Two groups of experiments were conducted. The first group of experiments compare our new approach with the standard decision tree approach, which directly uses the original features and the original training set. The second group of experiments investigate whether combining the"optimal" features constructed by our GP method with the original features together can improve the classification performance in the decision tree system. Both groups of experiments are examined on 12 benchmark data sets (described below) in terms of the classification accuracy, feature reduction, and learned decision tree size.

4.1 Data Sets

We used 12 data sets collected from the UCI machine learning repository [14] in the experiments. Table 1 summarises the main characteristics of these data sets. These include binary and multiple class classification problems and also problems with either relatively low or high number of features.

Table 1. Specification of data sets used in experiments

Problem	# Features	# Samples	# Classes
Balance Scale	4	625	3
Glass Identification	9	214	6
Iris Plant	4	150	3
Johns Hopkins Ionosphere	34	351	2
Liver disorders	6	345	2
Pima Indians Diabetes	8	768	2
Sonar	60	208	2
Thyroid Disease	5	215	3
Waveform	21	5000	3
Wine Recognition	13	178	3
Wisconsin Breast Cancer (WBC)-Diagnostic	30	569	2
Wisconsin Breast Cancer (WBC)-Original	9	683	2

4.2 Experiment Configurations

GPLAB, a genetic programming toolbox in MATLAB [15], was used as our GP platform. The ramped half-and-half method [13] was used for generating programs in the initial population and for the mutation operator. During the search process we used a heavy dynamic limit on tree depth [16] to control the code bloating. The probability of the crossover and mutation operators are adapted automatically in run time [17]. We also used a kind of elitism to keep the best individual of the generation. The population size is 300. The initial maximum program tree depth was set to 3 but the trees could be increased to a depth of 6 during evolution (due to the large number of features, the maximum tree depth was 8 for the JH Ionosphere and the Sonar sets). The evolution was terminated at a maximum number of generations of 30 because we found that in practice there is no significant improvement after that.

The C4.5 decision tree inducer [10] was used as the classification system. In all experiments we used the 10-fold cross validation method. The average results on the test sets will be described in next section.

5 Results

5.1 Experiment 1: The New Approach vs Basic DT

The first group of experiments were designed to compare the standard DT approach with our proposed system in which the original features are replaced by a set of new features constructed by GP. The comparison is in terms of the number of features, the error rate, and the learned decision tree size.

Table 2 presents the experiment results. As an example for the *Wine Recognition* data set, the number of original features is 13. We have constructed one new feature for every class in the problem (one complete GP run for each class in the problem), so as number of different classes in this problem is 3, we have constructed 3 new features for this problem which means we have had 76.9% dimension reduction. Comparing the error of DT classification using original features (9.0%) with the case that new constructed features have been used (3.3%) shows 63.3% improvement. Finally there is about 6.5% reduction in DT size. Error values are average over testing folds. As an example a constructed feature for this data set is like `(* (* (* (- x`$_1$` x`$_{13}$`) (* x`$_{10}$` x`$_1$`)) 0.02) (* (* x`$_{13}$` (* (/ x`$_{13}$` x`$_7$`) (* x`$_{10}$` x`$_{13}$`))) (* (+ 0.71 (+ x`$_9$` x`$_1$`)) (* x`$_1$` (* x`$_{10}$` x`$_2$`)))))` where x_i indicates the i-th original input.

Table 2. Results of the proposed approach and the basic DT approach

Problem	# of Features			Error Rate (%)			# of nodes in DT		
	Org	Cnst.	Reduction	Org.	Cnst.	Imprv.	Org.	Cnst.	Shrink
Balance Scale	4	3	25.0%	22.5	0.0	100.0	86.0	5.0	94.2%
Glass Identification	9	6	33.3%	32.2	27.0	16.2	46.6	37.6	19.3%
Iris Plant	4	3	25.0%	5.3	4.0	24.5	8.4	8.4	0.0%
JH Ionosphere	34	2	94.1%	10.8	10.0	7.4	27.0	10.4	61.5%
Liver Disorders	6	2	66.7%	35.3	29.6	16.2	44.6	8.8	80.3%
Pima Diabetes	8	2	75.0%	25.4	24.6	3.2	40.6	3.0	92.6%
Sonar	60	2	96.7%	27.4	24.1	12.0	27.2	5.0	82.2%
Thyroid Disease	5	3	40.0%	7.9	4.7	40.5	15.4	11.2	27.3%
Waveform	21	3	85.7%	23.1	19.4	16.0	532.4	48.2	90.9%
Wine Recognition	13	3	76.9%	9.0	3.3	63.3	9.2	8.6	6.5%
WBC-Diagnostic	30	2	93.3%	6.9	4.4	36.2	22.2	3.0	86.5%
WBC-Original	9	2	77.8%	4.2	2.8	33.3	22.0	6.2	71.8%
Average			65.8%			30.7%			59.4%

As shown in Table 2, on all the 12 data sets, the system has improved the classification performance while reducing the number of features and consequently having smaller learned DTs than when original features only are used. In particular, the feature reduction rate was over 50% on eight out of the 12 classification problems, the classification error rate improvement was over 15% on nine out of the 12 data sets, and the learned decision tree was shrunk more than 60% on eight out of the 12 data sets. The average rates for feature reduction, error rate improvement and the learned decision tree shrinking were 65.8%, 30.7% and 59.4%, respectively.

These results suggest that GP is effective in constructing high-level features for classification with improved classification performance and smaller learned decision tree classifier. The newly constructed features seem to be able to give more generalisation capability to the target learner for these classification tasks than the original features. The small learned decision trees have the advantages of easy interpretation and fast execution on the new unseen data for prediction/classification. However the actual constructed features may be difficult to interpret meaningfully.

5.2 Experiment 2: Extending Feature Sets

The second group of experiments examined whether combining the original features with the newly constructed features can result in better classification performance of DT. For this purpose we have compared the number of features and the average error rates achieved by the DT classifier using the original features only, the constructed features only, and the combination of the original and constructed features. The results are shown in Table 3.

Table 3. Results of the combined feature set

Problem	# Features			Error Rate (%)		
	Org.	Cnst.	Comb.	Org.	Cnst.	Comb.
Balance Scale	4	3	7	22.5	0.0	0.0
Glass Identification	9	6	15	32.2	27.0	27.5
Iris Plant	4	3	7	5.3	4.0	5.3
JH Ionosphere	34	2	36	10.8	10.0	10.2
Liver Disorders	6	2	8	35.3	29.6	32.5
Pima Diabetes	8	2	10	25.4	24.6	27.5
Sonar	60	2	62	27.4	24.1	22.7
Thyroid Disease	5	3	8	7.9	4.7	5.1
Waveform	21	3	24	23.1	19.4	19.6
Wine Recognition	13	3	16	9.0	3.3	3.9
WBC-Diagnostic	30	2	32	6.9	4.4	4.4
WBC-Original	9	2	11	4.2	2.8	3.2

According to Table 3, the combined feature set improved the classification performance (achieved smaller error rate) over only using the original features on 11 out 12 data sets, suggesting that the GP constructed features do play a positive role for classification. When comparing with constructed features alone, however, the classification results achieved using the combined feature set were worse in most cases. Besides, due to the bigger number of features in the combined set, this performance drop happens together with a dimension increase. A possible reason would be that the constructed features and the original features are redundant. However, if decision tree classifier had good feature selection ability, then the combined features should not lead to worse performance than the constructed features alone. This might suggest that C4.5 does not really have as good feature selection ability as mentioned in the literature.

6 Conclusions

The goal of this paper was to investigate a GP approach for feature construction in classification for improving classification performance, reducing feature dimensions and learning a smaller classifier. The goal was successfully achieved by developing a non-wrapper approach using GP to construct multiple (high-level) features from the original features and these constructed features are then used by decision trees for classification. As the feature construction is independent of the classification, the GP fitness function was designed based on the class interval and its entropy. This approach was examined and compared with the basic decision tree method, using the original features and using a combination of the original features and constructed features, on 12 benchmark classification problems. The results show that the new approach outperforms the decision tree method in terms of the classification performance. In addition, as the number of generated features is usually less than the number of original features, there is a considerable reduction in the number of input dimensions and complexity of the learned decision trees.

Combining the constructed features with the original features improved the classification performance on most of these tasks over the original features alone, but the results obtained were worse than those with the GP constructed features only. While this is partially because the combination of these features contains redundancy, another reason would be that the decision tree learning approach did not do a good job for feature selection on these data sets.

In future work, we will investigate new ways for automatic feature selection using GP and integrate feature construction into feature selection for improving classification performance. We will also compare this approach with other feature construction and dimension reduction methods such as support vector machines (SVMs) and principle component analysis (PCA).

Acknowledgment

This work was supported in part by the Marsden Fund (05-VUW-017) administrated by the Royal Society of New Zealand and the Research Fund (URF7/39) at Victoria University of Wellington.

References

1. Ekart, A., Markus, A.: Using genetic programming and decision trees for generating structural descriptions of four bar mechanisms. Artificial Intelligence for Engineering Design, Analysis and Manufacturing 17(3), 205–220 (2003)
2. Muni, D.P., Pal, N.R., Das, J.: Genetic programming for simultaneous feature selection and classifier design. IEEE Transactions on Systems, Man and Cybernetics, Part B 36(1), 106–117 (2006)
3. Krawiec, K., Bhanu, B.: Visual learning by coevolutionary feature synthesis. IEEE Transactions on System, Man, and Cybernetics – Part B 35(3), 409–425 (2005)
4. Bhanu, B., Krawiec, K.: Coevolutionary construction of features for transformation of representation in machine learning. In: Barry, A.M. (ed.) GECCO 2002. Proceedings of the Bird of a Feather Workshops, Genetic and Evolutionary Computation Conference, pp. 249–254. AAAI, New York (2002)
5. Otero, F.E.B., Silva, M.M.S., Freitas, A.A., Nievola, J.C.: Genetic programming for attribute construction in data mining. In: Ryan, C., Soule, T., Keijzer, M., Tsang, E.P.K., Poli, R., Costa, E. (eds.) EuroGP 2003. LNCS, vol. 2610, pp. 384–393. Springer, Heidelberg (2003)

6. Kohavi, R., John, G.: Wrappers for feature subset selection. Artificial Intelligence, 273–324 (1997)
7. Smith, M.G., Bull, L.: Genetic programming with a genetic algorithm for feature construction and selection. Genetic Programming and Evolvable Machines 6(3), 265–281 (2005)
8. Krawiec, K.: Genetic programming-based construction of features for machine learning and knowledge discovery tasks. Genetic Programming and Evolvable Machines 3(4), 329–343 (2002)
9. Muharram, M.A., Smith, G.D.: Evolutionary feature construction using information gain and gini index. In: Keijzer, M., O'Reilly, U.M., Lucas, S.M., Costa, E., Soule, T. (eds.) EuroGP 2004. LNCS, vol. 3003, pp. 379–388. Springer, Heidelberg (2004)
10. Quinlan, J.R.: C4.5: Programs for Machine Learning. Morgan Kaufmann, San Francisco (1993)
11. Kreyszig, E.: Advanced Engineering Mathematics, 8th edn. John Wiley, Chichester (1999)
12. Mitchell, T.: Machine Learning. McGraw-Hill, New York (1997)
13. Koza, J.R.: Genetic Programming. MIT Press, Cambridge (1992)
14. Asuncion, A.D.N.: UCI machine learning repository (2007)
15. Silva, S., Almeida, J.: Gplab - a genetic programming toolbox for matlab. In: Proceedings of the Nordic MATLAB Conference, pp. 273–278 (2003)
16. Silva, S., Costa, E.: Dynamic limits for bloat control: Variations on size and depth. In: Deb, K., et al. (eds.) GECCO 2004. LNCS, vol. 3103, pp. 666–677. Springer, Heidelberg (2004)
17. Davis, L.: Adapting operator probabilities in genetic algorithms. In: Proceedings of the Third International Conference on Genetic Algorithms, pp. 70–79. Morgan Kaufman, San Francisco (1989)

Adjusting Population Distance for the Dual-Population Genetic Algorithm

Taejin Park, Ri Choe, and Kwang Ryel Ryu

Department of Computer Engineering, Pusan National University
Jangjeon-Dong San 30, Kumjeong-Ku, Busan 609-735, Korea
{parktj, choilee, krryu}@pusan.ac.kr

Abstract. A dual-population genetic algorithm (DPGA) is a new multi-population genetic algorithm that solves problems using two populations with different evolutionary objectives. The main population is similar to that of an ordinary genetic algorithm, and it evolves in order to obtain suitable solutions. The reserve population evolves to maintain and offer diversity to the main population. The two populations exchange genetic materials using interpopulation crossbreeding. This paper proposes a new fitness function of the reserve population based on the distance to the main populations. The experimental results have shown that the performance of DPGA is highly related to the distance between the populations and that the best distance differs for each problem. Generally, it is difficult to decide the best distance between the populations without prior knowledge about the problem. Therefore, this paper also proposes a method to dynamically adjust the distance between the populations using the distance between good parents, i.e., the parents that generated good offspring.

Keywords: evolutionary algorithm, dual-population genetic algorithm, adjusting population distance, multi-modal function optimization.

1 Introduction

The population of a genetic algorithm (GA) evolves by using genetic operators inspired by evolutionary biology, such as selection, crossover, and mutation. Selection makes an algorithm exploit promising individuals by eliminating less-fit individuals. Crossover and mutation generate new offspring by either recombining or modifying the individuals of the current population. In general, as the population of a GA evolves, it loses its diversity and finally the entire population converges on an individual. If this convergence progresses rapidly, the population generally converges on one of the local optimum. This phenomenon is called *premature convergence*. In this state, since new offspring cannot be found through a crossover between individuals of the population, the population evolves only through mutation. Therefore, it is difficult for the population to escape from the local optimum and to continually evolve to the global optimum.

Many previous works have proposed various methods to cope with this problem. In some of these works, the convergence rate has been controlled by an adaptation of the mutation rate or selection pressure [1, 2]. In some other works, the population was made to converge to several species by using specially designed genetic operators such as fitness sharing [3, 4] and crowding [5]. Even after the population converges, new offspring could be generated by recombining different species. Finally, multi-population genetic algorithms (MPGAs) manipulate two or more sub-populations. The partially isolated nature of the sub-populations helps the population as whole to maintain diversity. There are various works pertaining to MPGAs, for example, the island model GA [6, 7], multinational GA [8], forking GA [9], and bi-objective multi-population algorithm (BMPGA) [10].

A dual-population genetic algorithm (DPGA) is a type of MPGA that uses an extra population to maintain diversity [11, 12]. The DPGA has two important differences as compared to the other MPGAs. Firstly, although the sub-populations of the other MPGAs have the same evolutionary objective of finding a good solution, the two populations of DPGA have different evolutionary objectives. The *main population* of DPGA evolves to find a good solution similar to that of ordinary GAs, but the *reserve population* merely evolves to provide diversity to the main population. Secondly, DPGA uses interpopulation crossbreeding instead of migration, which is most widely used in MPGAs, for exchanging genetic material between populations. According to the migration method, the populations evolve independently and exchange some of their individuals periodically. Since the sub-populations of other multi-population GAs use the same or similar fitness function, the migrants easily assimilate into the new population. However, since the populations of DPGA have completely different fitness functions, the migrants in the new population are very likely to have a low fitness value and will rarely be selected as parents. As a result, it is difficult for these migrants to assimilate into a different population. Since the crossbred offspring contain the genetic material of both populations, they are expected to have relatively high fitness and to assimilate more easily into a new population.

To provide the main population with sufficient diversity, the reserve population of DPGA has to maintain an appropriate distance from the main population. If the reserve population is too similar to the main population, it will be unable to provide sufficient diversity. On the contrary, if the reserve population is too different from the main population, the crossbred offspring will be unable to assimilate easily into the new population because of their low fitness. In our previous work [12], the algorithm maintains the distance between the populations by parent and survival selection. The algorithm selects similar individuals as parents for crossbreeding and selects different offspring from the individuals of the main population for the reserve population. However, this method cannot change the distances between the populations to the desired level. This paper proposes a new fitness function for the reserve population, which depends on the distance from the individuals of the main population. Due to this fitness function, the algorithm is able to maintain the distance between two populations at a desired level. The experiments that use this fitness function reveal that the performance of DPGA is closely related to the distance between the two populations and that the best distance differs for each problem. Therefore, this paper also proposes a method for adjusting the distance between two populations using the information gathered through evolution in order to determine the desirable distance.

Section 2 describes the DPGA and a new fitness function for the reserve population. Section 3 shows the method of adjusting the distance between the two populations of DPGA. Section 4 reports the experimental results of various population distances and a comparison with other genetic algorithms. Section 5 finally provides some conclusions and future works.

2 Dual-Population Genetic Algorithm

DPGA manipulates two populations with different evolutionary objectives, and the individuals of each population are evaluated by its own fitness function based on its evolutionary object. The fitness function *fm*(**x**) of the main population is similar to the evaluation function given by a problem, since its evolutionary objective is to find a good solution. Since the evolutionary objective of the reserve population is to provide diversity to the main population, the individuals of the reserve population are evaluated relative to the average distance from the individuals of the main population. This paper uses equation (1) as the fitness function of the reserve population so that each individual of the reserve population can maintain a given distance δ from the individuals of the main population.

$$fr_{\delta}(\mathbf{x}) = 1 - \left|\delta - d(\mathbf{M},\mathbf{x})\right| \tag{1}$$

where $0 \le \delta \le 1$, and $d(\mathbf{M}, \mathbf{x})$ is the average distance between the individuals of the main population **M** and an individual **x** of the reserve population. Assuming a binary representation for a chromosome, $d(\mathbf{M}, \mathbf{x})$ is calculated using equation (2).

$$d(\mathbf{M},\mathbf{x}) = \frac{1}{|\mathbf{M}|}\sum_{\mathbf{m}\in\mathbf{M}} dist(\mathbf{m},\mathbf{x}) = \frac{1}{l}\sum_{k=1}^{l}\left|f_{\mathbf{M},k} - x_k\right| \tag{2}$$

where n is the size of the main population and l is the length of a chromosome. $dist(\mathbf{m},\mathbf{x})$ is the distance between two chromosome vectors **m** and **x**, and it can be con sidered the Hamming distance for binary representation. $f_{\mathbf{M},k}$ is the frequency of the *k*t h gene value "1" of the main population **M**. x_i is the frequency of the *k*th gene value " 1" of the chromosome vector **x** and is identical to the *k*th gene value of the chromoso me. The details of this function can be found in previous works [12, 13]. When $\delta = 0$, more similar individuals to the main population are evaluated with higher fitness. Con trary, when $\delta = 1$, more different individuals from main population are evaluated with higher fitness. In [12], d(M,x) is used as the fitness function of the reserve population . It is identical to $fr_{\delta}(\mathbf{x})$ of $\delta = 1$.

DPGA begins with two randomly generated populations, and each population is evaluated with its fitness function. To generate new offspring, the algorithm selects four parents—two from the main population and two from the reserve population—according to their own fitness values. Given the four parents, the algorithm generates six offspring by inbreeding between parents from the same population twice and crossbreeding the parents randomly selected from different populations once. After mutating these six offspring, the algorithm selects offspring for the next generation of each population. The two offspring generated by crossbreeding and the two generated by inbreeding between the parents from the main population are candidates for the next-generation main population. The algorithm evaluates these four parents using *fm*(**x**) of the main population and then selects the two best offspring for the main population. Similarly, the two offspring generated by crossbreeding and the two generated by inbreeding between the parents from the reserve population become candidates for the next-generation reserve population. Using the fitness function $fr_\delta(\mathbf{x})$, the two best offspring are selected for the reserve population. Since the *crossbred* offspring generated by crossbreeding contain genetic material of both populations, they can be important mediums of information exchange. Figure 1 shows the DPGA algorithm, and additional details can be found in a previous work [12].

```
Procedure DPGA
begin
Initialize main population $\mathbf{M}_0$ and $\mathbf{R}_0$
Evaluate $\mathbf{M}_0$ using $fm(\mathbf{x})$
Evaluate $\mathbf{R}_0$ using $fr(\mathbf{x})$
$t := 0$
repeat
    repeat
        Parent Selection:
            A.  Select two parent $\mathbf{p}_{m1}$ and $\mathbf{p}_{m2}$ from $\mathbf{M}_t$
            B.  Select two parent $\mathbf{p}_{r1}$ and $\mathbf{p}_{r2}$ from $\mathbf{R}_t$
        Offspring generation:
            A.  (Inbreeding) Generate two offspring $\mathbf{c}_{m1}$ and $\mathbf{c}_{m2}$ by recombining $\mathbf{p}_{m1}$ and $\mathbf{p}_{m2}$.
            B.  (Inbreeding) Generate two offspring $\mathbf{c}_{r1}$ and $\mathbf{c}_{r2}$ by recombining $\mathbf{p}_{r1}$ and $\mathbf{p}_{r2}$.
            C.  (Crossbreeding) Generate two offspring $\mathbf{c}_{c1}$ and $\mathbf{c}_{c2}$ by recombining $\mathbf{p}_{m2}$ and $\mathbf{p}_{r1}$.
            D.  Mutate all generated offspring
        Evaluation & Survival selection:
            A.  Evaluate $\mathbf{c}_{m1}$, $\mathbf{c}_{m2}$ $\mathbf{c}_{c1}$ and $\mathbf{c}_{c2}$ using $fm(\mathbf{x})$ and add the best two to $\mathbf{M}_{t+1}$
            B.  Evaluate $\mathbf{c}_{r1}$, $\mathbf{c}_{r2}$, $\mathbf{c}_{c1}$ and $\mathbf{c}_{c2}$ using $fr(\mathbf{x})$ and add the best two to $\mathbf{R}_{t+1}$
    until $|\mathbf{M}_{t+1}| == pop\_size$
    $t := t + 1$
until terminated = true            // e.g., $t > t_{max}$
end
```

Fig. 1. DPGA algorithm

3 Adapting Population Distance

Each individual of the reserve population evolves such that it approaches the distance of a given δ from the main population because of its fitness function $fr_\delta(\mathbf{x})$. When δ is extremely small, the individuals of the reserve population are too similar to the indivi duals of the main population to provide enough diversity. On the contrary, when δ is t oo large, the parents for crossbreeding are too different from each other to generate go od offspring. This is because when the parents for crossbreeding are genetically very distant from each other, the probability that the offspring will have worse characteristi cs than their parents is high. Accordingly, as the population converges and their avera ge population fitness attains high values, it becomes difficult for the crossbred offspri ng to win the survival selection. In this case, the reserve population also cannot provid e sufficient diversity to the main population, since genetic materials are rarely exchan ged between populations.

From experiments, it has been observed that the performance of DPGA is highly dependent on the value of δ. In addition, the best value of δ differs in each problem. Due to this difficulty of determining the best value of δ, this paper proposes a method for adjusting the value of δ by using the information gathered through evolution.

For the reserve population to provide sufficient diversity to the main population, sufficient crossbred offspring have to be selected for the main population. The crossbred offspring should have high evaluation values and survive the competition from inbred offspring. $fr_\delta(\mathbf{x})$ of the reserve population makes the average distance between the parents for crossbreeding equal to the value of δ. Due to $fr_\delta(\mathbf{x})$ of the reserve population, the average distance between the parents for crossbreeding should be equal to δ. If we assume that the probability of generating good offspring, which contributes to providing diversity and increasing the fitness value, is highly correlated to the distance between the parents, we can set the value of δ to the distance. Furthermore, the value of δ should be adjusted according to the change in the populations in order to maintain appropriate diversity. The offspring satisfying one of the following conditions are defined as *decent* offspring.

– Both of its parents are selected from the main population and the offspring is better than both of its parents.
– One of its parents is selected from the main population and the offspring is better than the parent from the main population.

The value of δ_{t+1} for the next generation is decided from the set $\mathbf{G}_t$ consisting of parent pairs that have generated decent offspring at generation t. More precisely, the value of δ_{t+1} is decided from the following equation.

If $\mathbf{G}_t \neq \emptyset$,

$$\delta_{t+1} = (1-\alpha)\cdot\delta_t + \alpha\cdot\frac{1}{|\mathbf{G}_t|}\sum_{(\mathbf{p}_1,\mathbf{p}_2)\in\mathbf{G}_t}|\mathbf{p}_1-\mathbf{p}_2| \tag{3}$$

else,

$$\delta_{t+1} = 1 \tag{4}$$

where α is the update rate of δ_t, and a greater value indicates a more rapid change in δ_t. We used $\alpha = 0.1$ for the experiments. The second term of the right side of the equation is the average distance between parents' pair included in set $\mathbf{G}_t$. As the population converges, this average distance has a tendency to reduce in value. As mentioned previously, if δ_t has an extremely low value, the reserve population cannot provide sufficient diversity to the main population. This is the reason why a method for increasing the value is required. $\mathbf{G}_t = \varnothing$ implies that decent offspring have not been generated in the last generation. We consider that this situation occurs because of the lack of diversity and therefore we set the value of δ_t to the largest possible value "1", in order to provide the reserve population with sufficient diversity.

4 Experimental Results

Four multimodal functions have been used in the experiments: Rosenbrock, Griewang k, Rastrigin, and Schwefel (see Table 1). All these functions are widely used to compa re the performance of evolutionary algorithms. Ten-dimensional versions of all functi ons were used in the experiments, and each variable was represented with a ten-bit gra y code. Additional details of these functions can be found in [12, 14].

Table 1. Test functions for experiments

Name	Function	
Rosenbrock	$f(x_i \mid_{i=1,D}) = \sum_{i=1}^{D-1}[100(x_{i+1} - x_i^2)^2 + (1-x_i)^2]$	$x_i \in$ [-2.048, 2.047]
Griewangk	$f(x_i \mid_{i=1,D}) = 1 + \sum_{i=1}^{D}\frac{x_i^2}{4000} - \prod_{i=1}^{D}\cos(\frac{x_i}{\sqrt{i}})$	$x_i \in$ [-512, 511]
Rastrigin	$f(x_i \mid_{i=1,D}) = 10D + \sum_{i=1}^{D}(x_i^2 - 10\cos(2\pi \cdot x_i))$	$x_i \in$ [-5.12, 5.11]
Schwefel	$f(x_i \mid_{i=1,D}) = \sum_{i=1}^{D} - x_i \cdot \sin(\sqrt{x})$	$x_i \in$ [-512, 511]

We implemented a standard genetic algorithm (SGA) and three types of DPGAs—DPGA(δ), DPGA-A, and DPGA-S. The DPGA(δ)algorithm is described in this paper; it uses a fixed value of δ for the fitness function $fr_\delta(\mathbf{x})$ of the reserve population. DPGA-A is an algorithm for adjusting the value of δ using the method described in the previous chapter. The DPGA-S algorithm has been described in a previous work [13]. Since DPGA-S uses d(M,x) as the fitness function of the reserve population, the two populations of DPGA-S are considerably different from each other. Therefore, to maintain an appropriate distance between the populations, after selecting two parents from the main population using $fm(\mathbf{x})$, DPGA-A selects two parents that are similar to the parents from the main population instead of using $fr_\delta(\mathbf{x})$. However, $fr_\delta(\mathbf{x})$ is still used for survival selection to choose individuals for the reserve population of the next generation.

The population size was set to 100 for the SGA and 50 + 50 for all DPGAs. We used a two-point crossover with a crossover rate of 1.0 and bitwise mutation with a mutation rate of 0.01. Tournament selection along with elitism was used for the selection. For each experiment, 100 independent runs were executed and averaged and each algorithm was run for a maximum of 5,000 generations.

We have used DPGA(δ) with various values of δ in order to determine the best value of δ for each function. Table 2 shows the results of this experiment; the results of SGA can also be used for performance comparison. For the Rosenbrock and Griewangk functions, each cell in the table indicates the evaluated values of the optimal solution found during 5,000 generations. However, for the Rastrigin and Schwefel functions, each cell indicates the number of evaluations until an optimal solution is found since the algorithms found an optimal solution for most runs. For the Rosenbrock function, when $\delta = 0$, the algorithm provides the best performance, and as the value of δ increases, the performance deteriorates. For the Griewangk and Rastrigin functions, when $\delta = 0.1$, the algorithm provides the best performance. For the Schwefel function, the performance of the algorithm is worst when $\delta = 0$, while the best performance is obtained at a relatively large value—$\delta = 0.6$. On comparing these values with the results of SGA, when $0.1 \leq \delta \leq 0.4$, DPGA provides better performance for all test functions. These results reveal that for each problem, the best value of δ differs from these results; further, a large value of δ is required for more diversity.

Table 2. Experimental results for DPGA with various values of δ

	Rosenbrock	Griewangk	Rastrigin	Schwefel
SGA	2.0785	0.3821	60951	181239
DPGA(0.0)	**0.3812**	0.1475	49961	415467
DPGA(0.1)	0.4719	**0.0662**	**32006**	131196
DPGA(0.2)	1.0869	0.0743	40926	63360
DPGA(0.3)	1.5712	0.0778	47024	54866
DPGA(0.4)	1.7956	0.0835	50540	43980
DPGA(0.5)	2.1320	0.0872	55779	40063
DPGA(0.6)	2.1988	0.0846	53310	**36376**
DPGA(0.7)	2.3683	0.0868	65255	36445
DPGA(0.8)	2.2975	0.0859	64100	38014
DPGA(0.9)	2.3939	0.0905	77771	44057
DPGA(1.0)	2.3140	0.0926	74222	43056

Figure 2 shows the average diversity and the proportion of crossbred offspring of the main population over all generations with various values of δ for the Griewangk function. The other test functions also yield a similar graph. The diversity $div(\mathbf{M})$ of the main population $\mathbf{M}$ is calculated using equation (3) [13]

$$div(\mathbf{M}) = \sum_{i=1}^{n} \sum_{j=1}^{i-1} dist(\mathbf{c}_i, \mathbf{c}_j) \tag{5}$$

where n is the population size and $\mathbf{c}_i$ is the ith individual of the population $\mathbf{M}$. From this figure, we can observe that the main population cannot maintain sufficient diversity when δ is extremely small and that crossbred offspring are rarely selected for the main population when δ is considerably large.

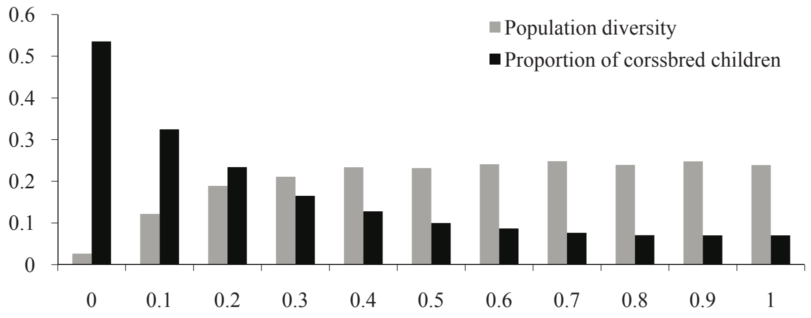


Fig. 2. Population diversity and proportion of crossbred children of main population with various values of δ

We compared SGA, DPGA-S, DPGA-A, and DPGA(δ) for those values of δ for which the best performance was obtained (as shown in Table 2); the results of this comparison are shown in Table 3. The performances of DPGA-S and DPGA-A are better than that of SGA but worse than that of DPGA(δ) with the best value of δ. With regard to the Griewangk and Rastrigin functions, for which 0.1 is the best value of δ, the DPGA-A and DPGA-S algorithms show similar performance; however, for the Rosenbrock and Schwefel functions, the performance of DPGA-A is better, and almost identical to that of DPGA(δ) with the best δ, than that of DPGA-S. On comparing with the results of Table 2, the results of DPGA-A are similar to those of DPGA(δ) for $0.1 \leq \delta \leq 0.2$ for all test functions. For the Rosenbrock, Griewangk, and Rastrigin functions, the results of DPGA-A are similar to those of DPGA(δ) for $0.1 \leq \delta \leq 0.2$; however, for the Schwefel function, which needs more diversity, the results of DPGA-A are similar to those of DPGA(δ) for $0.4 \leq \delta \leq 0.5$.

Table 3. Performance comparison of SGA and all DPGAs

Problem	SGA	DPGA-S	DPGA-A	DPGA(δ)
Rosenbrock	2.0785	0.8098	0.6419	0.3812
Griewangk	0.0807	0.0738	0.0729	0.0662
Rastrigin	60951	35074	36973	32006
Schwefel	181239	70047	43171	36376

Table 4 shows the average population distance of DPGA-S and DPGA-A for all generations. The population distances of DPGA-S are similar to each other for all functions, but the population distances of DPGA-A are relatively well adjusted for each problem. Figure 3 shows the population distance for each generation during the early 500 generations. The population distance of DPGA-S is relatively uniform over all generations, but the population distance of DPGA-A keeps changing as evolution continues. In particular, a problem that requires more diversity, e.g., the Schwefel function, changes more frequently.

Table 4. Average population distance of DPGA-S and DPGA-A

Problem	DPGA-S	DPGA-A
Rosenbrock	0.1709	0.1188
Griewangk	0.1564	0.1850
Rastrigin	0.1563	0.1563
Schwefel	0.1559	0.2161

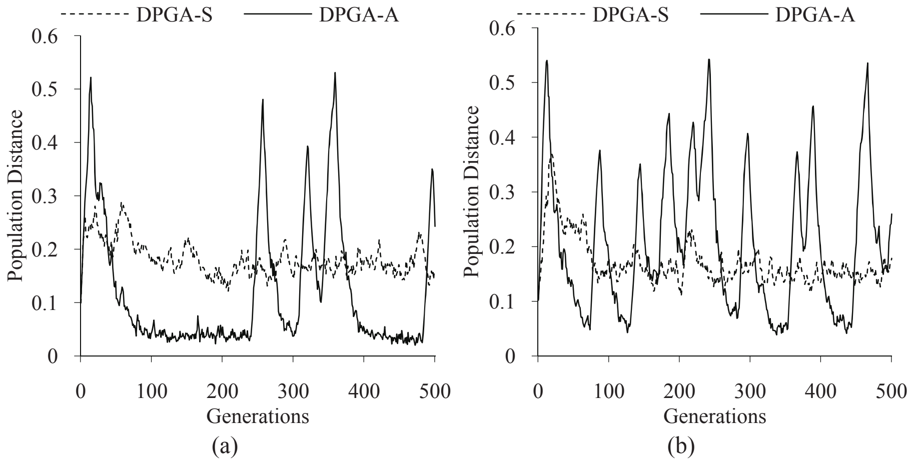


Fig. 3. Population distance of DPGA-S and DPGA-S for each generation: (a) Griewangk functi on and (b) Schwefel function

5 Conclusion and Future Works

DPGA is a novel genetic algorithm that uses an additional population to provide diversity to the main population. This paper proposes a new fitness function for the reserve population in order to maintain the population distance at the desired level. The experiments with this fitness function show that the population distance is greatly responsible for maintaining the diversity and exchanging information between populations, and the best population distance differs in each problem. In addition, when the best population distance is decided beforehand, we propose a method to adjust the population distance dynamically during evolution.

In the future works, we will examine a better method for adjusting the population distance by considering the property of a problem. Although the performance of the adjusting method proposed in this paper is better than that of the standard genetic algorithm and DPGA introduced in a previous work, its performance is worse than that of DPGA when the fixed best value of δ is used. Therefore, we will use the method of adjusting the population distance by using the additional information gathered through evolution. In addition, we will analyze the DPGA theoretically and experiment on more diverse problems. We particularly expect the proposed algorithm to exhibit a very good performance with regard to the non-stationary optimization problems or multi-objective optimization problems in which maintaining diversity is more important.

Acknowledgements

This work was supported by the Korea Research Foundation Grant funded by the Kor ean Government (MOEHRD) (The Regional Research Universities Program/Research Center for Logistics Information Technology).

References

1. Bäck, T.: The interaction of mutation rate, selection, and self-adaptation within a genetic algorithm. In: Parallel Problem Solving from Nature 2, PPSN-II, pp. 87–96. Elsevier, Amsterdam (1992)
2. Back, T.: Optimal mutation rates in genetic search. In: 5th International Conference on Genetic Algorithms, pp. 2–8. Morgan Kaufmann, San Francisco (1993)
3. Goldberg, D.E., Richardson, J.: Genetic Algorithms with sharing for multimodal function optimization. In: 2nd International Conference on Genetic Algorithm, Cambridge, Massachusetts, pp. 41–49 (1987)
4. Miller, B.L., Shaw, M.J.: Genetic algorithms with dynamic niche sharing for multimodal function optimization. In: IEEE International Conference on Evolutionary Computation, pp. 786–791. IEEE press, Los Alamitos (1996)
5. De Jong, K.A: An Analysis of the Behavior of a Class of Genetic Adaptive Systems. Ph.D. thesis, University of Michigan, Ann Arbor (1975)
6. Cantu-Paz, E.: A survey of parallel genetic algorithm. Technical Report, Illinois Genetic Algorithms Laboratory (1997)
7. Whitley, D., Rana, S., Heckendorn, R.B.: The island model genetic algorithm: on separability, population size and convergence. In: the AISB Workshop on Evolutionary Computation (1998)
8. Ursem, R.K.: Multinational evolutionary algorithms. In: IEEE Congress on Evolutionary Computation, vol. 3, pp. 1633–1640. IEEE Press, Los Alamitos (1999)
9. Tsutsui, S., Fujimoto, Y.: Forking genetic algorithm with blocking and shrinking modes (FGA). In: the 5th International Conference on Genetic Algorithms, pp. 206–215. Morgan Kaufmann, San Francisco (1993)
10. Jie, Y., Kharma, N., Grogono, P.: BMPGA: a bi-objective multi-population genetic algorithm for multi-modal function optimization. In: IEEE Congress on Evolutionary Computation, vol. 1, pp. 816–823. IEEE Press, Los Alamitos (2005)
11. Park, T., Ryu, K.R.: A dual-population genetic algorithm for balanced exploration and exploitation. In: Computational Intelligence, pp. 88–93. ACTA Press (2006)
12. Park, T., Ryu, K.R.: A dual population genetic algorithm with evolving diversity. In: IEEE Congress on Evolutionary Computation (to be published, 2007)
13. Wineberg, M., Oppacher, F.: Distances between populations. 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. 1481–1492. Springer, Heidelberg (2003)
14. Whitley, D., Mathias, K., Rana, S., Dzubera, J.: Evaluating evolutionary algorithms. Artificial Intelligence, 245–276 (1996)

An Improved Concurrent Search Algorithm for Distributed CSPs

Jian Gao, Jigui Sun, and Yonggang Zhang

College of Computer Science and Technology, Jilin University, Changchun, China 130012
Key Laboratory of Symbolic Computation and Knowledge Engineering of Ministry of Education, Changchun, China 130012
j_gao2006@hotmail.com, jgsun@jlu.edu.cn, ygzhang@163.com

Abstract. As an important area in AI, Distributed Constraint Satisfaction Problems (DisCSPs) can be used to model and solve many problems in multi-agent systems. Concurrent search, newly proposed, is an efficient technique for solving DisCSPs. In this paper, a novel concurrent search algorithm is presented. Dynamic Variable Ordering (DVO) is used in concurrent backtrack search instead of random variable ordering. In order to make DVO effective, domain sizes of unfixed variables are evaluated approximately according to current partial assignments after a variable is assigned. This method can be performed by a single agent and there is no need to send messages during heuristic computation. In addition, a simple look-ahead strategy inspired from centralized constraint programming techniques is added to the improved algorithm. Experiments on randomly generated DisCSPs demonstrate that the algorithm with DVO heuristic and look-ahead strategy can drastically improve performance of concurrent search.

Keywords: Concurrent search; Distributed CSPs; Dynamic variable ordering; Look-ahead strategy.

1 Introduction

As a significant problem in artificial intelligence, Constraint Satisfaction Problem (CSP) is a hotspot in recent years. Many kinds of real-world problems can be translated into CSPs. Some examples of such problems include: scheduling, planning, configuration, diagnostics, decision support[1,2,3]. A constraint satisfaction problem is described by a finite variable set and constraints among these variables. Variables and constraints in a distributed constraint satisfaction problem are in a distributed environment, and each agent contains a local constraint network, which global constraints connect to variables in other agents [4]. Agents assign values to variables in order to satisfy local constraints, and communicate with other agents to obtain a solution for the entire distributed problem[5]. Because many complex problems can be formalized as DisCSPs[6], DisCSPs are widely used to solve academic problems as well as problems in real world, such as distributed resource allocation, distributed scheduling and distributed multi-agent truth maintenance[2].

Some systematic and complete algorithms for solving DisCSPs were proposed in recent years. Centralized backtrack algorithm is a basic approach to solving DisCSPs. Synchronous backtrack algorithm performs backtracking among agents with a fixed agent ordering, an agent receives partial assignments from former agent, and instantiates local variables then sends new partial assignments to next agent. As a result, communication overhead decreases in this algorithm compared to centralized backtrack, but there is no parallelism. Makoto Yokoo et al. [5] proposed asynchronous backtrack, and it overcomes disadvantages in the synchronous backtrack algorithm. Agents work asynchronously. They instantiate local variables first, and then communicate with other agents to achieve a solution[7]. Concurrent search[8,9], introduced by Roie Zivan and Amnon Meisels, is a family of algorithms that contain multiple backtrack search processes, and search processes are performed asynchronously. Each search process has a Current Partial Assignment (CPA). CPAs, which have disjoint parts of the search space, pass among agents to find a solution in a backtracking manner. Moreover, concurrent backtrack search (ConcBT) with dynamic splitting CPAs strategy[8] can balance search spaces among CPAs according to difficulty of problems. Experiments indicate that method is efficient.

Some techniques for centralized search algorithm are efficient for solving CSPs. For example, Dynamic Variable Ordering heuristics[10] based on fail first principle [11] reduce the number of backtracks required in a search, which is one of the most important factors affecting the efficiency of centralized backtrack algorithms. Moreover, look-ahead strategy can detect inconsistency earlier so that unnecessary search is avoided. However, for reasons of communication overhead and private information [12], there is no agent that can perform those centralized methods without help of other agents. So those techniques can't be used in distributed environments, and concurrent search can't benefit from effect of minimum domain heuristic, which is one of the most efficient heuristics of dynamic variable ordering, or from effect of look-ahead strategy[3]. This paper improves concurrent search with ideas of centralized techniques, a distributed minimum domain heuristic is proposed, where domain sizes are evaluated approximately in distributed environments. Further improvement including look-ahead strategy is also combined with concurrent search.

The remainder of the paper is organized as follows. Section 2 introduces the basic concepts of distributed constraint satisfaction problems and the mechanism of concurrent search. A dynamic variable ordering heuristic for distributed constraint satisfaction problems is discussed in Section 3. In Section 4, the improved concurrent search algorithm is described in detail. Section 5 gives some results of experiments. From those experiments on random constraint satisfaction problems based on phase transition theory, it can be concluded that the newly proposed algorithm can outperform classical concurrent search. Section 6 is the conclusions.

2 Preliminaries

2.1 Problem Definitions

Constraint satisfaction problem (constraint network) is defined as a triplet $\langle X, D, C\rangle$, where $X = \{X_1, X_2, \cdots, X_n\}$ is a finite set of variables, each X_i taking its values in a finite domain Dx_i, and finite domain set $D = \{Dx_1, Dx_2, \cdots, Dx_n\}$, the Cartesian product of all the domains in D, denoted as $D_s = Dx_1 \times Dx_2 \times \cdots \times Dx_n$, is the search space of the problem, and C is a finite constraint set. A constraint c in C is defined on a set of variables $V(c)$ and restricts the combinations of values that can be taken by variables in $V(c)$. A CSP is a binary CSP, if all the c in C are binary ($|V(c)| = 2$). A binary CSP can be described by a graph, where vertexes represent variables in the CSP and edges between two vertexes represent constraints.

An assignment (a label) is a pair $\langle var, val\rangle$, where var is a variable in X and val is a value from var's domain that is assigned to it. A partial assignment (a compound label) is a set of assignments. A solution to a CSP is a partial assignment which includes all variables in X, and all constraints in C are satisfied. Value v_1 supports value v_2, if compound label $(\langle X_1, v_1\rangle, \langle X_2, v_2\rangle)$, where and $v_1 \in Dx_1$ and $v_2 \in Dx_2$, satisfies the constraint between variable X_1 and variable X_2. The set of all values in that are supported by v_1 is denoted as $S(X_1, v_1, X_2) = \{val | val \in Dx_2 \wedge v_1 \ supports \ val\}$.

A distributed constraint satisfaction problem is composed of a set of agents $A = \{A_1, A_2, \ldots, A_k\}$, each agent has a variable set $X_{A_1}, X_{A_2}, \ldots, X_{A_k}$, where $X_{A_1} \cup X_{A_2} \cup \ldots \cup X_{A_k} = X$. In each agent there is a local constraint network. Agents are connected by global constraints. A solution to a DisCSP must satisfy all the local constraints in agents and all the constraints among agents. The task of algorithms for solving DisCSPs is to find a solution or all solutions by agents' cooperating working. This paper uses the DisCSP model that is made in [4] and [8], where some assumptions are listed as follows.

1. All agents hold exactly one variable.
2. All constraints in the CSP are binary.
3. Agents communicate by sending messages. An agent can send messages to all the other agents.
4. Every agent holds constraints in which it is involved.

2.2 Concurrent Search

Concurrent search is a family of algorithms, where search processes run simultaneously. There are several current partial assignments that pass among agents, each CPA corresponds to a search process in backtrack based concurrent search. Each agent must maintain the information of received CPAs, such as CPA's ID and sub-search space. It sends the CPA to other agent after a successful assignment, and performs backtrack when all values have been tried just like centralized algorithms. Agents deal with different CPAs at the same time, and idle times of agents decrease compared to synchronous backtrack.

Concurrent search divides a certain variable's domain to get sub-search spaces of CPAs. This variable will be divided into n portions if there are n CPAs, and Cartesian product of a portion and domains of other variables represents a CPA's search space.

Example 1. A binary constraint satisfaction problem is shown in Fig. 1. It is composed of five variables and six binary constraints. If there are 2 CPAs, the domain of X_1 is divided into two portions {0,1} and {2,3}, which correspond to CPA$_1$ and CPA$_2$ respectively. The search spaces of CPA$_1$ and CPA$_2$ are shown in Table 1. This approach can be performed in only one agent, so the performance of dividing does not impact other agents.

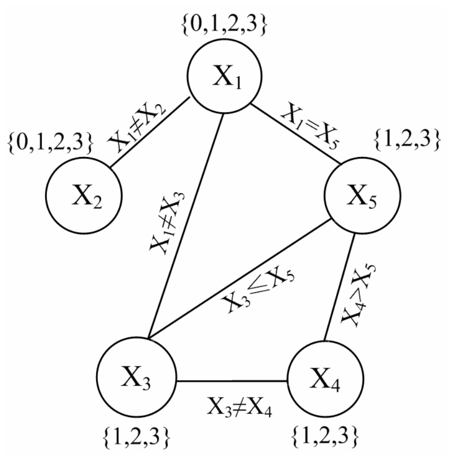


Fig. 1. A binary constraint satisfaction problem

At the beginning of searching, Initial Agent (IA) creates CPAs by dividing domain of its variable, then searching starts. For the sake of balancing overhead of CPAs, an approach to splitting CPAs dynamically is proposed in [8], and this approach uses a simple heuristic to decide whether the CPA should be split. When the number of a CPA's steps up to a fixed threshold, agents divides this CPA into several CPAs, and new created CPAs perform the same actions as old CPAs to search their sub-spaces until a solution is found or all search spaces are completed.

Many algorithms for solving centralized CSPs, such as backtrack and dynamic backtrack, can solve distributed CSPs under the framework of concurrent search,

Table 1. Search spaces of CPAs

SP ID	CPA ID	Search Space D$_s$
1	CPA$_1$	$\{0,1\} \times Dx_2 \times Dx_3 \times Dx_4 \times Dx_5$
2	CPA$_2$	$\{2,3\} \times Dx_2 \times Dx_3 \times Dx_4 \times Dx_5$

and experiments prove that algorithms based on concurrent search are more efficient than synchronous and asynchronous search [8].

3 Distributed Dynamic Variable Ordering

Backtrack search is an effective and complete algorithm for both classical CSPs and distributed CSPs. It instantiates variables one by one, and checks the new assignment after instantiating a variable. If there are conflicts in the partial assignment, it selects next value. Backtrack will be performed after all values are tried. When a variable is instantiated without conflicts, it chooses next variable from uninstantiated variables. When all variables are assigned, it gets a solution. In order to upgrade the efficiency of backtrack, constraint propagation is used after instantiating a variable. During this processing, domains of uninstantiated variables are filtered, and values without support will be deleted from their domains. As a result, dynamic variable ordering heuristics can choose the right variable to instantiate. However, the process of constraint propagation can't be fit for the distributed environment, and domains of uninstantiated variables can't be reduced. So concurrent backtrack search gets no interest from minimum domain heuristic.

Concurrent backtrack search uses random variable ordering technique. When a variable is instantiated, the next variable is selected randomly from the uninstantiated variable set. As a result, CPAs have different orders passing among agents, and agents' overheads are balanced, but it loses the effect of dynamic variable ordering.

In order to improve the efficiency of concurrent backtrack search, the improved algorithm calculates domain sizes approximately without processing of consistency checks. CPAs bring their approximate domain sizes to other agents, so this method can be performed in distributed environments. When a variable is assigned, domains of uninstantiated variables that have constraints with it will be reduced directly, and domains of other variables are impacted indirectly. According to this character, approximate domain sizes can be calculated by direct impacts.

First, it is easy to compute support set of each value in each domain, and support degree is defined as

$$U(X_i, val, X_j) = \begin{cases} |S(X_i, val, X_j)|/|Dx_j| & X_i \text{ and } X_j \text{ have a constraint} \\ 1 & \text{no constraints between } X_i \text{ and } X_j \end{cases}$$

Where the meaning of $U(X_i, val, X_j)$ is the proportion of values in Dx_j supported by *val* in Dx_i. 1 is assigned to U when there are no constraints between X_i and X_j.

Next, approximate domain sizes, denoted as $R(X)$, are evaluated. Assume a CPA has been assigned t times and sent to the agent including X_a, *Inst* is a set of current instantiated variables, it evaluates domain sizes approximately by

$$R^t_{Inst}(X_j) = |Dx_j| \times \prod_{x_i \in Inst} U(X_i, val(X_i), X_j) \ , \ X_j \in X - Inst$$

where $R^t_{Inst}(X_j)$ is approximate domain size of X_j under the set *Inst* after t assignment, and $val(X_i)$ is the assigned value to X_i.

After t+1 assignment, X_a is instantiated, it must update $R^{t+1}_{Inst}(X)$ from $R^t_{Inst}(X)$ in the received CPA. The relationships between $R^{t+1}_{Inst}(X)$ and $R^t_{Inst}(X)$ are

$$R^{t+1}_{Inst+\{X_a\}}(X_j) = R^t_{Inst}(X_j) \times U(X_a, val(X_a), X_j) \ , \ X_j \in X - Inst - \{X_a\}$$

At last, the smallest domain size is selected from $\{R^{t+1}_{Inst+\{X_a\}}(X_j) | X_j \in X - Inst - \{X_a\}\}$. Assume R_m is the minimum domain size and X_m is the corresponding variable, the CPA with $\{R^{t+1}_{Inst+\{X_a\}}(X_j) | X_j \in X - Inst - \{X_a\} - \{X_m\}\}$ is passed to the agent including X_m for t+2 assignment.

In Example 1, after X_5 is assigned 1, the CPA's domain sizes are $R^1_{Inst}(X_2) = 4$, $R^1_{Inst}(X_3) = 1$, $R^1_{Inst}(X_4) = 2$. Agent including X_1 receives this CPA, it assigns 1 to X_1, and then domain sizes are updated to $R^2_{Inst}(X_2) = 3$, $R^2_{Inst}(X_3) = 2/3$, $R^2_{Inst}(X_4) = 2$. X_3 is chosen as the next variable.

Each agent can access constraints in which its variable is involved, and support degrees are constants during searching, so they can be calculated in the local environment, and initialized before searching in order to avoid reduplicate computation. Because each agent initializes support degrees independently, messages between agents are not needed.

4 Improved Concurrent Backtrack Search Algorithm

Though it is difficult to perform look-ahead strategy in distributed environment, a simple look-ahead method can be combined with concurrent search by checking support degrees. When a value is assigned to a variable, support degrees of this variable are checked, if there exists a support degree to some uninstantiated variable is zero, next value will be tried instead of sending this CPA to next agent because the variable without this variable's support can't assign a consistent value in its domain. Function Lookahead (see Fig. 2) returns false if there exists a zero support degree and true is returned if no zero support degrees.

```
Function Lookahead($X_a$,$V_a$)
Input: the variable $X_a$ involved in this agent & its assigned value.
for each uninstantiated variable $X_u$
    if( $S(X_a, V_a, X_u)$ is zero)
        return false
    endif
endfor
return true
```

Fig. 2. Function of look-ahead strategy

With those improvements discussed above, a new concurrent backtrack search, which is called ConcBT_I, is proposed. Its sub-procedure Assign_Value (see Fig. 3) tries to assign values to a variable. At the same time, it computes approximate domain sizes. After a value is assigned, sizes of domains will be updated, and domain sizes before updating must be stored together with CPA's basic information (CPA's ID and splitting information) in the agent. When the CPA backtracks, the agent restores those information for next value. According to those updated sizes of domains, the variable that has the smallest domain size is selected, to whose agent CPA is sent.

```
Procedure Assign_Value
Input: received CPA, the variable $X_a$ involved in this agent
for each untried value in $Dx_a$
    set $X_a$ to $V_a$ in CPA
    if ( CPA is consistent & Lookahead($X_a$,$V_a$) )
        if(all the variable in CPA is instantiated)
            announce solution & send(STOP)
            return
        else
            update_domain_sizes(CPA)
            find the variable $X_m$ which has the smallest domain
            send(CPA, the agent that contains $X_m$)
            return
        endif
    endif
endfor
unassign $X_a$ in CPA
if (initial agent)
    announce no_solution & send(STOP)
else
    send(BACKTRACK, last_assignee)
endif
return
```

Fig. 3. Procedure of assigning values

Fig. 4 describes main procedure of the improved concurrent backtrack search. When a message is received, the agent identifies the type of this message. Assignment will be done if message type is CPA or BACKTRACK. If an agent finds a solution, it will notify all agents to stop searching.

5 Experimental Results

All experiments follow methods in [8] and [9]:

1. Randomly generated CSPs[13] are used to test algorithms. This benchmark is widely used in testing general algorithms for solving CSPs. Four parameters are taken into account in a binary random CSP: n is the number of variables, m is the domain size of each variable, p_1 is density of constraint graph, and p_2 is tightness of each constraint. Given the parameters $\langle n, m, p_1, p_2 \rangle$, there are $p_1 \times n \times (n-1)/2$ binary constraints and each constraint has $p_2 \times m^2$ inconsistent pairs of values. All test instances are generated by Random Uniform CSP Generator[1].

[1] http://www.lirmm.fr/∼bessiere/generator.html

```
Procedure ConcBT_I
Input:the variable $X_a$ involved in this agent & local constraints set $C$
Initialize support degrees
stop←false
if(initial agent)
    Create & initialize CPAs
endif
while(not stop)
    mess←wait & receive a message
    switch mess.type
        case: CPA
            store_domain_sizes (mess.CPA)
            Assign_Value(mess.CPA, $X_a$)
        case: BACKTRACK
            restore_domain_sizes (mess.CPA)
            Assign_Value (mess.CPA, $X_a$)
        case: STOP
            stop←true
endwhile
return
```

Fig. 4. The main procedure of the improved concurrent backtrack search

2. The number of non-concurrent constraint checks (NCCCs)[8] is counted to evaluate efficiency of algorithms. Similar to Lamport's logical clocks [14], every agent holds a local counter of constraint checks, and every CPA carries the agent's local counter to another agent. When this agent receives a message, it updates its local counter to the larger one between the current local counter and received CPA's counter. The number of non-concurrent constraint checks is reported if an agent finds a solution. The measure of non-concurrent constraint checks is implementation independent, and experiments can't be impacted by data structures and implementation details. In addition, assume there are no message delays [15].

The approach of dynamic splitting CPAs is added to ConcBT and ConcBT_I, in which steps_limit is fixed to 35. Instances are generated under $\langle 15, 10, 0.7, p_2\rangle$. For each p_2, 50 instances are generated. We report in Table 2 average numbers of NCCCs, as well as average messages (MSGs) sent and average CPAs created, on those 50 instances.

From Table 2, it is concluded that ConcBT_I is more efficient than ConcBT, especially for hard problems in the phase transition[16] areas, where p_2 is from 0.3 to 0.5. Bytes of total messages in ConcBT_I are slightly more than those in ConcBT when problems are easy to solve because of domain information passing with CPAs, but when problems become harder, both bytes of total messages and total CPAs needed in ConcBT_I are less than those in ConcBT. In [8], there is a concurrent search algorithm based on dynamic backtrack, which is called ConcDB. From Fig. 12 in [8], we can learn that it needs over 20000 non-concurrent constraint checks when p_2 is 0.4. However, when ConcBT_I is performed, the number of non-concurrent constraint checks is only about 15000.

Furthermore, we test the impact of the steps_limit on the ConcBT_I. Experiments have been done on the hardest instances and the parameters are $\langle 15, 10, 0.7, 0.4\rangle$. Steps_limit varies from 20 to 50. The results indicate (see

Table 2. Results of ConcBT and ConcBT_I

	NCCCs		MSGs		CPAs	
P_2	ConcBT	ConcBT_I	ConcBT	ConcBT_I	ConcBT	ConcBT_I
0.1	121	215	1067	1546	5	5
0.2	275	364	1589	2047	5.58	5.42
0.3	9350	2028	110577	24308	157.34	28.22
0.4	98151	15343	1339043	208404	1781.5	218.78
0.5	12467	2089	182911	19360	234.64	22.16
0.6	3298	518	40618	3744	53.4	5.84
0.7	1352	223	13712	1378	19.28	5
0.8	778	144	7889	558	12.02	5
0.9	389	106	3541	181	7.38	5

Table 3. The impact of steps_limit

Steps_limit	NCCCs	MSGs	CPAs
20	14704	215373	361.06
25	14923	212063	297.02
30	15179	209942	251.72
35	15348	208153	218.54
40	15555	207060	193.16
45	15713	205921	173
50	15881	204916	156

Table 3) that the number of NCCCs slightly decreases when steps_limit becomes smaller at the same time the bytes of total messages increase slightly.

6 Conclusions

Heuristic strategy holds the balance in algorithms for solving CSPs. Dynamic variable ordering can analyze domain sizes under current partial assignments and then decide the next variable to be instantiated. Minimum domain heuristic based on fail first principle is the most efficient method in dynamic variable ordering. In addition, look-ahead technique can reduce search space. In this paper, dynamic variable ordering and look-ahead technique are applied to concurrent backtrack search for solving distributed CSPs, where support degrees to variables under a partial assignment are introduced to evaluate domain sizes of unfixed variables. Experiments on randomly generated DisCSPs demonstrate the effiency of the new concurrent backtrack search. Especially for phase transition areas, ConcBT_I is about 6 times faster than ConcBT. Moreover, bytes of total messages have been improved.

Acknowledgments. This work is supported by the National Natural Science Foundation of China under Grant No. 60473003, the National Research Foundation for the Doctoral Program of Higher Education of China under Grant

No. 20050183065 and Program for New Century Excellent Talents in University (NCET).

References

1. Amilhastre, J., Fargier, H., Marquis, P.: Consistency restoration and explanations in dynamic CSPs—Application to configuration. Artificial Intelligence 135, 199–234 (2002)
2. Liu, J.M., Han, J., Tang, Y.Y.: Multi-Agent oriented constraint satisfaction. Artificial Intelligence 136, 101–144 (2002)
3. Dechter, R.: Constraint Processing. Morgan Kaufmann Publishers, San Francisco (2003)
4. Yokoo, M., Hirayama, K.: Algorithms for Distributed Constraint Satisfaction: A Review. Autonomous Agents and Multi-Agent Systems (3), 185–207 (2000)
5. Yokoo, M., Durfee, E.H., Ishida, T., Kuwabara, K.: Distributed constraint satisfaction for formalizing distributed problem solving. In: Proc. 12th IEEE Int. Conf. Distributed Comput. Syst., pp.614–621 (1992)
6. Solotorevsky, G., Gudes, E.: Solving a real-life time tabling and transportation problem using distributed CSP techniques. In: Freuder, E.C. (ed.) CP 1996. LNCS, vol. 1118, pp. 123–131. Springer, Heidelberg (1996)
7. Brito, I.: Synchronous, Asynchronous and Hybrid Algorithms for DisCSP. In: Wallace, M. (ed.) CP 2004. LNCS, vol. 3258, p. 791. Springer, Heidelberg (2004)
8. Zivan, R., Meisels, A.: Concurrent search for distributed CSPs. Artificial Intelligence 170, 440–461 (2006)
9. Zivan, R., Meisels, A.: Concurrent backtrack search for discsps. In: Proc. FLAIRS 2004, Miami, FL, pp. 776–781 (May 2004)
10. Bacchus, F., Vanrun, P.: Dynamic variable ordering in CSPs. In: Montanari, U., Rossi, F. (eds.) CP 1995. LNCS, vol. 976, Springer, Heidelberg (1995)
11. Edward, T.: Foundations of Constraint Satisfaction. Academic Press, London (1993)
12. Faltings, B., Yokoo, M.: Introduction: Special Issue on Distributed Constraint Satisfaction. Artificial Intelligence 161, 1–5 (2005)
13. Gent, I.P., MacIntyre, E., Prosser, P., Smith, B.M., Walsh, T.: Random constraint satisfaction: flaws and structure. Constraints 6(4), 345–372 (2001)
14. Lamport, L.: Time, clocks, and the ordering of events in distributed system. Comm. ACM. 95–114 (1978)
15. Zivan, R., Meisels, A.: Message delay and discsp search algorithms. In: DCR 2004. Proc. 5th Workshop on Distributed Constraints Reasoning, Toronto (2004)
16. Smith, B.M.: Locating the phase transition in binary constraint satisfaction problems. Artificial Intelligence 81, 155–181 (1996)

A Parameterized Local Consistency for Redundant Modeling in Weighted CSPs[⋆]

Y.C. Law, J.H.M. Lee, and M.H.C. Woo

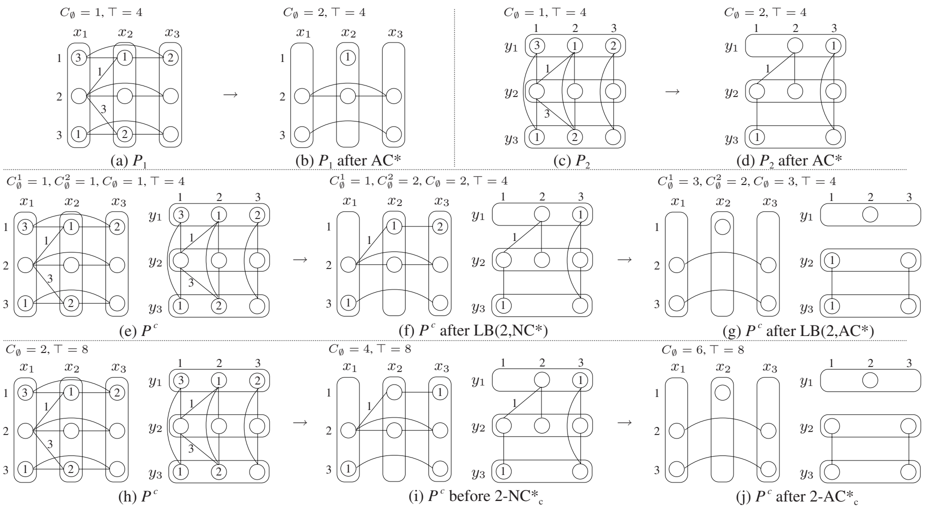
Department of Computer Science and Engineering
The Chinese University of Hong Kong, Shatin, N.T., Hong Kong
{yclaw,jlee,hcwoo}@cse.cuhk.edu.hk

Abstract. The *weighted constraint satisfaction problem* (WCSP) framework is a soft constraint framework which can model many real life optimization or over-constrained problems. While there are many local consistency notions available to speed up WCSP solving, in this paper, we investigate how to effectively combine and channel mutually redundant WCSP models to increase constraint propagation. This successful technique for reducing search space in classical constraint satisfaction has been shown non-trivial when adapted for the WCSP framework. We propose a parameterized local consistency LB(m,Φ), which can be instantiated with *any* local consistency Φ for single models and applied to a combined model with m sub-models, and also provide a simple algorithm to enforce it. We instantiate LB(2,Φ) with different state-of-the-art local consistencies AC*, FDAC*, and EDAC*, and demonstrate empirically the efficiency of the algorithm using different benchmark problems.

1 Introduction

The *weighted constraint satisfaction problem* (WCSP) framework is a well known soft constraint framework for modeling optimization or over-constrained problems. WCSPs are usually solved using backtracking branch and bound search incorporated with constraint propagation that helps reduce the search space. A crucial factor in the solving efficiency is therefore the level of constraint propagation during search. While many state-of-the-art local consistency notions, like AC* [1,2], FDAC* [3], and EDAC* [4], and their associated algorithms can effectively increase constraint propagation to speed up the search, another common approach is to use *redundant modeling* [5]. The technique, which has been applied successfully in classical constraint satisfaction, is to combine two different models of a problem using *channeling constraints*. Law, Lee, and Woo [6] showed that adapting redundant modeling to WCSPs is a non-trivial task; the node and arc consistency notions have to be refined so as to work on combined models. In this paper, instead of refining a particular local consistency, we propose a *parameterized* local consistency LB(m,Φ) and its associated enforcement algorithm. The advantages of our proposal are three-fold. First, the LB(m,Φ) consistency can be

[⋆] We thank the anonymous referees for their constructive comments. The work described in this paper was substantially supported by grants (CUHK4358/02E, CUHK4219/04E, and CUHK4132/07E) from the Research Grants Council of Hong Kong SAR.



$C_\emptyset = 1, \top = 4$
$C_\emptyset = 2, \top = 4$
(a) P_1
(b) P_1 after AC*
(c) P_2
(d) P_2 after AC*
$C^1_\emptyset = 1, C^2_\emptyset = 1, C_\emptyset = 1, \top = 4$
$C^1_\emptyset = 1, C^2_\emptyset = 2, C_\emptyset = 2, \top = 4$
$C^1_\emptyset = 3, C^2_\emptyset = 2, C_\emptyset = 3, \top = 4$
(e) P^c
(f) P^c after LB(2,NC*)
(g) P^c after LB(2,AC*)
$C_\emptyset = 2, \top = 8$
$C_\emptyset = 4, \top = 8$
$C_\emptyset = 6, \top = 8$
(h) P^c
(i) P^c before 2-NC*$_c$
(j) P^c after 2-AC*$_c$

Fig. 1. Enforcing node and arc consistencies on WCSPs $\mathcal{P}_1$, $\mathcal{P}_2$, and $\mathcal{P}^c$

instantiated with *any* local consistency Φ for single models and can be applied to a combined model with m sub-models. Second, the local consistency Φ used for instantiation needs not be refined. Third, enforcing LB(m,Φ) on a combined WCSP model $\mathcal{P}$ achieves *stronger* constraint propagation than enforcing Φ on any individual sub-model of $\mathcal{P}$ alone. We instantiate LB(2,Φ) with AC*, FDAC*, and EDAC* and perform experiments to confirm the feasibility and efficiency of our proposal. Empirical results show that the instantiations always achieve significantly better constraint propagation.

2 Background

WCSPs associate *costs* to tuples [7]. The costs are specified by a *valuation structure* $S(k) = ([0,\ldots,k], \oplus, \geq)$, where k is a natural number, $\oplus$ is defined as $a \oplus b = \min\{k, a+b\}$, and $\geq$ is the standard order among naturals. The minimum and maximum costs are denoted by $\bot = 0$ and $\top = k$ respectively. A binary WCSP is a quadruplet $\mathcal{P} = (k, \mathcal{X}, \mathcal{D}, \mathcal{C})$ with the valuation structure $S(k)$. $\mathcal{X} = \{x_1,\ldots,x_n\}$ is a finite set of *variables* and $\mathcal{D} = \{D_{x_1},\ldots,D_{x_n}\}$ is a set of finite *domains* for each $x_i \in \mathcal{X}$. An *assignment* $x \mapsto a$ in $\mathcal{P}$ is a mapping from variable x to value $a \in D_x$. A *tuple* is a set of assignments in $\mathcal{P}$. It is *complete* if it contains assignments of all variables in $\mathcal{P}$. $\mathcal{C}$ is a set of unary and binary constraints and a zero-arity constraint. A *unary constraint* involving variable x is a cost function $C_x : D_x \to \{0,\ldots,k\}$. A *binary constraint* involving variables x and y is a cost function $C_{x,y} : D_x \times D_y \to \{0,\ldots,k\}$. A *zero-arity constraint* $C_\emptyset$ is a constant denoting the lower bound of costs in $\mathcal{P}$. Fig. 1(a) shows a WCSP with variables $\{x_1, x_2, x_3\}$ and domains $\{1, 2, 3\}$. We depict the unary costs as labeled nodes and binary costs as labeled edges connecting two assignments. Unlabeled edges have $\top$ cost; $\bot$ costs are not shown for clarity.

The cost of a tuple $\theta = \{x_{i_j} \mapsto a_{i_j} \mid 1 \leq j \leq r\}$ in $\mathcal{P}$ is $\mathcal{V}(\theta) = C_\emptyset \oplus \sum_j C_{x_{i_j}}(a_{i_j}) \oplus \sum_{j<j'} C_{x_{i_j},x_{i_{j'}}}(a_{i_j}, a_{i_{j'}})$. If θ is complete and $\mathcal{V}(\theta) < \top$, θ is a *solution* of $\mathcal{P}$. Solving a WCSP is to find a solution θ with minimized $\mathcal{V}(\theta)$, which is NP-hard. The WCSP in Fig. 1(a) has an optimal solution $\{x_1 \mapsto 2, x_2 \mapsto 1, x_3 \mapsto 3\}$ with cost 3, as $C_\emptyset \oplus C_{x_1}(2) \oplus C_{x_2}(1) \oplus C_{x_3}(3) \oplus C_{x_1,x_2}(2,1) \oplus C_{x_1,x_3}(2,3) \oplus C_{x_2,x_3}(1,3) = 1 \oplus \bot \oplus 1 \oplus \bot \oplus 1 \oplus \bot \oplus \bot = 3$. A WCSP reduces to a classical CSP if each cost in the WCSP is either $\bot$ or $\top$. Two WCSPs are *equivalent* if they have the same variables and for every complete tuple θ, $\mathcal{V}(\theta)$ is the same for both WCSPs.

WCSPs can be solved by backtracking *branch and bound* (B&B) search that maintains some form of local consistency at each search node. A *local consistency* is simply some properties of a WCSP. *Enforcing* a local consistency Φ on a WCSP $\mathcal{P}$ means transforming $\mathcal{P}$ to an equivalent WCSP $\mathcal{P}'$ that satisfies the properties specified by Φ. $\mathcal{P}'$ is usually simplified in the sense that either some domain values of the variables are pruned or the lower bound is increased. In the following, we give the definitions of two common local consistencies NC* [1,2] and AC* [1,2] for WCSPs.

Definition 1. *A variable x_i in a WCSP $\mathcal{P}$ is* node consistent *(NC*) if (1) for all values $a \in D_{x_i}$, $C_\emptyset \oplus C_{x_i}(a) < \top$ and (2) there exists a value $a \in D_{x_i}$ such that $C_{x_i}(a) = \bot$. A WCSP $\mathcal{P}$ is NC* if every variable in $\mathcal{P}$ is NC*.*

Definition 2. *Given a constraint C_{x_i,x_j} in a WCSP $\mathcal{P}$, a value $b \in D_{x_j}$ is a* support *for a value $a \in D_{x_i}$ if $C_{x_i,x_j}(a,b) = \bot$. A variable x_i is* arc consistent *(AC*) if every value $a \in D_{x_i}$ has a support in every constraint C_{x_i,x_j}. A WCSP $\mathcal{P}$ is* arc consistent *(AC*) if every variable in $\mathcal{P}$ is AC*.*

NC* can be enforced by sending some costs from the *unary* constraints to $C_\emptyset$ and pruning the node-inconsistent values [1,2]. Similarly, AC* can be enforced by sending some costs from the *binary* constraints to the unary constraints, and then relying on NC* to further move the costs to $C_\emptyset$ or prune any values [1,2]. Thus, $C_\emptyset$ can be seen as a lower bound of costs of a WCSP. For example, the WCSP in Fig. 1(b) is AC* and equivalent to the one in Fig. 1(a). It has some domain values pruned and the lower bound $C_\emptyset$ is increased. In backtracking B&B search, whenever $C_\emptyset$ is increased to $\top$, we cannot continue to extend a tuple to obtain a solution, and hence a backtrack (or sometimes called a fail) is triggered. Also, whenever a solution θ is found, the $\top$ value will be set to $\mathcal{V}(\theta)$ to continue the search, ensuring that the next solution found must have a better cost than θ. At the end of the search, the last found solution is optimal.

3 A Parameterized Local Consistency for Redundant Modeling

In this section, we first describe how we can obtain redundant WCSP models. Then, we suggest a method to combine mutually redundant models into one model. A parameterized local consistency LB(m,Φ), which maintains the same local consistency Φ on each sub-model, and its enforcement algorithm are proposed for the suggested combined model. A bijective channel function, implemented using table lookup, is used to map assignments in one sub-model to the other. Theoretical comparisons are made between the proposed consistency and an existing approach for redundant modeling.

3.1 Obtaining a Redundant Model

Deriving multiple classical CSP models for the same problem is common, although not trivial. It is even more difficult to obtain an alternative model in WCSP since each problem solution is associated with a cost and we have to ensure the same cost distribution on the solutions of the redundant WCSP models. Two WCSPs $\mathcal{P}_1$ and $\mathcal{P}_2$ are *mutually redundant* [6] if (1) there is a bijection g between the two sets of all solutions of $\mathcal{P}_1$ and $\mathcal{P}_2$ and (2) for every solution θ of $\mathcal{P}_1$, the associated costs of solution θ of $\mathcal{P}_1$ and solution $g(\theta)$ of $\mathcal{P}_2$ are the same, i.e., $\mathcal{V}(\theta) = \mathcal{V}(g(\theta))$.

Based on these two requirements, Law et al. [6] proposed *generalized model induction* that generates mutually redundant *permutation WCSPs* from a given one. In a permutation WCSP, the variables in a solution must take all-different values. Given a WCSP $\mathcal{P} = (k, \mathcal{X}, \mathcal{D}_\mathcal{X}, \mathcal{C}_\mathcal{X})$. A *channel function* maps assignments in $\mathcal{P}$ to those in another set of variables. If $\mathcal{P}$ is a permutation WCSP, without loss of generality, we always have the bijective channel function $f(x_i \mapsto j) = y_j \mapsto i$. The constraints $\mathcal{C}_\mathcal{Y}$ in the *induced model* $\mathcal{P}' = (k, \mathcal{Y}, \mathcal{D}_\mathcal{Y}, \mathcal{C}_\mathcal{Y})$ are defined such that for $1 \leq a, i \leq n$, $C_{y_a}(i) = C_{x_i}(a)$, and for $1 \leq a, b, i, j \leq n$, $C_{y_a,y_b}(i,j) = C_{x_i,x_j}(a,b)$ if $i \neq j$; and $C_{y_a,y_b}(i,j) = \top$ otherwise. Note that the induced model $\mathcal{P}'$ must be a permutation WCSP, since $C_{y_a,y_b}(i,i) = \top$ for all $1 \leq a, b, i \leq n$. Fig. 1(c) shows the induced model $\mathcal{P}_2$ of $\mathcal{P}_1$ in Fig. 1(a). In the example, we have, say, the unary cost $C_{y_1}(2) = C_{x_2}(1) = 1$ and the binary cost $C_{y_2,y_3}(1,2) = C_{x_1,x_2}(2,3) = 3$.

3.2 Combining Mutually Redundant Models

Given a problem, we can always formulate it into different mutually redundant WCSP models. Redundant modeling is a common technique to take advantages of each model by combining them into one model using *channeling constraints*, which are constraints that set forth the relationship between the variables in any two models.

Consider m mutually redundant models $\mathcal{P}_i = (k, \mathcal{X}_i, \mathcal{D}_i, \mathcal{C}_i)$ for $1 \leq i \leq m$ of the same problem, we propose to connect them using a set of channeling constraints $\mathcal{C}^c$ to give a *combined model* $\mathcal{P}^c = (k, \bigcup_i \mathcal{X}_i, \bigcup_i \mathcal{D}_i, \bigcup_i \mathcal{C}_i \cup \mathcal{C}^c)$. In addition to a single $C_\emptyset$ for a combined model $\mathcal{P}^c$, we now associate each sub-model $\mathcal{P}_i$ with a *local zero-arity constraint* $C^i_\emptyset$ to denote the local lower bound of costs in $\mathcal{P}_i$. Since each sub-model $\mathcal{P}_i$ has its own $C^i_\emptyset$, enforcing NC* sends the costs of the unary constraints in sub-model $\mathcal{P}_i$ to $C^i_\emptyset$ of $\mathcal{P}^c$. Besides, $\mathcal{C}^c$ are hard constraints connecting sub-models in $\mathcal{P}^c$. For example, a channeling constraint is typically of the form $x_i = p \Leftrightarrow y_j = q$. It has the cost function $C_{x_i,y_j}(a,b) = \bot$ if $a = p \Leftrightarrow b = q$; and $C_{x_i,y_j}(a,b) = \top$ otherwise. In addition, the value k remains unchanged for each sub-model $\mathcal{P}_i$ in $\mathcal{P}^c$. The lower bound $C_\emptyset$ of $\mathcal{P}^c$ takes the maximum value of all local lower bounds $C^i_\emptyset$ of $\mathcal{P}_i$ (i.e., $C_\emptyset = \max_i\{C^i_\emptyset\}$). Based on the combined model, we denote a complete tuple in a sub-model $\mathcal{P}_i$ as a *semi-complete tuple* θ_i. Thus, similar to $C_\emptyset$, the cost of a complete tuple θ in $\mathcal{P}^c$ takes the maximum value of all semi-complete tuples θ_i in $\mathcal{P}_i$ (i.e., $\mathcal{V}(\theta) = \max_i\{\mathcal{V}(\theta_i)\}$). Fig. 1(e) shows the combined model $\mathcal{P}^c$ of $\mathcal{P}_1$ and $\mathcal{P}_2$ in Figs. 1(a) and 1(c) respectively.

```
function LB($m, \Phi, \mathcal{P}^c$)
1. repeat
2.     for each sub-model $\mathcal{P}_s$ of $\mathcal{P}^c$ do enforce $\Phi$ on $\mathcal{P}_s$;
3.     for each pair of sub-models $\mathcal{P}_s, \mathcal{P}_t$ of $\mathcal{P}^c$ $(s \neq t)$ do
4.         for each $x_{s,i} \in \mathcal{X}_s$ do
5.             for each $a \in D_{x_{s,i}}$ do
6.                 if $b \notin D_{x_{t,j}}$ where $x_{t,j} \mapsto b = f_{s,t}(x_{s,i} \mapsto a)$ then remove $a$ from $D_{x_{s,i}}$;
7. until $\mathcal{P}^c$ remains unchanged;
```

Fig. 2. Algorithms for enforcing LB(m,Φ)

3.3 Enforcing Consistency on Combined Models

In redundant modeling, mutually redundant models are connected using a set $\mathcal{C}^c$ of channeling constraints, which are mainly used to transmit pruning and variable instantiation information between sub-models. If we rely on the generic AC* or other local consistency algorithms to propagte the channeling constraints, it will cause a large overhead on the execution. Consider m mutually redundant WCSP sub-models in $\mathcal{P}^c$. With a set of channeling constraints of the form $x = a \Leftrightarrow y = b$ connecting sub-models $\mathcal{P}_i$ and $\mathcal{P}_j$, we can define a bijective channel function that maps assignments in $\mathcal{P}_i$ to those in $\mathcal{P}_j$, giving, for example, $f(x \mapsto a) = y \mapsto b$. A channel function can be implemented using table lookup in $O(1)$ time and can be used to aid the propagation of the channeling constraints more efficiently. During constraint propagation, when a value $a \in D_x$ is pruned from sub-model $\mathcal{P}_i$, we can use the channel function to immediately know that the corresponding value $b \in D_y$ should be pruned as well. Similarly, when a variable x in $\mathcal{P}_i$ is bound to the value a, according to the channel function, we can also know that the corresponding variable y in $\mathcal{P}_j$ should be bound to the value b.

Given a combined model $\mathcal{P}^c$ with m mutually redundant sub-models $\mathcal{P}_i$. We can enforce local consistency Φ on each sub-model $\mathcal{P}_i$ of $\mathcal{P}^c$ and use the channel function to transmit instantiation and pruning information between $\mathcal{P}_i$ to ensure that the bijective mapping between assignments of any two sub-models $\mathcal{P}_i$ and $\mathcal{P}_j$ for $1 \leq i < j \leq m$ is maintained. Based on these ideas, we proposed a parameterized local consistency LB(m,Φ) for combined models $\mathcal{P}^c$ with m mutually redundant sub-models $\mathcal{P}_i$. Note that Φ can be any local consistency that can be applied to a single WCSP model.

Definition 3. *Let $\mathcal{P}^c$ be a combined model of m mutually redundant sub-models $\mathcal{P}_s$ for $1 \leq s \leq m$, Φ be a local consistency, and $f_{s,t}$ be a bijective channel function from assignments of $\mathcal{P}_s$ to assignments of $\mathcal{P}_t$ for all $1 \leq s < t \leq m$. $\mathcal{P}^c$ is said to be* LB(m,Φ) *if: (1) all sub-models $\mathcal{P}_s$ are Φ, and (2) for all assignments $x_{s,i} \mapsto a$ of $\mathcal{P}_s$, $a \in D_{x_{s,i}} \Leftrightarrow b \in D_{x_{t,j}}$, where $f_{s,t}(x_{s,i} \mapsto a) = x_{t,j} \mapsto b$.*

Consider the combined model $\mathcal{P}^c$ in Fig. 1(e). It is not LB(2,AC*) since both sub-models $\mathcal{P}_1$ and $\mathcal{P}_2$ are not AC*. After enforcing AC* on each sub-model and sharing the pruning information between sub-models, Fig. 1(g) gives an equivalent combined model $\mathcal{P}^{c\prime}$, which is now LB(2,AC*).

LB(m,Φ) can be enforced using a simple algorithm shown in Fig. 2. In this algorithm, we enforce Φ on each sub-model (line 2). This ensures that all sub-models $\mathcal{P}_s$ satisfy the Φ property (condition 1). For condition 2, if there is a value $a \in D_{x_{s,i}}$ being pruned in one sub-model $\mathcal{P}_s$, the corresponding value $b \in D_{x_{t,j}}$ obtained via the channel function will also be pruned in other sub-models $\mathcal{P}_t$ for $1 \leq t \leq m$ and $s \neq t$ (lines 3–6). The algorithm repeats until there are no more changes in any sub-models, and $\mathcal{P}^c$ is then made LB(m,Φ). Since each sub-model $\mathcal{P}_s$ has its local lower bound $C^s_\emptyset$, unary constraints are projected towards its own $C^s_\emptyset$ when enforcing NC*. For example, unary constraints in $\mathcal{P}_1$ are projected to $C^1_\emptyset$ and those in $\mathcal{P}_2$ are projected to $C^2_\emptyset$. During constraint propagation, when the global lower bound $C_\emptyset$ of the combined model $\mathcal{P}^c$ reaches the global upper bound $\top$ (i.e., $C_\emptyset = \max_s\{C^s_\emptyset\} = \top$), this means that there exists at least one sub-model $\mathcal{P}_s$ in which its local lower bound $C^s_\emptyset$ is increased to $\top$, and we cannot extend a tuple of this sub-model to obtain a solution. Since all the sub-models are mutually redundant to each other, the other sub-models will also lead to failure. Therefore, a backtrack is triggered in the search. The following theorem states that the algorithm in Fig. 2 enforces LB(m,Φ).

Theorem 1. *Let $\mathcal{P}^c$ be a combined model of m WCSP sub-models $\mathcal{P}_s$ for $1 \leq s \leq m$, and Φ be any local consistency. The LB$(m, \Phi, \mathcal{P}^c)$ algorithm transforms $\mathcal{P}^c$ into an equivalent combined model $\mathcal{P}^{c\prime}$.*

Proof. When a combined model $\mathcal{P}^c$ is passed to the LB$(m, \Phi, \mathcal{P}^c)$ algorithm, enforcing Φ on each sub-model $\mathcal{P}_i$ transforms $\mathcal{P}_i$ to an equivalent sub-model $\mathcal{P}'_i$. Furthermore, the mutual redundancy of two sub-models $\mathcal{P}_s$ and $\mathcal{P}_t$ guarantees that if value b is not in $D_{x_{t,j}}$, then value a must not be in the domain of $x_{s,i}$, where $x_{t,j} \mapsto b = f_{s,t}(x_{s,i} \mapsto a)$. Thus, removing value a from $D_{x_{s,i}}$ in line 6 will not remove any values that belong to a solution of the combined model $\mathcal{P}^c$. Hence, upon termination of the algorithm, the transformed model $\mathcal{P}^{c\prime}$ is equivalent to the input model $\mathcal{P}^c$. □

Following Debruyne and Bessiere [8], we define some notions to compare the propagation strength of two local consistencies Φ_1 and Φ_2. Φ_1 is said to be *stronger than* [8] Φ_2 if in any WCSP in which Φ_1 holds, then Φ_2 holds. Φ_1 is said to be *strictly stronger than* [8] Φ_2 if (1) Φ_1 is stronger than Φ_2 and (2) there exists a WCSP where Φ_2 holds but Φ_1 does not hold.

Theorem 2. *Let $\mathcal{P}^c$ be a combined model of m sub-models $\mathcal{P}_s$ for $1 \leq s \leq m$. Enforcing LB(m,Φ) on $\mathcal{P}^c$ is strictly stronger than enforcing Φ on any $\mathcal{P}_s$.*

Proof. By definition 3, $\mathcal{P}^c$ is LB(m,Φ) if all sub-models $\mathcal{P}_s$ for $1 \leq s \leq m$ are Φ. This shows that LB(m,Φ) is stronger than Φ. To show strictness, consider the model in Fig. 1(e) which is a combined model of $\mathcal{P}_1$ and $\mathcal{P}_2$ in Figs. 1(a) and 1(c) respectively. Enforcing AC* on $\mathcal{P}_1$ and $\mathcal{P}_2$ individually yields the models in Figs. 1(b) and 1(d) respectively. However, the combined model of these two sub-models is not LB(2,AC*). In fact, enforcing LB(2,AC*) results in the model in Fig. 1(g). Hence the result. □

Note that unlike classical CSPs, enforcing AC* on a WCSP can result in more than one possible outcome, depending on the order of the domain values to be pruned and the constraints to be handled in an algorithm [2]. Therefore, although we have the "strictly stronger" notion, when comparing a LB(m,AC*) combined model and a AC* sub-model, we cannot guarantee that the domain of a variable in the combined model must be a subset of that in the single sub-model. Nonetheless, such theoretical comparison is still worthwhile as it shows that enforcing one local consistency can generally prune more domain values than enforcing another.

3.4 Comparison with Existing Work

Given m mutually redundant WCSPs $\mathcal{P}_s = (k_s, \mathcal{X}_s, \mathcal{D}_{\mathcal{X}s}, \mathcal{C}_s)$ for $1 \leq s \leq m$, Law et al. [6] suggested another way to form a combined model. Instead of taking the same k and $\top$ as each sub-model has, the combined model $\mathcal{P}^c = (\sum_s k_s, \bigcup_s \mathcal{X}_s, \bigcup_s \mathcal{D}_s, \bigcup_s \mathcal{C}_s \cup \mathcal{C}^c)$ uses the values k and $\top$ which are the sum of all values k_s and $\top_s$ respectively of its sub-models $\mathcal{P}_s$. For example, Fig. 1(h) gives a combined model $\mathcal{P}^c$ of two mutually redundant models $\mathcal{P}_1$ and $\mathcal{P}_2$ in Figs. 1(a) and 1(c) respectively, with $C_\emptyset = 1 \oplus 1 = 2$ and $\top = 4 \oplus 4 = 8$. Besides, $\mathcal{P}^c$ has only one lower bound $C_\emptyset$; there are no individual lower bound for sub-models. Any cost that is sent from the unary constraints in any sub-model all goes to $C_\emptyset$. Since $C_\emptyset$ and $\top$ are shared among sub-models in $\mathcal{P}^c$, the local consistency has to be refined for combined models. Law et al. [6] proposed new notions of node consistency m-NC$_{\mathrm{c}}^*$ and arc consistency m-AC$_{\mathrm{c}}^*$ to transmit pruning and cost movement information between sub-models. In our proposed combined model, sub-models have their own local lower bound; pruning information are transmitted via the channel function. Thus, refinements of local consistencies are not required.

When enforcing m-NC$_{\mathrm{c}}^*$ and m-AC$_{\mathrm{c}}^*$, not only the instantiation and pruning information but also the cost projection information is transmitted between sub-models in the combined model. Transmitting cost projection information can further discover and remove more node inconsistent values or increase the global lower bound. Thus, enforcing 2-NC$_{\mathrm{c}}^*$ and 2-AC$_{\mathrm{c}}^*$ achieves more constraint propagation than enforcing LB(2,NC*) and LB(2,AC*) respectively.

Theorem 3. *Let $\mathcal{P}^c$ be a combined model of m mutually redundant sub-models $\mathcal{P}_s$ for $1 \leq s \leq m$. Enforcing m-NC$_{\mathrm{c}}^*$* (resp. *m-AC$_{\mathrm{c}}^*$) on $\mathcal{P}^c$ is* strictly stronger *than enforcing LB(m,NC*)* (resp. *LB(m,AC*)) on $\mathcal{P}^c$.*

Due to space limitation, we do not provide the formal definitions of m-NC$_{\mathrm{c}}^*$ and m-AC$_{\mathrm{c}}^*$ and thus the proof. Instead, we give an example to demonstrate the theorem. Consider the problem in Fig. 1. Figs. 1(f) and 1(i) give two combined WCSPs after enforcing LB(2,NC*) and 2-NC$_{\mathrm{c}}^*$ respectively. We can observe that enforcing 2-NC$_{\mathrm{c}}^*$ achieves more domain prunings and a greater lower bound (i.e., $C_\emptyset = 4 > C_\emptyset^1 \oplus C_\emptyset^2 = 1 \oplus 2 = 3$). Similarly, the WCSP in Fig. 1(j) is 2-AC$_{\mathrm{c}}^*$ and has a greater lower bound than the LB(2,AC*) WCSP in Fig. 1(g).

4 Experiments

To evaluate the effectiveness and efficiency of the combined models, we implement our approach in ToolBar[1], a branch and bound WCSP solver maintaining local consistencies at each search tree node. Three benchmarks, n-queens problem, knight's tour problem,

[1] Available at http://carlit.toulouse.inra.fr/cgi-bin/awki.cgi/ToolBarIntro.

and Langford's problem are modeled as WCSPs to test our approach. Comparisons are made among AC* [1,2], FDAC* [3], and EDAC* [4] on a single model $\mathcal{P}$, 2-AC$^*_c$ [6] on a combined model $\mathcal{P}^c$, and LB(2,AC*), LB(2,FDAC*), and LB(2,EDAC*) on a combined model $\mathcal{P}^c$ proposed in this paper. All combined models $\mathcal{P}^c$ contain a single model $\mathcal{P}$ and its induced model $\mathcal{P}'$, generated automatically using generalized model induction [6], as sub-models.

The experiments are run on a Sun Blade 2500 (2 $\times$ 1.6GHz US-IIIi) workstation with 2GB memory. We use the *dom/deg* variable ordering heuristic which chooses the variable with the smallest ratio of domain size to future degree. Values are chosen using the *smallest-cost-first* and the *dual-smallest-domain-first* [5] heuristics on $\mathcal{P}$ and $\mathcal{P}^c$ respectively. The initial $\top$ value provided to the solver is n^2, where n is the number of variables in a model. We generate ten random models for each soft instance and report the average number of fails (i.e., the number of backtracks occurred in solving a model) and CPU time in seconds to find the first optimal solution. In the tables, the first column shows the problem instances; those marked with "*" have a $\perp$ optimal cost. The subsequent columns show the results of enforcing various local consistencies on either $\mathcal{P}$ or $\mathcal{P}^c$. The best number of fails and CPU time among the results for each instance are highlighted in bold. A cell labeled with "-" denotes a timeout after two hours.

Table 1(a) shows the experimental results of soft n-queens problem. The n-queens problem is to place n queens on a $n \times n$ chessboard so that no two queens are placed on the same row, same column, or same diagonal. To model the problem into a WCSP $\mathcal{P}_1 = (\mathcal{X}, \mathcal{D}_\mathcal{X}, \mathcal{C}_\mathcal{X})$, we use n variables $\mathcal{X} = \{x_1, \ldots, x_n\}$. Each variable x_i denotes the row position of queen i in column i of the chessboard. The domains of the variables are thus $\{1, \ldots, n\}$. The constraints can be expressed using these variables accordingly.

Since the n-queens problem has many solutions, we assert preferences among the solutions by assigning to each allowed binary tuple a random cost from $\perp$ to $\top$ inclusive for soft instances. From the results, we observe that solving a combined model $\mathcal{P}^c$ achieves fewer number of fails and mostly faster runtime than solving a single model $\mathcal{P}$, which means that 2-AC$^*_c$, LB(2,AC*), LB(2,FDAC*), and LB(2,EDAC*) do more prunings than AC*, FDAC*, and EDAC* and reduce more search space. LB(2,AC*) is the most efficient for the larger instances. Although 2-AC$^*_c$ is stronger than LB(2,AC*), it gives a longer runtime than LB(2,AC*) even it achieves smaller number of fails in solving the soft instances. This is mainly due to the overhead of cost projection transmitting between sub-models in 2-AC$^*_c$. LB(2,EDAC*) always has the fewest number of fails since EDAC* is stronger than AC* and FDAC*. However, the runtime of LB(2,EDAC*) is not the fastest, as the extra prunings cannot compensate for the overhead of EDAC* over AC*. According to Larrosa (personal communication), the implementation of EDAC* is optimized for small variable domains. The same applies to the timing of AC* and EDAC* on $\mathcal{P}$. We perform experiments on the classical n-queens problem as well, and the results are similar to those of the soft cases. Due to space limitation, we omit the details.

Tables 1(b) and 1(c) show the results of classical and soft knight's tour problem. This problem is to find a sequence of moves by a knight on a $m \times n$ chessboard so that each square on the chessboard is traversed exactly once. We model this problem into a WCSP $\mathcal{P}$ using mn variables $\mathcal{X} = \{x_1, \ldots, x_{mn}\}$. Each variable x_i denotes a square on the

Table 1. Experimental results on solving n-queens and knight's tour problems

(a) Soft n-queens problem

	AC*		FDAC*		EDAC*		2-AC$_c^*$		LB(2,AC*)		LB(2,FDAC*)		LB(2,EDAC*)	
n	fail	time	fail	time	fail	time	fail	time	fail	time	fail	time	fail	time
15*	5608	**0.46**	5044	0.50	4959	0.63	2759	0.58	2803	0.58	2546	0.65	**2488**	0.77
16*	10611	**0.96**	9566	1.03	9382	1.29	5340	1.23	5427	1.21	4873	1.37	**4769**	1.61
17*	17369	**1.65**	15139	1.80	14703	2.25	7291	1.85	7429	1.81	6668	2.08	**6493**	2.46
18*	58375	**6.23**	51073	6.76	49309	8.29	26370	7.73	26762	7.48	24479	8.49	**23880**	9.95
19*	81022	**9.20**	70179	10.05	67341	12.41	32921	9.91	33681	9.60	29197	11.07	**27919**	12.81
20*	172220	**21.65**	150939	23.50	145062	28.94	94256	32.44	95941	31.16	86345	35.83	**83442**	42.09
21*	535145	73.28	463225	79.78	441403	97.35	178540	65.39	182731	**62.39**	159087	72.03	**151279**	83.29
22*	1287717	196.07	1130132	211.09	1078297	257.76	470117	200.77	478708	**188.64**	432133	214.43	**415874**	249.10
23*	4780028	810.60	4256142	868.88	4076254	1060.10	1274332	617.77	1294032	**575.07**	1183741	650.64	**1143417**	755.17
24*	11079154	2042.66	9928478	2182.87	9518203	2637.39	3256536	1738.86	3307770	**1602.52**	3029570	1806.44	**2921698**	2081.07

(b) Classical knight's tour problem

	AC*		FDAC*		EDAC*		2-AC$_c^*$		LB(2,AC*)		LB(2,FDAC*)		LB(2,EDAC*)	
(m,n)	fail	time	fail	time	fail	time	fail	time	fail	time	fail	time	fail	time
(3,5)	672	0.07	672	0.07	672	0.10	**210**	0.08	**210**	0.08	**210**	**0.06**	**210**	0.08
(3,6)	4416	0.45	4416	0.53	4416	0.70	**816**	0.36	**816**	0.34	**816**	**0.29**	**816**	0.33
(3,7)*	545	**0.06**	545	0.07	545	0.10	**158**	0.08	**158**	0.08	**158**	0.08	**158**	0.08
(3,8)*	2657	0.33	2657	0.38	2657	0.50	**732**	0.35	**732**	0.33	**732**	**0.29**	**732**	0.33
(3,9)*	11535	1.66	11535	1.94	11535	2.61	**2005**	1.29	**2005**	1.21	**2005**	**1.07**	**2005**	1.20
(3,10)*	72183	11.79	72183	13.74	72183	18.72	**10628**	8.25	**10628**	7.71	**10628**	**6.72**	**10628**	7.49
(3,11)*	13225	2.17	13225	2.46	13225	3.39	**1878**	1.79	**1878**	1.60	**1878**	**1.31**	**1878**	1.48
(3,12)*	2349445	467.80	2349445	550.14	2349445	743.09	**212657**	269.15	**212657**	240.85	**212657**	**201.84**	**212657**	228.67
(3,13)	766731	172.10	766731	202.52	766731	274.80	**49684**	94.08	**49684**	83.70	**49684**	**68.33**	**49684**	74.73
(3,14)	-	-	-	-	-	-	-	-	**3679309**	6816.84	**3679309**	**5550.03**	**3679309**	6249.42
(3,15)	1498214	367.45	1498214	426.99	1498214	588.97	**58889**	147.85	**58889**	134.74	**58889**	**107.39**	**58889**	118.52
(4,5)*	336	**0.03**	336	**0.03**	336	0.04	**77**	0.04	**77**	**0.03**	**77**	0.04	**77**	0.04
(4,6)*	7914	0.68	7914	0.76	7914	1.04	**1415**	0.56	**1415**	0.53	**1415**	**0.46**	**1415**	0.51
(4,7)*	40344	4.19	40344	4.62	40344	6.55	**5526**	2.70	**5526**	2.51	**5526**	**2.07**	**5526**	2.30
(4,8)*	1183823	145.18	1183823	161.48	1183823	228.35	**76661**	53.58	**76661**	49.00	**76661**	**39.74**	**76661**	43.63
(4,9)	462506	67.28	462506	74.85	462506	105.82	**33575**	31.86	**33575**	28.07	**33575**	**23.35**	**33575**	25.66
(4,10)	29317520	5166.23	29317520	5769.60	-	-	**971162**	1469.12	**971162**	1316.55	**971162**	**1034.83**	**971162**	1144.33

(c) Soft knight's tour problem

	AC*		FDAC*		EDAC*		2-AC$_c^*$		LB(2,AC*)		LB(2,FDAC*)		LB(2,EDAC*)	
(m,n)	fail	time	fail	time	fail	time	fail	time	fail	time	fail	time	fail	time
(3,5)	685	**0.08**	672	0.11	683	0.19	**210**	0.10	**210**	0.09	**210**	0.09	**210**	0.12
(3,6)	4482	0.52	4418	0.81	4463	1.46	**816**	0.39	**816**	**0.37**	**816**	0.43	**816**	0.62
(3,7)	21005	2.94	13629	3.85	13785	5.74	4576	2.13	4750	**2.09**	3571	2.42	**3536**	3.49
(3,8)	112979	19.93	61182	24.30	60967	34.70	21452	**11.46**	22971	11.48	14320	12.57	**14022**	17.96
(3,9)	679347	140.38	341322	166.53	339664	238.65	113032	72.89	121562	**72.20**	73220	79.67	**71996**	118.24
(3,10)	3822955	952.94	1720858	1061.83	1697349	1505.17	511042	431.53	557168	**423.39**	299257	440.19	**294434**	651.79
(3,11)	-	-	-	-	-	-	2319839	2587.78	2590939	2579.96	1236203	**2432.43**	**1216082**	3579.21
(4,5)	45516	4.86	27468	5.85	27374	8.72	5515	1.96	5719	**1.94**	4165	2.14	**4111**	3.05
(4,6)	863270	116.09	415083	129.63	407198	187.90	98151	**42.53**	104256	42.71	61805	43.33	**60522**	63.25
(4,7)	12899196	2335.48	5005880	2339.55	4831495	3248.17	1344421	815.54	1468226	814.30	719927	**742.79**	**702569**	1085.02

chessboard and the domains of the variables $\{1,\ldots,mn\}$ denote the order of a move sequence. There are constraints to ensure that the knight takes a valid move (i.e., moves either one square horizontally and two squares vertically, or two squares horizontally and one square vertically). Similar to the n-queens problem, we soften the problem by assigning a random cost to each allowed binary tuple for soft instances. The random cost is assigned from $\perp$ to mn inclusively. In this problem, the induced model $\mathcal{P}'$ is a better model than $\mathcal{P}$. Therefore, we use $\mathcal{P}'$ as a basic model instead of $\mathcal{P}$.

For both classical and soft instances, enforcing local consistencies on $\mathcal{P}^c$, similar to the n-queens problem, achieves smaller number of fails and shorter runtime than those on $\mathcal{P}$. For the classical instance $(3,14)$, using AC*, FDAC*, EDAC*, and even 2-AC$_c^*$ cannot solve the problem before timeout, but using LB(2,AC*), LB(2,FDAC*), and LB(2,EDAC*) can. Among the local consistencies on $\mathcal{P}^c$, LB(2,FDAC*) always achieves the fastest runtime. LB(2,AC*), LB(2,FDAC*), LB(2,EDAC*), and 2-AC$_c^*$ on

$\mathcal{P}^c$ all result in the same number of fails because in the classical cases, EDAC* and FDAC* degenerate to AC*. In soft cases, LB(2,AC*) and 2-AC$_c^*$ have similar runtime but 2-AC$_c^*$ has a slightly smaller number of fails. LB(2,EDAC*) achieves the smallest number of fails in most instances, while LB(2,FDAC*) has the shortest runtime in large and difficult instances.

For the Langford's problem[2] (which we do not present the experimental data), using LB(2,Φ) is not always faster than using 2-AC$_c^*$, although both are much better than using single models. In classical cases, LB(2,AC*) and LB(2,FDAC*) have a slightly faster runtime than 2-AC$_c^*$ for the large instances, but they are slightly slower than 2-AC$_c^*$ in the soft cases. Although LB(2,Φ) does not always perform better than 2-AC$_c^*$, it has the flexibility of choosing *any* local consistency for single models to solve a problem.

5 Conclusion

Combining mutually redundant WCSP models is a non-trivial task. In this paper, we have proposed a parameterized local consistency LB(m,Φ) that is specifically designed for combined models to increase constraint propagation. One important advantage of our proposal is that LB(m,Φ) can be used with *any* number of sub-models and *any* existing or future local consistencies that are targeted for solving single models, making our proposal highly flexible. Experimental results confirm that LB(2,Φ) performs well when instantiated with the state-of-the-art AC*, FDAC*, and EDAC*. The search space is significantly reduced when compared with using single models. LB(2,Φ) is also competitive with, if not better than, 2-AC$_c^*$ in the benchmarks.

Redundant modeling for WCSPs is a new concept which has much scope for future work. In classical CSPs, it is common to have one model in integer variables while another in set variables. We can investigate if WCSPs with different types of variables can be combined and solved. In this paper, we focus on permutation problems. We are interested in knowing how generalized model induction can generate induced models of non-permutation WCSPs and if the concept of redundant modeling can be extended to combined models connected with non-bijective channeling constraints.

References

1. Larrosa, J.: Node and arc consistency in weighted CSP. In: Proc. of AAAI 2002, pp. 48–53 (2002)
2. Larrosa, J., Schiex, T.: Solving weighted CSP by maintaining arc consistency. Artificial Intelligence 159(1-2), 1–26 (2004)
3. Larrosa, J., Schiex, T.: In the quest of the best form of local consistency for weighted CSP. In: Proc. of IJCAI 2003, pp. 239–244 (2003)
4. de Givry, S., Heras, F., Zytnicki, M., Larrosa, J.: Existential arc consistency: Getting closer to full arc consistency in weighted CSPs. In: Proc. of IJCAI 2005, pp. 84–89 (2005)
5. Cheng, B.M.W., Choi, K.M.F., Lee, J.H.M., Wu, J.C.K.: Increasing constraint propagation by redundant modeling: an experience report. Constraints 4(2), 167–192 (1999)

[2] Prob024 in CSPLib, available at http://www.csplib.org.

6. Law, Y.C., Lee, J.H.M., Woo, M.H.C.: Speeding up weighted constraint satisfaction using redundant modeling. In: Sattar, A., Kang, B.H (eds.) AI 2006. LNCS (LNAI), vol. 4304, pp. 59–68. Springer, Heidelberg (2006)
7. Schiex, T., Fargier, H., Verfaillie, G.: Valued constraint satisfaction problems: hard and easy problems. In: Proc. of IJCAI 1995, pp. 631–637 (1995)
8. Debruyne, R., Bessière, C.: Some practicable filtering techniques for the constraint satisfaction problem. In: Proc. of IJCAI 1997, pp. 412–417 (1997)

Modeling and Solving Semiring Constraint Satisfaction Problems by Transformation to Weighted Semiring Max-SAT

Louise Leenen[1], Anbulagan[2], Thomas Meyer[3], and Aditya Ghose[4]

[1] DSL, SCSSE, University of Wollongong, Australia and DPSS, CSIR, South Africa
[2] NICTA* and ANU, Canberra, Australia
[3] Meraka Institute, CSIR, Pretoria, South Africa
[4] DSL, SCSSE, University of Wollongong, Australia

Abstract. We present a variant of the Weighted Maximum Satisfiability Problem (Weighted Max-SAT), which is a modeling of the Semiring Constraint Satisfaction framework. We show how to encode a Semiring Constraint Satisfaction Problem (SCSP) into an instance of a propositional Weighted Max-SAT, and call the encoding Weighted Semiring Max-SAT (WS-Max-SAT). The clauses in our encoding are highly structured and we exploit this feature to develop two algorithms for solving WS-Max-SAT: an incomplete algorithm based on the well-known GSAT algorithm for Max-SAT, and a branch-and-bound algorithm which is complete. Our preliminary experiments show that the translation of SCSP into WS-Max-SAT is feasible, and that our branch-and-bound algorithm performs surprisingly well. We aim in future to combine the natural flexible representation of the SCSP framework with the inherent efficiencies of SAT solvers by adjusting existing SAT solvers to solve WS-Max-SAT.

1 Introduction

The Semiring Constraint Satisfaction Problem (SCSP) framework is an approach to constraint satisfaction and optimization that generalises classical Constraint Satisfaction Problems (CSPs), Partial Constraint Satisfaction Problems, Fuzzy CSPs, Weighted CSPs, Probabilistic CSPs, and others (over finite domains) [1,2,3,4], while also permitting the specification of interesting new instances. Considerable interest has been shown in solving SCSPs. Complete methods exist [5,6,7] but are likely to require exponential time. Bistarelli *et al.* [8] present a prototype SCSP solver based on an incomplete local search method. Wilson [9] shows how to use decision diagrams to solve SCSPs but provides no implementation.

A significant amount of work has been devoted to solving propositional satisfiability (SAT) problems, and specifically the well-known maximum satisfiability problem (Max-SAT) [10,11]. There is continuing interest in translations between

* NICTA is funded through the Australian Government's *Backing Australia's Ability* initiative, in part through the Australian Research Council.

CSPs and SAT problems [12,13,14]. This prompted us to explore the application of methods for solving Max-SAT to SCSPs. We modify the support encoding of CSP to SAT by Gent [12] in order to translate SCSPs as a variant of the Weighted Max-SAT Problem, and call this encoding WS-Max-SAT. Our encoding results in propositional clauses whose structure can be exploited in our algorithms to solve WS-Max-SAT: a local search algorithm that is a modification of the GSAT algorithm for solving Max-SAT [15], and a branch-and-bound (BnB) algorithm.

We show how to formulate a SCSP as a WS-Max-SAT, present a local search and a BnB algorithm, and some experimental results. A comparison of our BnB algorithm with CONFLEX [3], a fuzzy CSP solver, shows that our algorithm performs surprisingly well, and is significantly faster than CONFLEX.

2 Semiring Constraint Satisfaction Problems

The SCSP framework is a popular approach for the representation of constraint satisfaction and in particular, partial constraint satisfaction problems [1].

Definition 1. *A c-semiring is a tuple $S = \langle A, +, \times, \mathbf{0}, \mathbf{1}\rangle$ such that*

i) A is a set with $\mathbf{0}, \mathbf{1} \in A$;
ii) $+$ is defined over (possibly infinite) sets of elements of A as follows a) for all $a \in A$, $\sum(\{a\}) = a$; b) $\sum(\emptyset) = \mathbf{0}$ and $\sum(A) = \mathbf{1}$; c) $\sum(\bigcup A_i, i \in I) = \sum(\{\sum(A_i), i \in I\})$ for all sets of indices I (flattening property);
iii) $\times$ is a commutative, associative, and binary operation with $\mathbf{1}$ as unit element and $\mathbf{0}$ as absorbing element; iv) $\times$ distributes over $+$ (i.e., for any $a \in A$ and $B \subseteq A$, $a \times \sum(B) = \sum(\{a \times b, b \in B\})$).

The elements of the set A are the preference values to be assigned to tuples of values of the domains of constraints. A partial ordering $\leqslant_S$ over the set A can be defined: $\alpha \leqslant_S \beta$ iff $\alpha + \beta = \beta$.

Definition 2. *A constraint system is a 3-tuple $CS = \langle S_p, D, V\rangle$, where $S_p = \langle A_p, +_p, \times_p, \mathbf{0}, \mathbf{1}\rangle$ is a c-semiring, V is an ordered finite set of variables, and D is a finite set containing the allowed values for the variables in V. Given $CS = \langle S_p, D, V\rangle$, where $S_p = \langle A_p, +_p, \times_p, \mathbf{0}, \mathbf{1}\rangle$, a constraint over CS is a pair $c = \langle def_c^p, con_c\rangle$ where $con_c \subseteq V$ is called the signature of the constraint, and $def_c^p : D^k \to A_p$ (k is the cardinality of con_c) is called the value of the constraint.*

Definition 3. *Given $CS = \langle S_p, D, V\rangle$, a SCSP over CS is a pair $P = \langle C, con\rangle$ where C is a finite set of constraints over CS and $con = \bigcup_{c \in C} con_c$. Assume $\langle def_{c_1}^p, con_c\rangle \in C$ and $\langle def_{c_2}^p, con_c\rangle \in C$ implies $def_{c_1}^p = def_{c_2}^p$.*

Definition 4. *Given $CS = \langle S_p, D, V\rangle$ with V totally ordered via $\preceq$, consider any k-tuple $t = \langle t_1, t_2, \ldots, t_k\rangle$ of values of D and two sets $W = \{w_1, \ldots, w_k\}$ and $W' = \{w'_1, \ldots, w'_m\}$ such that $W' \subseteq W \subseteq V$ and $w_i \preceq w_j$ if $i \leq j$ and $w'_i \preceq w'_j$ if $i \leq j$. Then the projection of t from W to W', written $t \downarrow^W_{W'}$, is defined as the tuple $t' = \langle t'_1, \ldots, t'_m\rangle$ with $t'_i = t_j$ iff $w'_i = w_j$.*

For $CS = \langle S_p, D, V\rangle$, $S_p = \langle A_p, +_p, \times_p, \mathbf{0}, \mathbf{1}\rangle$, and two constraints $c_1 = \langle def^p_{c_1}, con_{c_1}\rangle$ and $c_2 = \langle def^p_{c_2}, con_{c_2}\rangle$ over CS, their combination, $c_1 \otimes c_2$, is the constraint $c = \langle def^p_c, con_c\rangle$ with $con_c = con_{c_1} \cup con_{c_2}$ and $def^p_c(t) = def^p_{c_1}(t \downarrow^{con_c}_{con_{c_1}}) \times_p def^p_{c_2}(t \downarrow^{con_c}_{con_{c_2}})$. Let $\bigotimes C$ denote $c_1 \otimes ... \otimes c_m$ with $C = \{c_1, ..., c_m\}$.

Definition 5. *Given $CS = \langle S_p, D, V\rangle$, with $S_p = \langle A_p, +_p, \times_p, \mathbf{0}, \mathbf{1}\rangle$, a constraint $c = \langle def^p_c, con_c\rangle$ over CS, and a set I of variables ($I \subseteq V$), the projection of c over I, written $c \Downarrow I$, is the constraint $c' = \langle def^p_{c'}, con_{c'}\rangle$ over CS with $con_{c'} = I \cap con_c$ and $def^p_{c'}(t') = \sum_{\{t|t\downarrow^{con_c}_{I\cap con_c}=t'\}} def^p_c(t)$. Given a SCSP $P = \langle C, con\rangle$ over CS, the solution of P is a constraint defined as $Sol(P) = \bigotimes C$.*

Definition 6. *Given a SCSP problem $P = \langle C,\ con\rangle$, consider $Sol(P) = \langle def^p_c, con\rangle$. The abstract solution of P is the set $ASol(P) = \{\langle t,\ v\rangle \mid def^p_c(t) = v$ and there is no t' such that $v <_{S_p} def^p_c(t')\}$. Let $ASolV(P) = \{v \mid \langle t, v\rangle \in ASol(P)\}$.*

3 Max-SAT and Weighted Max-SAT

The SAT problem is to determine whether a given propositional formula has a model (is satisfiable) or is unsatisfiable. The maximum satisfiability problem is the optimisation variant of SAT: given a set of clauses, the problem is to find a variable assignment that maximises the number of satisfied clauses. In *Weighted* Max-SAT a weight is assigned to each clause and the goal becomes to maximise the total weight of the satisfied clauses. More formally, let $C = \{C_1, C_2, \ldots, C_m\}$ be the set of clauses that involve n Boolean variables $x_1, x_2, \ldots, x_n$ that can take the value *true* or *false*, and w_i with $i = 1, \ldots, m$, is the weight assigned to each clause. The objective function is defined as $f(x) = \sum_{i=1}^{m} w_i \cdot I(C_i)$, where $I(C_i)$ is one if and only if the clause C_i is satisfied, and otherwise is zero. When we refer to Max-SAT in the remainder of this paper we assume the weighted version.

Some of the best exact Max-SAT solvers [16,10,11,17,18] implement variants of a BnB algorithm: given a CNF formula θ, a BnB algorithm explores the search tree representing all possible assignments for θ in a depth-first manner. The lower bound (LB) of a node is the sum of the weights of all clauses satisfied by the current partial assignment plus an over-estimation of the sum of the weigths of all clauses that will become satisfied if the current assignment is completed. If LB $\leq$ UB (UB is the upper bound, or best solution found *so far*), the subtree below the current node is pruned, and the algorithm backtracks to a node at a higher level in the tree. If LB $>$ UB, the algorithm tries to find a better solution by extending the current partial solution by instantiating one more variable. The solution is the value of UB after the whole tree has been explored.

4 Weighted Semiring Max-SAT

In this section we show how to encode a SCSP into a WS-Max-SAT. Our encoding of a SCSP into a WS-Max-SAT is based on the support encoding of CSPs into Boolean SAT [12]. Every propositional clause in our encoding will receive a semiring value from the semiring associated with the SCSP as a weight.

For the remainder of this paper let $P = \langle C, con \rangle$ be a SCSP over a constraint system $CS = \langle S_p, D, V \rangle$, such that every variable $i \in V$ has $D = \{d_1, \ldots, d_d\}$ as its domain, and $S_p = \langle A_p, +_p, \times_p, \mathbf{0}, \mathbf{1} \rangle$. Assume the cardinality of V is n, the cardinality of C is m, and that there are s semiring values in the set A_p. Every constraint $c \in C$, $c = \langle def_c^p, con_c \rangle$ with r the cardinality of con_c and $con_c \subseteq V$.

For every SCSP variable $i \in V$, we have d WS-Max-SAT variables: the WS-Max-SAT variable $X_{i,j}$ is true if the SCSP variable i has the value $d_j \in D$ with $j = 1...d$. Our *WS-Max-SAT encoding* (encoding of P) has three kinds of clauses:

1. *at-least-one* clauses to ensure that each SCSP variable is assigned at least one value from its domain;
2. *at-most-one* clauses to ensure that each SCSP variable is assigned at most one value from its domain;
3. *support* clauses to represent all the possible value pairs (tuples) with their associated semiring values for every constraint. For every constraint we have at most s such clauses, and every clause receives the associated semiring value as its weight.

- For every variable $i \in V$ we have one *at-least-one* clause:
  $X_{i,1} \vee X_{i,2} \vee \ldots \vee X_{i,d}$.
- For every variable $i \in V$ we have $0.5[d(d-1)]$ *at-most-one* clauses: one clause, $\neg X_{i,v} \vee \neg X_{i,w}$, for every pair (d_v, d_w), where $1 \leq v < w \leq d$.
- For every constraint, we have at most s support clauses: one clause to represent all the tuples that have the same associated semiring value. Suppose $A_p = \{p_1, p_2, \ldots, p_s\}$, then every constraint $c \in C$ has at most s support clauses, $sup_c^{p_t}$ with $t = 1, \ldots, s$. A support clause $sup_c^{p_t}$ for a constraint $c = \langle def_c^p, con_c \rangle$ is a disjunction of the representation of all tuples $\langle X_{c_1,v_1}, \ldots, X_{c_r,v_r} \rangle$ (the conjunction of all these literals) that has the associated semiring value p_t, $c_i \in con_c$, and $v_i \in D$ for $i = 1 \ldots r$.

All the clauses except the support clauses have to be satisfied and we call these clauses the *hard* clauses. The minimum semiring value, $\mathbf{0}$, is assigned as the weight of the negation of each hard clause. If a hard clause is not satisfied by an assignment, this unsatisfied clause contributes the minimum semiring value to the total weight. Note that at most one support clause for each constraint can be satisfied by a truth assignment if we assume that the *at-most-one* and *at-least-one* clauses are all satisfied. We regard the support clauses as soft clauses: the weight assigned to each support clause is the combined semiring value of the tuples represented in that support clause.

Definition 7. *Given a SCSP P and a WS-Max-SAT encoding of P, let the set Sup contain the support clauses in the encoding and α be some truth assignment for the clauses in Sup. Then $t_{sol}^{\alpha} = \langle X_{1,val_1}, \ldots, X_{n,val_n} \rangle$ is a solution tuple for the encoding iff the following holds:*

- *for every constraint $c \in C$ exactly one of its support clauses, sup_c^k, ($1 \leq k \leq p_s$) is satisfied by α. This support clause represents a disjunction of tuples* *of this constraint, each as a conjunction of WS-Max-SAT variables, with exactly one tuple, $t_c = \langle X_{c_1}, \ldots, X_{c_r} \rangle$ with $c_i \in con_c$, $v_i \in D$, $i = 1 \ldots r$, and such that all $X_{c_i,v_i} = \text{true}$, and*

– *if c_i is equal to j $(j = 1, \ldots, n)$ then $v_i = val_j$.*

The semiring value associated with t^{α}_{sol} is $sr^{\alpha}_{sol} = sp_1 \times_p \ldots \times_p sp_m$ where $sp_c = k$ and sup^k_c is the support clause satisfied by assignment α for a constraint c.

Definition 8. *Given a SCSP P, let Cl be the set of clauses that represents the WS-Max-SAT encoding of P. Let $Cl_h = \langle cl_1, \ldots cl_h \rangle \subseteq Cl$ be the clauses that are satisfied by some truth assignment α. The WS-Max-SAT problem is to find max $f(\alpha) = \bigotimes_{i=1}^{h} w_i$, where $w_i \in A_p$ is the weight assigned to clause $i \in Cl_h$, $f(\alpha)$ is maximal with respect to the partial order $<_{S_p}$, and $t^{\alpha}_{sol} = \langle X_{1,val_1}, \ldots, X_{n,val_n} \rangle$ is the solution tuple for the encoding Cl. The ordered pair $\langle \langle val_1, \ldots, val_n \rangle, f(\alpha) \rangle \in$ ASol(P).*

Example 1. Suppose we have a SCSP $P = \langle \{c_1, c_2, c_3\}, V \rangle$ over $\langle S_p, D, V \rangle$, with $V = \{A, B, C\}$ and $D = \{1, 2, 3\}$. The three constraints are explicitly defined by $c_1 : A < B$, $c_2 : B < C$, and $c_3 : C < A$. The semiring is $S_p = \langle \{0, 5, 10\}, max, min, 0, 10 \rangle$. See the constraint definitions in Table 1. Note

Table 1. Constraint definitions

t	$c_1 : A < B$	$c_2 : B < C$	$c_3 : C < A$
$\langle 1,1 \rangle$	5	5	5
$\langle 1,2 \rangle$	10	10	10
$\langle 1,3 \rangle$	10	10	10
$\langle 2,1 \rangle$	0	0	0
$\langle 2,2 \rangle$	5	5	5

t	$c_1 : A < B$	$c_2 : B < C$	$c_3 : C < A$
$\langle 2,3 \rangle$	10	10	10
$\langle 3,1 \rangle$	0	0	0
$\langle 3,2 \rangle$	0	0	0
$\langle 3,3 \rangle$	5	5	5

that the first coordinate reflects the value of the first variable in a constraint and the second coordinate reflects the value of the second variable. For example, the preference value associated with the tuple $\langle 1, 3 \rangle$ for constraint c_1 is when A has the value 1 and B has the value 3. Tuples that satisfy the constraints get the highest preference value. Among the tuples that do not satisfy the constraints, we prefer those where the first coordinate equals the second coordinate. These tuples get a preference value of 5, and all the remaining tuples get the worst value, 0. The WS-Max-SAT encoding of the problem follows below.

At-least-one clauses: $A_1 \vee A_2 \vee A_3$, $B_1 \vee B_2 \vee B_3$, and $C_1 \vee C_2 \vee C_3$
At-most-one clauses: $\neg A_1 \vee \neg A_2$, $\neg A_2 \vee \neg A_3$, $\neg A_1 \vee \neg A_3$, $\neg B_1 \vee \neg B_2$, $\neg B_2 \vee \neg B_3$, $\neg B_1 \vee \neg B_3$, $\neg C_1 \vee \neg C_2$, $\neg C_2 \vee \neg C_3$, and $\neg C_1 \vee \neg C_3$
Support clauses:
$c^{10}_1 : [(A_1 \wedge B_2) \vee (A_1 \wedge B_3) \vee (A_2 \wedge B_3)]$, $c^5_1 : [(A_1 \wedge B_1) \vee (A_2 \wedge B_2) \vee (A_3 \wedge B_3)]$, $c^0_1 : [(A_2 \wedge B_1) \vee (A_3 \wedge B_1) \vee (A_3 \wedge B_2)]$, $c^{10}_2 : [(B_1 \wedge C_2) \vee (B_1 \wedge C_3) \vee (B_2 \wedge C_3)]$, $c^5_2 : [(B_1 \wedge C_1) \vee (B_2 \wedge C_2) \vee (B_3 \wedge C_3)]$, $c^0_2 : [(B_2 \wedge C_1) \vee (B_3 \wedge C_1) \vee (B_3 \wedge C_2)]$, $c^{10}_3 : [(C_1 \wedge A_2) \vee (C_1 \wedge A_3) \vee (C_2 \wedge A_3)]$,

$c_3^5 : [(C_1 \wedge A_1) \vee (C_2 \wedge A_2) \vee (C_3 \wedge A_3)]$,
$c_3^0 : [(C_2 \wedge A_1) \vee (C_3 \wedge A_1) \vee (C_3 \wedge A_2)]$

In our example, each support clause has its associated semiring value as its weight. The negation of all the remaining clauses are each given a weight of 0:

1. $\neg A_1 \wedge \neg A_2 \wedge \neg A_3$	2. $\neg B_1 \wedge \neg B_2 \wedge \neg B_3$	3. $\neg C_1 \wedge \neg C_2 \wedge \neg C_3$
4. $A_1 \wedge A_2$	5. $A_2 \wedge A_3$	6. $A_1 \wedge A_3$
7. $B_1 \wedge B_2$	8. $B_2 \wedge B_3$	9. $B_1 \wedge B_3$
10. $C_1 \wedge C_2$	11. $C_2 \wedge C_3$	12. $C_1 \wedge C_3$

We want to find an assignment that does not satisfy any of the (hard) clauses numbered 1 to 12. A truth assignment that satisfies one or more of the support clauses with an associated semiring value of 10 is preferred over one with lower semiring values. We can prove that our encoding is correct.

Theorem 1. *Let SCSP P be a set of clauses C_l containing a WS-Max-SAT encoding for P, and a truth assignment α.*

– *If $t_{sol}^{\alpha} = \langle X_{1,val_1}, \ldots, X_{n,val_n} \rangle$ is a solution tuple for the WS-Max-SAT encoding of P, then $t = \langle X_1, \ldots, X_n \rangle$ with $X_i = val_i$ for $i = 1, \ldots, n$ is a solution tuple for P.*
– *If sr_{sol}^{α} is the semiring value associated with t_{sol}^{α}, then the associated semiring value of t is sr_{sol}^{α}.*

5 Algorithms to Solve WS-Max-SAT

Most SAT and Max-SAT solvers require the propositional clauses to be in conjunctive normal form (CNF). In our encoding, the propositional clauses are highly structured and we do not have to convert them into CNF. We only allow truth assignments where the *at-least-one-variable* and *at-most-one-variable* clauses are satisfied in our implementation. This leaves only the support clauses that have to be checked for satisfiability: exactly one support clause can be satisfied for every constraint. We simply search for the support clause with the best associated semiring value for each constraint under the current truth assignment.

5.1 The GSAT Algorithm for WS-Max-SAT

GSAT [15] is a greedy algorithm that tries to maximise the number of satisfied clauses by selecting different variable assignments based on the *score* of a variable x under the current assignment α: this is defined as the difference between the weight of the clauses *un*satisfied by α and the assignment obtained by flipping x in α. GSAT is not a complete algorithm and can get stuck in local minima. GSAT starts with a random truth assignment. After a maximum number of allowed flips has been performed, a new random truth assignment is generated. This improves the probability to find a solution.

We have adjusted GSAT, as presented in Algorithm 1, to solve a WS-Max-SAT problem for SCSPs with binary constraints. The procedure *chooseVariable*

Algorithm 1. GSAT for WS-Max-SAT

```
Require: $C_l$ (WS-Max-SAT encoding); $W$, (set of weights); $S_p$, (semiring);
    $MaxFlips$, $MaxSteps$
 1: Initialise $\alpha_{Best} = \mathbf{0}$;
 2: for $i = 1$ to $MaxSteps$ do
 3:    Let $\alpha$ = a randomly generated truth assignment for $C_l$;
 4:    if $f(\alpha_{BEST}) <_{S_p} f(\alpha)$ then
 5:       $\alpha_{BEST} = \alpha$;
 6:    for $j = 1$ to $MaxFlips$ do
 7:       if $f(\alpha_{BEST}) <_{S_p} f(\alpha)$ then
 8:          $\alpha_{BEST} = \alpha$;
 9:       if $f(\alpha) = \mathbf{1}$ then
10:          return $\alpha$
11:       else
12:          $x$ = chooseVariable($C_l$,$\alpha$);
13:          $\alpha = \alpha$ with truth value of $x$ flipped;
14: return $\alpha_{BEST}$;
```

selects the next variable to be flipped by considering the *score* of each variable x in WS-Max-SAT. The score of a variable x in an assignment α is the f value a truth assignment α_x has if α_x is identical to α except for the truth value assigned to x. There may be more than one variable with a maximum score. In this case any variable with a maximal f value is chosen at random to be flipped. When a WS-Max-SAT variable $X_{i,j}$ is chosen to be flipped, we consider two cases:

- The current value of $X_{i,j} = true$ and has to be flipped to become $false$. Some variable $X_{i,l}$ with $l = 1, \ldots, d$ and $l \neq j$ is chosen at random to become *true*.
- The current value of $X_{i,j} = false$ and has to be flipped to become *true*. Some variable $X_{i,l}$ with $l = 1, \ldots, d$ and $l \neq j$ is currently *true* and is given the value $false$.

Example 2. Suppose we generate the following random truth assignment α_1: $A_1 = A_2 = B_1 = B_2 = C_1 = C_3 = false$ and $A_3 = B_3 = C_2 = true$. The following three clauses are satisfied: c_1^5, c_2^0, and c_3^{10} with $f(\alpha_1) = min(5, 0, 10) = 0$. The procedure *chooseVariable* finds that all flips result in f values of 0, so it randomly selects C_1 to be flipped. Now $C_1 = true$ and $C_2 = false$. In the next iteration it finds that C_3 has a score of 5 and it flips C_3: in this case the three clauses c_1^5, c_2^5, and c_3^5 with $f(\alpha_3) = min(5, 5, 5) = 5$.

5.2 The BnB Algorithm for WS-Max-SAT

We present a BnB algorithm to solve WS-Max-SAT, as sketched in Algorithm 2. The variables are ordered according to their membership in the signatures of the constraints. We place the variables in the signature of the first constraint on a queue, and then check whether these variables appear in the signatures of other constraints. If they do, the unlisted variables in the others constraints' signatures are also placed on the queue. Repeat this step until all constraints' signatures

Algorithm 2. BnB(NoInstantiated,LB)

```
Require: Input variables: $C_l$, (WS-Max-SAT encoding); $W$, (set of weights); $S_p$ (a
    semiring); $N$ (number of variables)
    Global variables: Queue (variables in order of instantiation); UB (upper bound)
    Value parameters NoInstantiated; LB (lower bound)
 1: if (NoInstantiated < N) then
 2:   var = pop Queue;
 3:   while (var_domain not empty) do
 4:     var_value = finds best value from var_domain;
 5:     var_domain = var_domain - var_value;
 6:     NewLB = lower bound for current node;
 7:     NewLB = $\times_p$(LB, NewLB);
 8:     if (UB $<_{S_p}$ NewLB) then
 9:       LB = NewLB;
10:       if (NoInstantiated = N-1) then
11:         UB = LB;
12:         Best-Solution = current assignment;
13:         if (UB = 1) then
14:           return 1;
15:         Prune all tuples with associated semiring value $\leq_{S_p}$ UB;
16:       if ( BnB(Noinstantiated+1,LB) = 1) then
17:         return 1;
18: return 0;
```

have been checked. Every variable has a *main constraint* which is the constraint where the variable has been identified to be placed on the queue. To find a value for a variable we search among the tuples of its main constraint for a tuple with a maximal associated semiring value under the current partial assignment. From this chosen tuple we get a value to instantiate the current variable.

After a variable has been instantiated, we calculate a lower bound value for the current node in the search tree, i.e. we calculate an estimated semiring value for the variable-value tuple of the current (in most cases, partial) assignment. The lower bound is computed by looking at constraints whose signatures have been entirely instantiated by the current (possibly partial) solution and combining the semiring values assigned to the projection of the current solution to the signatures of these constraints. For all the remaining constraints (i.e with uninstantiated variables), we estimate the maximal semiring value for the purpose of finding a lower bound. The upper bound is initialised with the worst semiring value and the lower bound with the best semiring value. The variable *var* is instantiated with the next variable from *Queue* and *var_domain* contains the domain values for that variable.

6 Experimental Setting and Results

We used an Intel Pentium 4 processor at 2.53GHz with 512MB RAM.

6.1 Results of GSAT-Based Algorithm

We solved three sets of randomly generated binary fuzzy CSPs where each set contains 100 instances. Each instance has 10 domain values. Instances in Set 1 has 80 variables and 10 constraints, in Set 2 100 variables and 10 constraints, and in Set 3 has 120 variables and 20 constraints.

In each set of 100 instances, 50 instances have a tightness of 70% and the other 50 instances have a tightness of 90%. A tightness (T) of x% means that (100-x)% of the possibles tuples have been assigned the maximum semiring value. All the problems have the following semiring values, $A_p = \{0, 0.3, 0.5, 0.8, 1\}$, the comparative operator is *max* and the combination operator is *min*. The results are in Table 2. In the third row (S=1) we show for how many instances the algorithm found a maximal solution (with a solution tuple that has a semiring value of 1). In the next rows we show for how many instances we found solution tuples with semiring values of 0.8, 0.5 and 0.3, respectively.

Table 2. Results for the GSAT-based algorithm

Set	1	1	2	2	3	3
T	70%	90%	70%	90%	70%	90%
S = 1	50	2	50	0	1	0
S = 0.8	0	48	0	49	33	0
S = 0.5	0	0	0	1	16	38
S = 0.3	0	0	0	0	0	12
steps	1.34	2.00	1.28	n/a	1	n/a
flips	452.4	836.5	438.5	n/a	329.0	n/a

There were no instances where a solution with the minimum semiring value was found. The last two rows contain information for those instances where the maximal solution were found: it shows on average when the solution was found. We show during which step, and after how many flips, the algorithm halted. Each step consists of a maximum of 1000 flips.

All the problems instances have maximal solutions apart from three instances in set 3 with tightness 90% (last column). These three instances have a best solution with a combined semiring value of 0.8. Our GSAT algorithm performs reasonably well for a relatively small number of steps and flips.

6.2 Results of BnB Algorithm

We solved two sets of randomly generated binary fuzzy CSPs with CONFLEX [3] and with our BnB algorithm. We used the CONFLEX executable for Windows downloaded from the CONFLEX website.[1] The 40 instances in Set 4 has 10 variables, 10 constraints, 10 domain values (20 instances with tightness 60% and 20 with tightness 80%). The 20 instances in Set 5 has 15 variables, 10

[1] http://www.inra.fr/internet/Departements/MIA/T//conflex/adressesConflex.html

Table 3. Average runtime for Sets 4 and 5 (in seconds)

Set	4	4	5	5
Tightness	60%	80%	60%	80%
CONFLEX	12.34	19.89	154.42	165.14
BnB	0.44	0.29	0.05	0.26

Table 4. Runtime for the BnB algorithm (in seconds)

Set	1	1	2	2	3	3	5
T	70%	90%	70%	90%	70%	90%	90%
Runtime	0.27	0.48	0.35	0.52	0.86	1.52	5.30

constraints, 10 domain values (10 instances with tightness 60% and 10 instances with tightness 80%). Table 3 gives the results.

We experienced difficulty in running larger problems on the Windows version of the CONFLEX software. However, it is clear from these experiments that our BnB algorithm is considerably faster than CONFLEX.

We also solved the three sets of problems of Table 2 and a fifth set containing 50 instances, where each instance has 150 variables, 30 constraints and a domain size of 20. All the problems in Set 5 have a maximal solution. See Table 4.

7 Conclusion and Future Work

We define a WS-Max-SAT problem by translating a SCSPs into propositional weighted Max-SAT problem. We present an incomplete GSAT-based algorithm to show that this translation is feasible to solve SCSPs. We also present a BnB algorithm that performs surprisingly well, outperforming the CONFLEX system. A next step is to investigate the adaptation of existing efficient Weighted Max-SAT algorithms to solve WS-Max-SAT. We also plan to test our BnB algorithm by solving benchmark problems.

References

1. Bistarelli, S., Montanari, U., Rossi, F.: Semiring-based constraint solving and optimization. Journal of the ACM 44(2), 201–236 (1997)
2. Wilson, M., Borning, A.: Hierarchical constraint logic programming. Journal of Logic Programming 16, 277–318 (1993)
3. Dubois, D., Fargier, H., Prade, H.: The calculus of fuzzy restrictions as a basis for flexible constraint satisfaction. In: Proc. of IEEE Conf on Fuzzy Syst. (1993)
4. Fargier, H., Lang, J.: Uncertainty in constraint satisfaction problems: a probabilistic approach. In: Moral, S., Kruse, R., Clarke, E. (eds.) ECSQARU 1993. LNCS, vol. 747, Springer, Heidelberg (1993)
5. Georget, Y., Codognet, P.: Compiling semiring-based constraint with clp(FD,s). In: Maher, M.J., Puget, J.F. (eds.) CP 1998. LNCS, vol. 1520, Springer, Heidelberg (1998)

6. Rossi, F., Pilan, I.: Abstracting soft constraints: Some experimental results. In: Proc. of Joint Annual Workshop of ERCIM Working Group on Constraints and the CologNET area on Constraint and Logic Programming (2003)
7. Bistarelli, S., Fargier, H., Montanari, U., Rossi, F., Schiex, T., Verfaillie, G.: Semiring-based CSPs and valued CSPs: Basic properties and comparison. Constraints 4, 199–240 (1999)
8. Bistarelli, S., Fung, S., Lee, J., Leung, H.: A local search framework for semiring-based constraint satisfaction problems. In: Soft 2003 (2003)
9. Wilson, N.: Decision diagrams for the computation of semiring valuations. In: IJCAI 2005 (2005)
10. Xing, Z., Zhang, W.: MaxSolver: An efficient exact algorithm for (weighted) maximum satisfiability. Artificial Intelligence 164(1-2), 47–80 (2005)
11. Alsinet, T., Manya, F., Planes, J.: Improved exact solver for weighted Max-SAT. In: Bacchus, F., Walsh, T. (eds.) SAT 2005. LNCS, vol. 3569, pp. 371–377. Springer, Heidelberg (2005)
12. Gent, I.P.: Arc consistency in SAT. In: ECAI 2002 (2002)
13. Walsh, T.: SAT v CSP. In: Dechter, R. (ed.) CP 2000. LNCS, vol. 1894, pp. 441–456. Springer, Heidelberg (2000)
14. Bennaceur, H.: The satisfiability problem regarded as a constraint satisfaction problem. In: ECAI 1996, pp. 155–159 (1996)
15. Selman, B., Levesque, H., Mitchell, D.: A new method for solving hard satisfiability problems. In: AAAI 1992, pp. 440–446 (1992)
16. Li, C.M., Manyà, F., Planes, J.: Detecting disjoint inconsistent subformulas for computing lower bounds for Max-SAT. In: AAAI 2006 (2006)
17. de Givry, S., Larrosa, J., Meseguer, S.T.: Solving Max-SAT as weighted CSP. In: Rossi, F. (ed.) CP 2003. LNCS, vol. 2833, Springer, Heidelberg (2003)
18. Larossa, J., Heras, F.: Resolution in Max-SAT and its relation to local consistency in weighted CSPs. In: IJCAI 2005 (2005)

Advances in Local Search for Satisfiability

Duc Nghia Pham, John Thornton, Charles Gretton, and Abdul Sattar

SAFE Program, Queensland Research Lab, NICTA and
Institute for Integrated and Intelligent Systems, Griffith University, QLD, Australia
{duc-nghia.pham,john.thornton,charles.gretton,abdul.sattar}@nicta.com.au

Abstract. In this paper we describe a stochastic local search (SLS) procedure for finding satisfying models of satisfiable propositional formulae. This new algorithm, gNovelty$^+$, draws on the features of two other WalkSAT family algorithms: R+AdaptNovelty$^+$ and G^{2}WSAT, while also successfully employing a dynamic local search (DLS) clause weighting heuristic to further improve performance.

gNovelty$^+$ was a Gold Medal winner in the random category of the 2007 SAT competition. In this paper we present a detailed description of the algorithm and extend the SAT competition results via an empirical study of the effects of problem structure and parameter tuning on the performance of gNovelty$^+$. The study also compares gNovelty$^+$ with two of the most representative WalkSAT-based solvers: G^{2}WSAT, AdaptNovelty$^+$, and two of the most representative DLS solvers: RSAPS and PAWS. Our new results augment the SAT competition results and show that gNovelty$^+$ is also highly competitive in the domain of solving *structured* satisfiability problems in comparison with other SLS techniques.

1 Introduction

The satisfiability (SAT) problem is one of the best known and well-studied problems in computer science, with many practical applications in domains such as theorem proving, hardware verification and planning. The techniques used to solve SAT problems can be divided into two main areas: complete search techniques based on the well-known Davis-Putnam-Logemann-Loveland (DPLL) algorithm [1] and stochastic local search (SLS) techniques evolving out of Selman and Kautz's 1992 GSAT algorithm [2]. As for SLS techniques, there have been two successful but distinct avenues of development: the WalkSAT family of algorithms [3] and the various dynamic local search (DLS) clause weighting approaches (e.g. [4]).

Since the early 1990s, the state-of-the-art in SAT solving has moved forward from only being able to solve problems containing hundreds of variables to the routine solution of problems with millions of variables. One of the key reasons for this success has been the keen competition between researchers and the public availability of the source code of the best techniques. Nowadays the SAT community organises regular competitions on large sets of benchmark problems and awards prizes to the best performing algorithms in different problem categories. In this paper we introduce the current 2007 SAT competition[1] Gold Medal winner in the satisfiable random problem category: gNovelty$^+$.

gNovelty$^+$ draws on the strengths of two WalkSAT variants which respectively came first and second in the random category of the 2005 SAT competition: R+AdaptNovelty$^+$ [5] and G^{2}WSAT [6]. In addition, gNovelty$^+$ connects the two branches of SLS (WalkSAT and DLS) by successfully employing a clause weighting heuristic to gain more efficiency.

In the remainder of the paper we describe in more detail the G^{2}WSAT and R+AdaptNovelty$^+$ techniques upon which gNovelty$^+$ is based. We then provide a full explanation of the execution of gNovelty$^+$ followed by a previously unpublished empirical evaluation of the algorithm. This evaluation examines the performance of gNovelty$^+$ on a range of structured problems and reports the effects of parameter tuning in comparison with two of the most representative WalkSAT based solvers: G^{2}WSAT and AdaptNovelty$^+$, and two of the most representative clause weighting solvers: RSAPS and PAWS. Finally we discuss these results and present our conclusions.

2 Existing Techniques

2.1 G^{2}WSAT

During the mid-1990s, Novelty [3] was considered to be one of the most competitive techniques in the WalkSAT family and was able to solve many hard problems faster than the best complete search techniques of that time. However, one key problem with Novelty is its deterministic variable selection,[2] which can cause it to loop indefinitely and fail to return a solution even where one existed [6,7]. The first practical solution to this problem was to add a random walk behaviour with a probability *wp* to the Novelty procedure [7]. More recently Li and Huang [6] revisited this problem and proposed a more diversified heuristic to weaken the determinism in Novelty. This new Novelty^{++} solver selects the least recently flipped variable for the next move with a *diversification probability dp*, otherwise it performs as Novelty. Li and Huang [6] further improved Novelty^{++} by integrating it with a new gradient-based greedy heuristic based on the count of current false clauses. The resulting G^{2}WSAT solver (depicted in the left hand side of Figure 1) always selects the most promising variable that, if flipped, will reduce the number of false clauses the most. If there is more than one variable with the best score, G^{2}WSAT selects the least recently flipped one, and if the search hits a local minimum, G^{2}WSAT performs as Novelty^{++} until it escapes.

[1] http://www.satcompetition.org

[2] Novelty deterministically selects the next move from the two best variables of a randomly selected false clause [3].

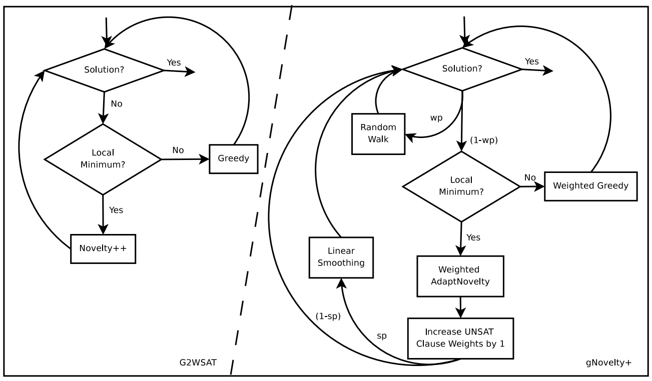


Fig. 1. Flow-chart comparison between G^{2}WSAT and gNovelty$^+$

2.2 R+AdaptNovelty$^+$

The performance of every WalkSAT variant critically depends on the setting of a noise parameter p which controls the level of greediness of the search [3,8]. This means that without extensive empirical tuning, the average case performance of a WalkSAT algorithm is quite poor. Hoos [8] addressed this problem by proposing an adaptive version of WalkSAT that dynamically adjusts the noise value based on the automatic detection of search stagnation. For example, AdaptNovelty$^+$, the adaptive version of Novelty$^+$, starts with $p = 0$ (i.e. the solver is completely greedy in searching for the next move). If the search enters a stagnation stage (i.e. it encounters a local minimum), the noise value is gradually increased to allow more non-greedy moves to be performed, and hence allow the search to eventually overcome its stagnation. As soon as the search escapes the local minimum, the noise value is reduced to make the search more greedy. Hoos [8] demonstrated experimentally that this adaptive noise mechanism is effective both with Novelty$^+$ and other WalkSAT variants.

In 2005, Anbulagan *et al.* [5] introduced R+AdaptNovelty$^+$, a two-phase SLS solver which improved the performance of AdaptNovelty$^+$ by utilising resolution in the initial phase to derive extra information from the input. In particular, the new solver applies a restricted resolution procedure to all clauses of length ≤ 3 from the input. This process adds resolvent clauses of length ≤ 3 to the problem, and also removes duplicate clauses, tautologies, and literals that appear twice in a single clause. It then runs AdaptNovelty$^+$ on the resulting problem.

3 Our Approach: gNovelty$^+$

The initial development of gNovelty$^+$ focussed on preparing for the 2007 SAT competition. This meant concentrating on the random problem category, where SLS solvers have traditionally outperformed complete solvers. Consequently we paid considerable attention to the previously best performing techniques from this category: R+AdaptNovelty$^+$, G^{2}WSAT and AdaptNovelty$^+$.

Algorithm 1. gNovelty$^+$(F)

```
1: for $try = 1$ to $maxTries$ do
2:     initialise the weight of each clause to 1;
3:     randomly generate an assignment A;
4:     for $step = 1$ to $maxSteps$ do
5:         if A satisfies F then
6:             return A as the solution;
7:         else
8:             if within a walking probability $wp$ then
9:                 randomly select a variable $x$ that appears in a false clause;
10:            else if there exist promising variables then
11:                greedily select a promising variable $x$, breaking ties by selecting the least recently
                   flipped promising variable;
12:            else
13:                select a variable $x$ according to the weighted AdaptNovelty heuristic;
14:                update the weights of false clauses;
15:                with probability $sp$ smooth the weights of all clauses;
16:            end if
17:            update A with the flipped value of $x$;
18:        end if
19:    end for
20: end for
21: return 'no solution found';
```

Firstly, we noted that although G^{2}WSAT came second in the random SAT category in the 2005 competition, it was the best solver for random 3-SAT instances. As neither R+AdaptNovelty$^+$ nor AdaptNovelty$^+$ were competitive with G^{2}WSAT on those instances, we conjectured that the superior performance of G^{2}WSAT was due to its greedy behaviour. This supposition is consistent with the results reported in [6] where G^{2}WSAT was compared with Novelty$^+$.

On the other hand, R+AdaptNovelty$^+$ outperformed G^{2}WSAT on the 5-SAT and 7-SAT instances in the 2005 competition. This could not be explained by the resolution preprocessor employed by R+AdaptNovelty$^+$ because, for such instances, it simply reorders the occurrence of literals in a clause. We also empirically found that G^{2}WSAT relies heavily on its Novelty^{++} component when solving these instances. Finally we observed that AdaptNovelty$^+$ is generally a more effective Novelty variant for use by G^{2}WSAT than Novelty^{++}.

On the basis of the preceding observations, we constructed gNovelty$^+$ by replacing the Novelty^{++} heuristic in G^{2}WSAT with the AdaptNovelty heuristic to enhance performance on the 5-SAT and 7-SAT instances. We then incorporated a clause weighting heuristic to further improve overall performance. Previously, clause weighting DLS techniques have not performed competitively in the SAT competitions because of their reliance on optimally tuned parameters (the rules of the competition do not allow the hand-tuning of parameters to particular problems). However, DLS generally performs better than WalkSAT when parameter tuning is allowed [9,10], indicating that clause weights can provide useful guidance. The problem for DLS approaches has been the lack of an effective adaptive mechanism such as the one developed for WalkSAT.[3] Our aim with gNovelty$^+$

[3] Note, RSAPS [9] is a DLS technique that uses a WalkSAT-type adapting mechanism, but RSAPS only adapts one of three possible parameters (see [10]) and remains uncompetitive with the best WalkSAT techniques in the SAT competition.

was to cross this boundary and implement clause weight guidance within an adaptive WalkSAT framework. The resulting gNovelty$^+$ solver is sketched out in Algorithm 1 and depicted diagrammatically in Figure 1.

At every search step, gNovelty$^+$ selects the most promising variable that is also the least recently flipped, based on a *weighted* objective function. The objective is to minimise the sum of weights of all false clauses and a promising variable is any variable that, if flipped, will reduce the overall weighted cost of the solution. If no such promising variable exists, the next variable is selected using a heuristic based on AdaptNovelty that again uses the *weighted* objective function. After an AdaptNovelty step, gNovelty$^+$ increases the weights of all current false clauses by one.[4]

In order to flexibly control the greediness of the search, we also incorporated a new linear version of the probabilistic weight smoothing from SAPS [9]. According to a *smoothing probability sp*, each time gNovelty$^+$ updates the clause weights, this heuristic will reduce the weight of all *weighted* clauses by one (a clause is weighted if its weight is greater than one). Finally, we added a probabilistic-walk heuristic (i.e. the *plus* heuristic from Novelty$^+$ [7]) to further improve the balance between diversification and greediness of the search.

3.1 The Question of Parameters

For SAT 2007, gNovelty$^+$ was specifically tuned for random k-SAT problems with $sp = 0.4$ for $k = 3$ and 1.0 otherwise. It should be noted, however, that an sp value of 1.0 *turns off* the weighting component of gNovelty$^+$ as each weight increase is immediately followed by a weight decrease.

This raises the question as to whether the superior performance of gNovelty$^+$ in the SAT competition was largely dependent on a fortuitous selection of fixed sp values (i.e. that exploited the narrow domain of potential competition problems), or whether the algorithm has a wider field of useful application. To answer this question, we devised an experimental study to test gNovelty$^+$ in comparison with four other state-of-the-art SLS SAT solvers and across a range of benchmark structured problems.

4 Experimental Study

As the performance of gNovelty$^+$ in the SAT random category is already a matter of public record,[5] we based our experimental study on a range of structured benchmark problems that have been used in previous SLS comparison studies.[6] Our problem test set comprises of two circuit synthesis formula problems (3bitadd_31 and 3bitadd_32), two all-interval series problems (ais10 and ais12), two blocksworld planning problems (bw_large.c and bw_large.d), two "flat" graph

[4] We decided to use the additive weight increase at each local minimum as it is cheaper to maintain than its multiplicative weighting counterpart [10].

[5] See http://www.cril.univ-artois.fr/SAT07/slides-contest07.pdf

[6] See http://www.satlib.org

colouring problems (flat200-med and flat200-har), four large DIMACS graph colouring problems (g125.17 to g250.29), two logistics planning problems (logistics.c and logistics.d), five 16-bit parity function learning problems (par16-1-c to par16-5-c), and five hard quasi-group problems (qg1-08 to qg7-13).

As gNovelty$^+$ crosses the boundary both between WalkSAT and clause weighting algorithms and between adaptive and manually-based parameter tuning, for comparison purposes we selected algorithms from each of the four possible categories, i.e. manual WalkSAT (G^{2}WSAT [6]), adaptive WalkSAT (AdaptNovelty$^+$ [8]), manual clause weighting (PAWS [10]) and adaptive clause weighting (RSAPS [9]). Using these algorithms, we manually tuned PAWS, G^{2}WSAT and gNovelty$^+$ to obtain optimal performance for each category of the problem set. These settings are shown in Table 1 (note, only one parameter setting per algorithm was allowed for each of the eight problem categories). Here we not only manipulated the gNovelty$^+$ *sp* parameter but on some categories we also manually tuned the noise parameter of its Novelty component. For G^{2}WSAT we used the optimal settings for the noise and *dp* parameters published in [6,11], and for PAWS we tuned the w_{inc} parameter (w_{inc} is analogous to gNovelty$^+$'s *sp* parameter except that gNovelty$^+$ reduces weight with probability *sp* whereas PAWS reduces weight after every w_{inc} number of increases [10]).

Table 1. Optimal parameter settings for each problem category

Method	Parameter	Problem Category							
		3bitadd	ais	bw_large	flat200	g	logistics	par16	qg
gNovelty$^+$	noise	adapted	adapted	0.08	adapted	0.10	adapted	0.05	0.02
	sp	0.00	0.00	1.00	0.00	1.00	0.00	0.10	0.00
G^{2}WSAT	noise	0.50	0.20	0.20	0.50	0.30	0.20	0.50	0.40
	dp	0.05	0.05	0.00	0.06	0.01	0.05	0.01	0.03
PAWS	w_{inc}	9	52	4	74	4	100	40	10

4.1 Results

Table 2 shows the results obtained after manually tuning gNovelty$^+$, G^{2}WSAT and PAWS in comparison to the default adaptive behaviour of RSAPS and AdaptNovelty$^+$. Here the results for the best performing algorithm on each problem are shown in bold, with all results reporting the mean and median of 100 runs of each algorithm on each instance. It is worth noting that G^{2}WSAT uses unit propagation to preprocess input problems. We expect that the performance of G^{2}WSAT without unit propagation would be degraded as our benchmark instances, especially the quasi-group ones, contain many unit clauses. All experiments were performed on cluster of 16 computers, each with a single AMD Opteron 252 2.6GHz processor with 2GB of RAM, and each run was timed out at 600 seconds.

A brief overview shows that gNovelty$^+$ has the best results for all 3bitadd, ais, bw_large and logistics problems, is about equal with G^{2}WSAT on the flat graph colouring problems and has some success on the hardest qg problems (although G^{2}WSAT also performs well in this class). Of the other algorithms, PAWS is

Table 2. Optimally tuned results shown in the form: $\frac{median}{mean}$

Instances	gNovelty$^+$		G^{2}WSAT		AdaptNovelty$^+$		RSAPS		PAWS	
	#flips	#secs	#flips	#secs	#flips	#secs	#flips	#secs	#flips	#secs
3bitadd_31	**17,948 / 19,746**	**0.06 / 0.07**	0% success		151,079 / 162,828	0.44 / 0.47	0% success		0% success	
3bitadd_32	**15,116 / 15,510**	**0.06 / 0.06**	0% success		131,019 / 134,891	0.45 / 0.47	0% success		0% success	
ais10	**8,116 / 11,596**	**0.01 / 0.01**	28,450 / 58,101	0.02 / 0.05	1,290,782 / 1,936,648	1.14 / 1.67	13,122 / 18,293	0.02 / 0.02	13,184 / 19,823	0.01 / 0.02
ais12	**48,480 / 63,938**	**0.06 / 0.08**	329,284 / 446,670	0.33 / 0.44	24,315,102 / 35,291,601	28.14 / 40.83	103,156 / 151,088	0.14 / 0.20	104,414 / 192,256	0.15 / 0.26
bw_large.c	**799,665 / 1,277,197**	**1.44 / 2.22**	1,807,797 / 2,658,924	1.84 / 2.68	6,350,267 / 9,354,872	6.55 / 9.70	3,606,150 / 5,360,847	10.72 / 15.92	990,959 / 1,603,890	1.46 / 2.27
bw_large.d	**937,056 / 1,131,405**	**2.65 / 3.10**	2,308,922 / 3,121,103	3.60 / 4.82	21,462,024 / 30,615,678	37.05 / 55.24	70% success		1,029,200 / 1,370,793	2.77 / 3.51
flat200-med	163,566 / 241,467	0.07 / 0.10	**140,542 / 185,529**	**0.05 / 0.07**	260,162 / 391,731	0.09 / 0.13	287,330 / 376,523	0.15 / 0.20	248,096 / 348,112	0.14 / 0.18
flat200-har	2,576,103 / 4,037,105	**1.04 / 1.64**	3,713,653 / 14,243,118	1.36 / 5.01	17,879,059 / 22,572,057	6.05 / 7.58	3,143,106 / 4,199,967	1.59 / 2.15	**2,402,889 / 3,343,704**	1.28 / 1.76
g125.17	687,290 / 1,066,447	3.06 / 4.79	**592,518 / 634,772**	**2.00 / 2.12**	981,362 / 1,264,481	4.11 / 5.41	2% success		492,028 / 693,579	2.31 / 3.42
g125.18	12,654 / 15,255	0.09 / 0.10	8,538 / 10,152	0.11 / 0.12	35,346 / 35,829	0.10 / 0.10	1,057,010 / 1,786,612	5.00 / 8.38	**10,683 / 12,624**	**0.08 / 0.08**
g250.15	2,585 / 2,668	0.11 / 0.11	2,387 / 2,417	0.41 / 0.43	3,310 / 4,033	0.11 / 0.12	**2,208 / 2,219**	**0.08 / 0.08**	2,239 / 2,247	0.09 / 0.09
g250.29	637,612 / 704,455	12.72 / 14.96	295,924 / 347,680	**5.99 / 6.43**	755,477 / 894,775	9.96 / 11.98	0% success		**263,322 / 319,511**	8.07 / 9.24
logistics.c	**6,332 / 6,873**	**0.01 / 0.01**	53,876 / 64,369	0.04 / 0.04	122,186 / 151,855	0.07 / 0.09	6,812 / 7,814	0.01 / 0.01	10,152 / 11,642	0.01 / 0.01
logistics.d	**26,880 / 32,211**	**0.04 / 0.04**	88,108 / 106,585	0.09 / 0.10	170,170 / 196,295	0.10 / 0.11	22,558 / 32,636	0.04 / 0.05	29,954 / 42,457	0.05 / 0.06
par16-1-c	6,943,449 / 9,621,258	2.92 / 4.06	98% success		17,185,866 / 32,062,102	5.87 / 10.98	73% success		**1,556,886 / 2,470,449**	**0.86 / 1.37**
par16-2-c	28,290,998 / 38,825,548	11.97 / 16.49	90% success		159,404,628 / 260,240,406	53.41 / 87.52	39% success		**3,674,518 / 4,804,860**	**2.02 / 2.67**
par16-3-c	18,135,772 / 27,626,027	7.75 / 11.79	70,947,285 / 105,838,330	27.62 / 41.26	66,941,711 / 102,782,522	23.07 / 35.49	42% success		**2,613,043 / 4,106,440**	**1.46 / 2.31**
par16-4-c	11,146,298 / 16,938,387	4.81 / 7.22	121,641,036 / 220,317,867	46.73 / 84.22	80,419,118 / 112,779,831	27.91 / 38.67	69% success		**1,034,957 / 2,182,801**	**0.57 / 1.21**
par16-5-c	11,829,928 / 17,544,960	5.02 / 7.44	61% success		90,819,874 / 126,328,695	31.23 / 43.84	41% success		**3,169,028 / 4,092,230**	**1.74 / 2.24**
qg1-08	647,290 / 920,411	15.64 / 22.48	**539,888 / 950,976**	**1.90 / 3.28**	99% success		59% success		80% success	
qg2-08	2,545,216 / 3,294,816	51.12 / 69.99	**3,948,297 / 5,859,212**	**13.94 / 20.68**	43% success		36% success		20% success	
qg5-11	99% success		74% success		1% success		**2,287,392 / 3,287,772**	**24.57 / 35.40**	22% success	
qg6-09	726,178 / 3,090,041	2.23 / 9.32	**20,455 / 95,756**	**0.04 / 0.15**	14% success		28,846 / 43,886	0.11 / 0.17	832,520 / 1,263,106	3.29 / 5.00
qg7-13	**98% success**		31% success		0% success		3% success		0% success	

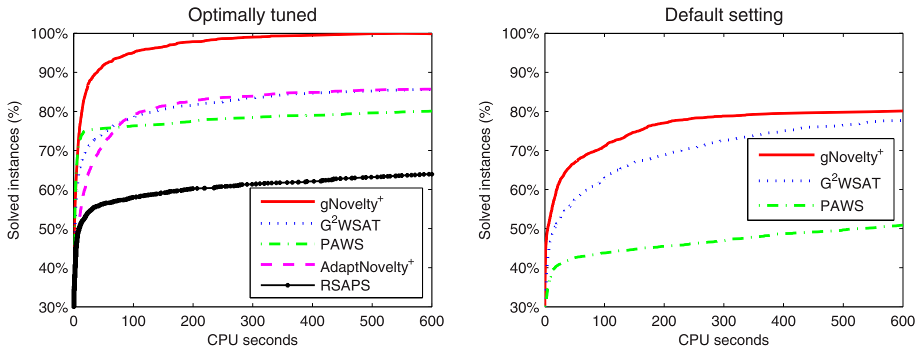


Fig. 2. Run-time distributions over the complete data set

the best for the parity problems, with G^{2}WSAT and PAWS coming about equal first on the large graph colouring problems, and RSAPS winning on one quasi-group and one large graph instance. On this basis gNovelty$^+$ emerges as the best algorithm both in terms of the number of problems (ten) and the number of problem classes (four) in which it dominates.

Table 3. Default parameter setting results shown in the form: $\frac{median}{mean}$

Instances	gNovelty$^+$		G^{2}WSAT		PAWS	
	#flips	#secs	#flips	#secs	#flips	#secs
3bitadd_31	**17,948** / **19,746**	**0.06** / **0.07**	0% success		0% success	
3bitadd_32	**15,116** / **15,510**	**0.06** / **0.06**	0% success		0% success	
ais10	**8,116** / **11,596**	**0.01** / **0.01**	79,932 / 112,419	0.07 / 0.09	68,462 / 94,817	0.08 / 0.11
ais12	**48,480** / **63,938**	**0.06** / **0.08**	1,073,489 / 1,661,896	1.10 / 1.72	917,232 / 1,424,442	1.41 / 2.17
bw_large.c	**831,984** / **1,131,058**	**1.49** / **2.04**	53% success		7,237,556 / 9,440,193	17.31 / 22.49
bw_large.d	**4,802,306** / **5,498,694**	**14.52** / **16.74**	0% success		39% success	
flat200-med	163,566 / 241,467	0.07 / 0.10	**111,942** / **168,466**	**0.04** / **0.07**	145,014 / 180,628	0.08 / 0.10
flat200-har	**2,576,103** / **4,037,105**	**1.04** / **1.64**	5,422,588 / 13,873,574	2.00 / 4.89	5,922,780 / 8,168,341	3.12 / 4.30
g125.17	**3,524,594** / **4,229,083**	**15.89** / **18.81**	99% success		6% success	
g125.18	185,058 / 184,953	0.97 / 0.95	**12,314** / **13,472**	**0.16** / **0.17**	18,774 / 24,911	0.15 / 0.19
g250.15	2,254 / 2,283	**0.08** / **0.08**	2,445 / 2,475	0.43 / 0.46	**2,211** / **2,236**	0.11 / 0.11
g250.29	**4,984,444** / **5,008,310**	**136.75** / **141.15**	47% success		0% success	
logistics.c	**6,332** / **6,873**	**0.01** / **0.01**	36,389 / 54,279	0.03 / 0.04	118,259 / 157,790	0.10 / 0.13
logistics.d	**26,880** / **32,211**	**0.04** / **0.04**	2,056,840 / 3,050,451	1.75 / 2.56	275,307 / 385,814	0.28 / 0.38
par16-1-c	20% success		**97% success**		23% success	
par16-2-c	10% success		**85% success**		1% success	
par16-3-c	5% success		**114,988,942** / **166,070,580**	**44.78** / **64.59**	5% success	
par16-4-c	9% success		**98% success**		16% success	
par16-5-c	5% success		**52% success**		3% success	
qg1-08	**852,874** / **1,133,742**	18.90 / 24.94	1,175,202 / 1,660,389	**4.45** / **6.25**	83% success	
qg2-08	**3,154,862** / **4,092,638**	68.67 / 91.04	5,954,183 / 9,058,721	**20.96** / **31.67**	23% success	
qg5-11	5,012,010 / 6,863,167	39.12 / 50.50	**1,237,121** / **1,979,199**	**7.10** / **10.19**	22% success	
qg6-09	342,864 / 2,281,251	1.16 / 7.32	**11,376** / **19,469**	**0.03** / **0.04**	968,044 / 1,229,162	4.04 / 5.10
qg7-13	**74% success**		33% success		0% success	

An even clearer picture emerges when we look at the overall proportion of runs that completed within 600 seconds. Here, gNovelty$^+$ achieves a 99.88% success rate compared with 85.63% for AdaptNovelty$^+$, 85.58% for G^{2}WSAT, 80.08% for PAWS and 63.92% for RSAPS. This result is reinforced in the run time distributions (RTDs) on the left-hand of Figure 2 where the gNovelty$^+$ curve dominates over the entire time range.

Overall, gNovelty$^+$ not only outperforms the other techniques in the greatest number of problem classes, it is within an order of magnitude of the best performing algorithms in all remaining cases. It is this robust average case performance (that gNovelty$^+$ also demonstrated in the SAT competition) that argues strongly for its usefulness as a general purpose solver.

However, if such robust behaviour depends critically on manually tuned parameter settings then the case for gNovelty$^+$ must weaken. To evaluate this we tested gNovelty$^+$ on the same problem set with a default *sp* value of 0.00 (meaning clause weights are increased in each local minimum but never decreased) and with the noise parameter adjusted using gNovelty$^+$'s adapt mechanism.[7] These

[7] Although gNovelty$^+$'s noise parameter was also adjusted in Table 1, performance was not greatly improved, with the main benefits coming from adjusting *sp*.

results and the results of the default parameter values for G^{2}WSAT ($dp = 0.05$ and noise $= 0.5$) and PAWS ($w_{inc} = 10$) are shown in Table 3.

In this second comparison, the results of RSAPS and AdaptNovelty$^+$ from Table 2 should also be considered (as both algorithms were run without the benefit of manual tuning). Tables 2 and 3 show that AdaptNovelty$^+$ now has the best overall success rate of 85.63% followed by the default valued gNovelty$^+$ at 80.13%, G^{2}WSAT at 77.67%, with RSAPS (63.92%) and PAWS (50.88%) coming last (this is also illustrated in the RTDs in Figure 2).

However, the situation changes when we consider individual problem categories, with gNovelty$^+$ dominating in five classes (3bitadd, ais, bw_large, flat and logistics), AdaptNovelty$^+$ in two (g and par16) and G^{2}WSAT on one (qg). Looking in more detail, we can see that the main negative impact of a fixed parameter on gNovelty$^+$ has come from its failure on the parity problems. Similarly, AdaptNovelty$^+$ fails only on the quasi-group problems. If we put these two data sets aside, then the default gNovelty$^+$ shows a clear advantage over AdaptNovelty$^+$, dominating on five of the remaining six problem classes.

5 Discussion and Conclusions

The experimental evidence of this paper and the SAT competition demonstrates that gNovelty$^+$ is a highly competitive algorithm for random SAT problems, and, with parameter tuning, that it can dominate several of the previously best performing SLS algorithms on a range of structured problems. If parameter tuning is ruled out (as it would be in most real-world problem scenarios), then gNovelty$^+$ still performs well, and only lost to its closest rival, AdaptNovelty$^+$, on one structured problem class.

Once again, as with PAWS and SAPS, the addition of a clause weighting heuristic to gNovelty$^+$ has required the addition of a sensitive weight decay parameter to get competitive results. Nevertheless, the situation with gNovelty$^+$'s parameter does differ from SAPS and PAWS in that highly competitive performance can be obtained from a relatively small set of parameter values (i.e. 0.0, 0.1, 0.4 and 1.0). In contrast, SAPS and PAWS require much finer distinctions in parameter values to get even acceptable results [10]. This smaller set of values means that the process of tuning *sp* is considerably simplified in comparison to other clause weight techniques. More importantly, gNovelty$^+$'s more robust behaviour indicates that it may be easier to devise an automatic adapting mechanism for *sp*. To date, procedures for automatically adapting weight decay parameters have not produced the fastest algorithms.[8] In future work, it therefore appears promising to try and develop a simple heuristic that will effectively adapt *sp* in the structured problem domain.

In conclusion, we have introduced the new SLS random satisfiable problem category winner of the SAT 2007 competition, gNovelty$^+$. We have extended

[8] Although machine learning techniques that are trained on test sets of existing instances and then applied to unseen instances have proved useful for setting SAPS and Novelty parameters [12].

the SAT 2007 results and shown that gNovelty$^+$ is also effective in solving structured SAT problems. In fact, gNovelty$^+$ has not only outperformed four of the strongest current SLS SAT solvers, it has also demonstrated significant robustness in solving a wide range of diverse problems. In achieving this performance, we have highlighted gNovelty$^+$'s partial dependence on the setting of its *sp* weight decay parameter. This leads us to recommend that future work should concentrate on the automatic adaptation of this parameter.

Acknowledgements. We thankfully acknowledge the financial support from NICTA and the Queensland government. NICTA is funded by the Australian Government's *Backing Australia's Ability* initiative, and in part through the Australian Research Council.

References

1. Davis, M., Logemann, G., Loveland, D.: A machine program for theorem proving. Communications of the ACM 5(7), 394–397 (1962)
2. Selman, B., Levesque, H., Mitchell, D.: A new method for solving hard satisfiability problems. Proceedings of AAAI 1992, 440–446 (1992)
3. McAllester, D.A., Selman, B., Kautz, H.A.: Evidence for invariants in local search. In: Proceedings of AAAI 1997, pp. 321–326 (1997)
4. Morris, P.: The Breakout method for escaping from local minima. In: Proceedings of AAAI 1993, pp. 40–45 (1993)
5. Anbulagan, P.D.N., Slaney, J., Sattar, A.: Old resolution meets modern SLS. In: Proceedings of AAAI 2005, pp. 354–359 (2005)
6. Li, C.M., Huang, W.Q.: Diversification and determinism in local search for satisfiability. In: Bacchus, F., Walsh, T. (eds.) SAT 2005. LNCS, vol. 3569, pp. 158–172. Springer, Heidelberg (2005)
7. Hoos, H.H.: On the run-time behaviour of stochastic local search algorithms for SAT. In: Proceedings of AAAI 1999, pp. 661–666 (1999)
8. Hoos, H.H.: An adaptive noise mechanism for WalkSAT. In: Proceedings of AAAI 2002, pp. 635–660 (2002)
9. Hutter, F., Tompkins, D.A., Hoos, H.H.: Scaling and probabilistic smoothing: Efficient dynamic local search for SAT. In: Van Hentenryck, P. (ed.) CP 2002. LNCS, vol. 2470, pp. 233–248. Springer, Heidelberg (2002)
10. Thornton, J.R.: Clause weighting local search for SAT. Journal of Automated Reasoning 35(1-3), 97–142 (2005)
11. Li, C.M., Wei, W., Zhang, H.: Combining adaptive noise and look-ahead in local search for SAT. In: Proceedings of LSCS 2006 Workshop, pp. 2–16 (2006)
12. Hutter, F., Hamadi, Y., Hoos, H.H., Leyton-Brown, K.: Performance prediction and automated tuning of randomized and parametric algorithms. In: Benhamou, F. (ed.) CP 2006. LNCS, vol. 4204, pp. 213–228. Springer, Heidelberg (2006)

Clone: Solving Weighted Max-SAT in a Reduced Search Space

Knot Pipatsrisawat and Adnan Darwiche

Computer Science Department
University of California, Los Angeles
{thammakn,darwiche}@cs.ucla.edu

Abstract. We introduce a new branch-and-bound Max-SAT solver, Clone, which employs a novel approach for computing lower bounds. This approach allows Clone to search in a reduced space. Moreover, Clone is equipped with novel techniques for learning from soft conflicts. Experimental results show that Clone performs competitively with the leading Max-SAT solver in the broadest category of this year's Max-SAT evaluation and outperforms last year's leading solvers.

1 Introduction and Background

The maximum satisfiability problem (Max-SAT) is one of the optimization counterparts of the Boolean satisfiability problem (SAT). In Max-SAT, given a Boolean formula in conjunctive normal form (CNF), we want to determine the maximum number of clauses that can be satisfied by any complete assignment.[1]

Two important variations of the Max-SAT problem are the weighted Max-SAT and the partial Max-SAT problems. The *weighted* Max-SAT problem is the Max-SAT problem, in which each clause is assigned a positive weight. The objective of this problem is to maximize the sum of weights of satisfied clauses by any assignment. The *partial* Max-SAT problem is the Max-SAT problem, in which some clauses cannot be left falsified by any solution.[2] The combination to both variations is called the *weighted partial* Max-SAT problem. For the rest of this paper, we use the term Max-SAT to refer to any variation of the Max-SAT problem, while noting that our solver is meant for the broadest category: *weighted partial* Max-SAT.

There are two main approaches used by contemporary exact Max-SAT solvers: the satisfiability-based approach and the branch-and-bound approach. The former converts each Max-SAT problem with different hypothesized maximum weights into multiple SAT problems and uses a SAT solver to solve these SAT problems to determine the actual solution. Examples of this type of solver are ChaffBS, ChaffLS [1] and SAT4J-MaxSAT [2]. The second approach, which

[1] A clause is a disjunction of literals and a literal is simply a variable or its negation.
[2] In practice, a madatory clause is represented by a clause with a sufficiently large weight.

seems to dominate in terms of performance based on recent Max-SAT evaluations [3,4], utilizes a depth-first branch-and-bound search in the space of possible assignments. An evaluation function which computes a bound is applied at each search node to determine any pruning opportunity.

The methods used to compute bounds vary among branch-and-bound solvers and often give rise to difference in performance. Toolbar utilizes local consistencies to aid bound computations [5,6]. Lazy, MaxSatz, and MiniMaxSat compute bounds using some variations of unit propagation and disjoint component detection [7,8,9,10]. In this paper, we introduce a method of bound computation based on formula compilation. Our approach can be thought of as a new paradigm that combines search and compilation together. A recently developed solver, Sr(w), by Ramírez and Geffner [11] employs a very similar approach for computing bounds. Clone and Sr(w) were developed independently and both participated in this year's Max-SAT evaluation [4]. The performance of both solvers in the evaluation is available at http://www.maxsat07.udl.es/ms07.pdf, showing that Clone dominated Sr(w) in most of the evaluation categories. We note, however, that the version of Clone described here is an improved version whose performance is significantly better than the one that participated in the Max-SAT evaluation 2007.

For the remaining of this paper, we describe the components of our Max-SAT solver. Clone consists of two main components. The first is the preprocessor which takes a Max-SAT problem as an input and produces a data structure necessary for computing bounds. The second is the branch-and-bound search engine that takes advantage of the data structure and many inference techniques to aid the search. None of the techniques used by Clone are specific to any particular type of problem, making Clone applicable to any variation of Max-SAT.

In the next section, we discuss the preprocessor and our approach for computing bounds. In Section 3, we describe the search component and the inference techniques used. Experimental results are presented in Section 4 and we close with some remarks in Section 5.

2 Bound Computation

In the literature, the Max-SAT problem is often viewed as the problem of minimizing the costs (weights) of falsified clauses of any assignment. We will follow this interpretation and use the term *cost* to refer to the sum of weights of clauses that cannot be satisfied. Moreover, we will use UB (upper bound) to denote the best cost of $\underline{\text{any complete assignment found so far}}$ and LB (lower bound) to denote the guaranteed cost of the $\underline{\text{current partial assignment}}$. Branch-and-bound search algorithm can prune all children of a node whenever $LB \geq UB$.

To compute lower bounds, we take advantage of a tractable language called deterministic decomposable negation normal form (d-DNNF) [12,13]. Many useful queries can be answered about sentences in d-DNNF in time linear in the size of these sentences. One of these queries is (weighted) minimum cardinality, which is similar to Max-SAT, except that weights are associated with variables instead of clauses. Our approach is indeed based on reducing Max-SAT on the given CNF to minimum cardinality on a d-DNNF compilation of the CNF. If this compilation is successful, the Max-SAT problem is solved immediately since minimum cardinality can be solved in time linear in the d-DNNF size. Unfortunately, however, the compilation process is often difficult. Our solution to this problem is then to compile a relaxation of the original CNF, which is generated carefully to make the compilation process feasible. The price we pay for this relaxation is that solving minimum cardinality on the relaxed d-DNNF compilation will give lower bounds instead of exact solutions. Our approach is then to use these lower bounds for pruning in our brand-and-bound search.

We show how a Max-SAT problem can be reduced to a minimum cardinality problem in Section 2.1. We will then discuss problem relaxation in Section 2.2, followed by compilation in Section 2.3.

2.1 Reducing Max-SAT to Minimum Cardinality

Given a CNF formula and a cost for each literal of the formula, the weighted minimum cardinality problem asks for a satisfying assignment that costs the least. The cost of an assignment is the sum of costs of all literals that it sets to true. To reduce a Max-SAT problem into a minimum cardinality problem, we introduce a distinct selector variable to each clause of the Max-SAT problem and assign the clause's cost to the positive literal of the selector variable [14]. All other literals have zero cost. For example, the clause $C = (a \vee b \vee c)$ becomes $C' = (s \vee a \vee b \vee c)$ after the selector variable s is added. If C originally had cost w, then w is assigned to s and any assignment that set $s = \textbf{true}$ will incur this cost. After this conversion, the formula will be trivially satisfiable, because every clause contains a distinct selector variable. Nevertheless, finding a satisfying assignment with the lowest cost is not easy. The minimum cardinality problem is NP-hard for CNF formulas. However, it can be solved efficiently once we have the formula in d-DNNF. Any solution to this problem can be converted back to a solution for the original Max-SAT problem by ignoring assignments of the selector variables.

At this point, we are almost ready to compile the CNF formula. The only remaining issue is the time complexity of the compilation, which is, in the worst case, exponential in the treewidth of the constraint graph of the CNF formula. In most cases, straight compilation will be impractical. As a result, we need to relax the formula to lower its treewidth.

2.2 Problem Relaxation by Variable Splitting

The approach we use to relax the problem is called *variable splitting*, which was inspired by the work of Choi *et al* in [15]. In general, splitting a variable v involves introducing new variables for all but one occurrence of v in the original CNF formula.[3] For example, splitting a in the CNF $(a \vee b) \wedge (\neg a \vee c) \wedge (a \vee d) \wedge (b \vee$

[3] This type of splitting is called full splitting. While other degrees of splitting are possible, we focus our attention only to this method.

$\neg c) \wedge (c \vee \neg d)$ results in the formula $(a \vee b) \wedge (\neg a_1 \vee c) \wedge (a_2 \vee d) \wedge (b \vee \neg c) \wedge (c \vee \neg d)$. In this case, a is called the *split variable*. The new variables (a_1 and a_2 in this case) are called the *clones* of the split variable. Figure 1 illustrates the constraint graph of the above CNF before and after the split.

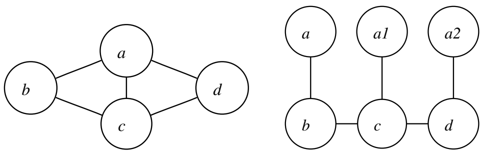


Fig. 1. (left) The constraint graph of $(a \vee b) \wedge (\neg a \vee c) \wedge (a \vee d) \wedge (b \vee \neg c) \wedge (c \vee \neg d)$. (right) The constraint graph after splitting a. The treewidth is reduced from 2 to 1.

After splitting, the resulting problem becomes a relaxation of the original problem because any assignment in the original problem has an assignment in the split problem with the same cost.[4] Therefore, the lowest cost of any split formula is a lower bound of the lowest cost of the original formula. The strategy we use for selecting split variables is the same as the one described in [15].

2.3 CNF to d-DNNF Compilation

Once the treewidth of the problem is low enough, the problem can be practically compiled. The process of compiling a CNF formula into a d-DNNF formula is performed by a program called C2D [13,16]. C2D takes a CNF formula as input and produces an equivalent sentence in d-DNNF. The output formula is fed to the search engine and will be used for later bound computations.

2.4 Computing Bounds from d-DNNF

Every d-DNNF formula can be represented as a rooted DAG. Each node in the DAG is either a Boolean constant, a literal, or a logical operator (conjunction or disjunction). The root of the DAG corresponds to the formula. For example, consider the DAG of a d-DNNF formula $(a \wedge ((b \wedge c) \vee (\neg b \wedge \neg c))) \vee (\neg a \wedge ((b \vee \neg c) \vee (\neg b \vee c)))$ in Figure 2. Let the cost of every positive literal be 1 and the cost of every negative literal be 2. The minimum cardinality of the formula is simply the value of the root [14], which is defined recursively as:

1. The value of a literal node is the value of the literal
2. The value of an AND node is the sum of the values of all its children
3. The value of an OR node is the minimum of the values of its children

If the formula is a relaxed formula, then the computed minimum cardinality becomes a lower bound of the minimum cardinality of the formula before relaxation (hence a lower bound of the optimal cost of the original Max-SAT problem).

[4] This can be obtained by setting the value of every clone according to its split variable.

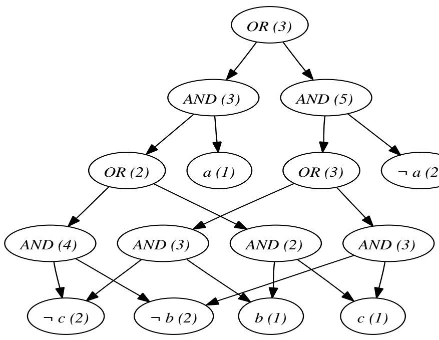


Fig. 2. The DAG of the d-DNNF formula $(a \wedge ((b \wedge c) \vee (\neg b \wedge \neg c))) \vee (\neg a \wedge ((b \vee \neg c) \vee (\neg b \vee c)))$. Each node in this graph is also labeled with the value used to compute the minimum cardinality of the root.

The d-DNNF formula can also be efficiently conditioned on any partial or complete assignment of its variables. Conditioning only affects the values of the nodes whose literals are set to false. The values of such nodes become ∞, which may in turn affect the values of their parents or ancestors. In practice, bound computation can be done incrementally. Only the nodes whose values have changed since the last evaluation need to be inspected. The resulting bound computed from the conditioned formula will be a lower bound of the optimal cost of any assignment that extends the conditioning assignment.

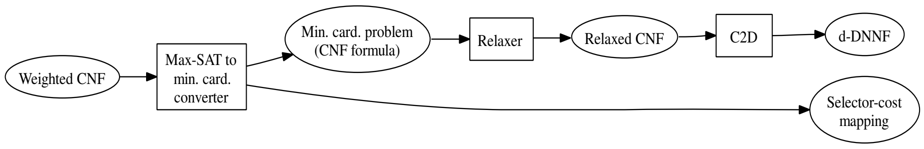

Weighted CNF
Max-SAT to min. card. converter
Min. card. problem (CNF formula)
Relaxer
Relaxed CNF
C2D
d-DNNF
Selector-cost mapping

Fig. 3. A system diagram of Clone's preprocessor

Figure 3 summarizes the relationship between different parts of the preprocessor. The system's input is a Max-SAT problem (weighted CNF formula). The system produces a d-DNNF formula and information about the costs of selector variables. These outputs are passed on to the branch-and-bound search engine.

3 Search and Inference

The next component of Clone is the branch-and-bound search engine. The engine uses bounds computed from the d-DNNF formula for pruning. The search algorithm only branches on variables in the original Max-SAT problem (as opposed to clones or selectors). Every time a split variable is assigned a value, the algorithm ensures that all its clones are set to the same value. Otherwise, a solution in this search space may be meaningless in the original problem.

Apart from standard inference techniques such as unit propagation, non-chronological backtracking [17,18,19], and conflict clause learning [18,20], the search engine of Clone employs many novel techniques to improve the efficiency of the search. We describe them next. Some of these techniques require access to the clauses in the original Max-SAT problem. Hence, although we compile the CNF formula into a d-DNNF, the original formula is also given to the search engine.

3.1 Reducing the Size of the Search Space

Given our method for computing bounds, it is possible to reduce the size of the search space that the algorithm needs to explore. Since the relaxed problem differs from the original problem only on split variables and their clones, as soon as every split variable (and its clones) is set to a value, the two problems become identical under the current assignment. Therefore, the bound computed from the d-DNNF formula at this point must be the exact Max-SAT optimal cost of the original problem under the current assignment.

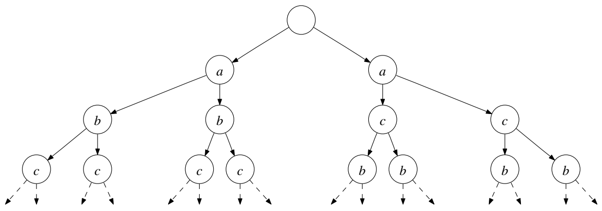


Fig. 4. An example search space with each node labeled with the last branch variable

For example, consider the search space of a problem with 3 split variables–a, b, and c–in Figure 4. The bound computed at each node is a lower bound of the optimal cost of any complete assignment that extends the assignment at the node. However, once all three split variables are instantiated, the bound becomes exact. Therefore, there is no need to visit any node below depth 3 in this search tree. This realization suggests a natural way of reducing the size of the search space; we only need to search in the space of assignments of split variables.

3.2 Dealing with Soft Conflicts

Whenever $LB \geq UB$, the solver can prune the current branch of the search. This situation is called a *soft conflict*, because no hard clause is violated. A *hard clause* is one whose cost is greater than or equal to UB. A *soft clause* is a clause that is not hard. A hard clause cannot be falsified by any optimal solution.[5] Tra-

[5] Violation of a hard clause results in a hard conflict, which is handled by normal conflict analysis as in [20,21].

ditionally, branch-and-bound search algorithm backtracks chronologically (flip the most recent unflipped decision) when a soft conflict is found. The standard SAT techniques for dealing with conflicts do not apply directly to this type of conflicts, because of the lack of a violated hard clause. However, in some cases, we could efficiently construct a violated hard clause.

Upon each soft conflict, clearly $LB \geq UB$. Moreover, some soft clauses may already be falsified by the current assignment. However, it is not necessary that the sum S of the costs of the violated soft clauses be greater than or equal to UB. This is simply because the method used for computing lower bounds may be able to implicitly identify certain clauses that can never be satisfied at the same time, even when their truth values are still in question. In any case, whenever $S \geq UB$, a violated hard clause can be artificially constructed. In particular, let $\{C_1, C_2, ..., C_k\}$ be a set of soft clauses whose sum of costs exceeds (or equal) UB. Then, $C = \vee_{i=1}^{k} C_i$ is a hard clause. The clause C simply makes explicit the fact that $C_1, ..., C_k$ cannot all be false at the same time.

To perform conflict analysis on a soft conflict, we need to find a minimal set of violated soft clauses whose sum of costs is greater than UB. The disjunction of the literals of these clauses is a violated hard clause, which can be used by the standard conflict analysis algorithm to generate a conflict clause, which is a hard clause, and to compute a level to non-chronologically backtrack to.

A similar learning scheme for soft conflicts was proposed in the solver PMS [22]. This learning scheme utilizes Max-SAT resolution rule [23] and, according to [22], has not been coupled with non-chronological backtracking.

3.3 Avoiding Work Repetition

Both chronological and non-chronological backtracking are employed by Clone. The former is used when the solver encounters a soft conflict for which no violated hard clause can be easily constructed. In this case, no new clause is learned by the solver. Non-chronological backtracking is used for the other pruning opportunities and a new conflict clause is learned every time.

Combining these two types of backtracking naively may lead the solver to repeat a lot of work in some situations. For example, consider the search tree in Figure 5 (a). In this search tree, the variable a was flipped to $\neg a$ by a chronological backtrack, which indicates that the whole subtree under $a =$ **true** had been exhausted. After a few more decisions (b and c), the solver may run into a hard conflict and derive a conflict clause ($\neg c$). As a result, non-chronological backtracking will erase every assignment and set $c =$ **false** at the very top level (level 0). The search tree after backtracking is shown in Figure 5 (b). The assignment $a =$ **false** had been erased in the process and there is no way to recover the fact that the branch $a =$ **true** had been fully explored. MiniMaxSAT [9] is the only other solver employing both chronological and non-chronological backtracking that we are aware of and it may suffer from this inefficiency.[6]

To solve this problem, every time Clone non-chronologically backtracks past any chronologically flipped assignment, it records a *blocking clause*, which is a

[6] Based on the paper and communication with one of the authors.

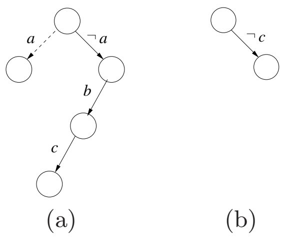


Fig. 5. (a) A search tree in which a has been flipped by chronological backtracking. (b) The search tree after non-chronological backtracking and learning conflict clause $(\neg c)$.

hard clause that captures that information. More formally, let ℓ be a current literal assignment that was flipped by a chronological backtrack at level k. If the solver non-chronologically backtracks past ℓ, it will learn $C = (\ell \vee \neg d_1 \vee \neg d_2 \vee ... \vee \neg d_{k-1})$, where d_i is the decision assignment at level i. In the above example, $(\neg a)$ would be learned since it was the very first decision in the search tree.

4 Experimental Results

In this section, we present experimental results that compare Clone against other leading weighted Max-SAT solvers that are publicly available. The first solver is Toolbar (version 3.1), which was the best solver in the weighted category of the Max-SAT evaluation 2006 and also participated in the Max-SAT evaluation 2007. The second solver is Lazy, which was outperformed by only Toolbar in the same category in 2006. The last solver is MiniMaxSat, which outperformed others in the weighted partial Max-SAT category of the 2007 evaluation.

The version of Clone used here is an improved version of the one in the Max-SAT evaluation 2007. In addition to the techniques for handling soft con-

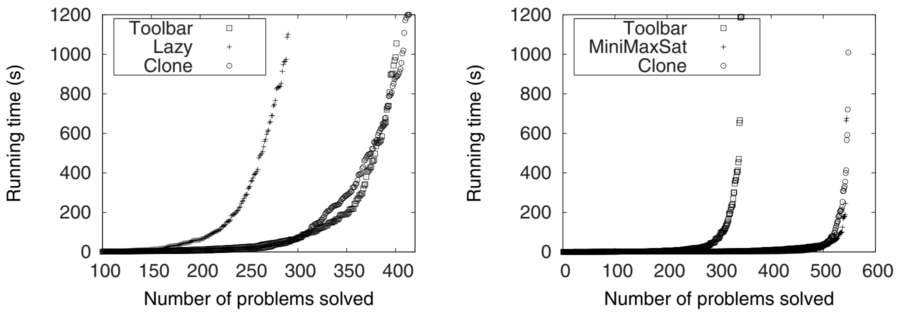


Fig. 6. Running time profile of solvers on Max-SAT problems from Max-SAT evaluation 2006 (left) and 2007 (weighted partial Max-SAT category) (right)

flicts described earlier, this version of Clone uses MaxWalkSat [24] to find initial seeds. Moreover, it utilizes an improved variable ordering heuristic which always starts with the Jeroslow-Wang heuristic [25] and dynamically switches to the VSIDS [26] heuristic if hard conflicts are encountered frequently enough. To make memory usage sustainable, Clone periodically deletes inactive learned clauses in the manner that maintains the completeness of the solver.

In all experiments, every Max-SAT problem is relaxed until its treewidth is less than or equal to 8, which we found to be empirically optimal. All experiments were run on two Intel Pentium 4 machines, each with 3.79GHz and 4GB of RAM. The time-out is 1,200 seconds per problem. We considered all non-random weighted Max-SAT problems from the 2006 evaluation and all non-random weighted partial Max-SAT problems from the 2007 evaluation.[7]

Table 1. Number of solved problems by suite (left) from the Max-SAT evaluation 2006 and (right) from the Max-SAT evaluation 2007

Suite	Total	Toolbar	Lazy	Clone
AUCTION (PATHS)	30	26	20	**28**
AUCTION (REGIONS)	30	30	30	30
AUCTION (SCHED.)	30	**30**	**30**	28
MAXONE	45	**44**	24	33
MAXCLIQUE	62	**33**	29	19
MAXCSP	180	120	53	**169**
QCP	25	10	6	**24**
RAMSEY	48	35	28	**36**
SPOT5	42	9	5	**12**
WMAXCUT	67	59	**60**	28
WQUEENS	7	5	5	5
Total	566	401	290	412

Suite	Total	Toolbar	MiniMaxSat	Clone
AUCTION (PATHS)	88	88	88	88
AUCTION (REGIONS)	84	84	84	84
AUCTION (SCHED.)	84	82	**84**	78
QCP	25	10	20	**24**
PLANNING	71	52	**71**	**71**
PSEUDO MIPLIB	16	**7**	5	5
PSEUDO FACTOR	186	8	**186**	**186**
SPOT5	42	9	7	**12**
Total	596	340	545	548

Figure 6 shows performance profiles of solvers on different set of benchmarks. In each plot, problems from each Max-SAT evaluation are used to compare Clone against solvers that participated in the evaluation. The left plot of Figure 6 shows that the performance of Clone is comparable to that of Toolbar and better than that of Lazy on problems from the 2006 evaluation. The right plot shows that Clone is comparable to MiniMaxSat, which does significantly better than Toolbar on problems from the 2007 evaluation. Table 1 reports the number of solved problems by suite. According to the left table, out of 566 problem, Clone solved 11 problems more than Toolbar and 122 problems more than Lazy. The right table shows that Clone solved roughly the same number of problems as MiniMaxSat on most suites and solved 3 more problems overall.

5 Conclusions

We presented in this paper a new weighed Max-SAT solver, Clone, which employs (1) a new approach for computing lower bounds, (2) a corresponding technique for reducing the search space, and (3) some novel techniques for handling soft conflicts. The performance of Clone on non-random problems used in the recent

[7] In 2006, unweighted Max-SAT was the only other evaluation category. In 2007, there were 3 other categories. Here, we consider the broadest, most general category.

Max-SAT evaluations is competitive with those of other leading solvers, which are based on more mature approaches for Max-SAT. Our results demonstrate the potential of the new techniques adopted by Clone, some of which are quite different from the techniques adopted by classical Max-SAT solvers.

Acknowledgments

We would like to thank the anonymous reviewers for their constructive comments and for suggesting clarifications on the similarities between Clone and other solvers.

References

1. Fu, Z., Malik, S.: On solving the partial max-sat problem. In: Biere, A., Gomes, C.P. (eds.) SAT 2006. LNCS, vol. 4121, Springer, Heidelberg (2006)
2. Le Berre, D.: Sat4j project homepage, http://www.sat4j.org/
3. Argelich, J., Li, C.M., Manya, F., Planes, J.: First evaluation of max-sat solvers, http://www.iiia.csic.es/~maxsat06/
4. Argelich, J., Li, C.M., Manya, F., Planes, J.: Second evaluation of max-sat solvers, http://www.maxsat07.udl.es/
5. Larrosa, J., Heras, F., de Givry, S.: A Logical Approach to Efficient Max-SAT solving. ArXiv Computer Science e-prints (November 2006)
6. de Givry, S., Heras, F., Zytnicki, M., Larrosa, J.: Existential arc consistency: Getting closer to full arc consistency in weighted csps. In: IJCAI, pp. 84–89 (2005)
7. Alsinet, T., Manya, F., Planes, J.: A max-sat solver with lazy data structures. In: Proc. of 9th Ibero-American Conf. on Artificial Intelligence (2004)
8. Li, C.M., Manya, F., Planes, J.: New inference rules for max-sat. JAIR (2007)
9. Viaga, F.H., Larrosa, J., Oliveras, A.: Minimaxsat: a new weighted max-sat solver. In: Proceedings of SAT 2007 (2007)
10. Li, C.M., Manya, F., Planes, J.: Exploiting unit propagation to compute lower bounds in branch anb bound max-sat solvers. In: van Beek, P. (ed.) CP 2005. LNCS, vol. 3709, Springer, Heidelberg (2005)
11. Ramírez, M., Geffner, H.: Structural relaxations by variable renaming and their compilation for solving mincostsat. In: Proceedings of CP 2007. LNCS, vol. 4741, Springer, Heidelberg (to appear, 2007)
12. Darwiche, A., Marquis, P.: A knowlege compilation map. Journal of Artificial Intelligence Research 17, 229–264 (2002)
13. Darwiche, A.: New advances in compiling CNF to decomposable negational normal form. In: Proceedings of European Conference on Artificial Intelligence (2004)
14. Darwiche, A., Marquis, P.: Compiling propositional weighted bases. Artificial Intelligence 157(1-2), 81–113 (2004)
15. Choi, A., Chavira, M., Darwiche, A.: Node splitting: A scheme for generating upper bounds in bayesian networks. In: Proceedings of UAI'07 (2007)
16. Darwiche, A.: The c2d compiler. Available at http://reasoning.cs.ucla.edu/c2d/
17. Stallman, R., Sussman, G.: Forward reasoning and dependency-directed backtracking in a system for computer-aided circuit analysis. Artificial Intel. 9 (1977)

18. Marques-Silva, J., Sakallah, K.: Grasp: A search algorithm for propositional satisfiability. IEEE Trans. Computers 5, 506–521 (1999)
19. Bayardo, R.J.J., Schrag, R.C.: Using CSP look-back techniques to solve real-world SAT instances. In: Proceedings of AAAI'97, pp. 203–208 (1997)
20. Zhang, L., Madigan, C.F., Moskewicz, M.W., Malik, S.: Efficient conflict driven learning in boolean satisfiability solver. In: ICCAD, pp. 279–285 (2001)
21. Ryan, L.: Efficient Algorithms for Clause-Learning SAT Solvers. Master's thesis, Simon Fraser University (2004)
22. Argelich, J., Manya, F.: Partial max-sat solvers with clause learning. In: Proceedings of SAT'07, pp. 28–40 (2007)
23. Larrosa, J., Heras, F.: Resolution in max-sat and its relation to local consistency in weighted csps. In: Proc. of the Intnl. Jnt. Conf. on Artfcl. Intel., pp. 193–198 (2005)
24. Selman, B., Kautz, H.: Walksat home page, http://www.cs.rochester.edu/u/
25. Wang, J.: A branching heuristic for testing propositional satisfiability. In: Systems, Man and Cybernetics, 1995. Intelligent Systems for the 21st Century, IEEE International Conference, pp. 4236–4238 (1995)
26. Moskewicz, M., Madigan, C., Zhao, Y., Zhang, L., Malik, S.: Chaff: Engineering an efficient sat solver. In: 39th Design Automation Conference (DAC) (2001)

A CNF Formula Hierarchy over the Hypercube

Stefan Porschen

Institut für Informatik, Universität zu Köln,
Pohligstr. 1, D-50969 Köln, Germany
porschen@informatik.uni-koeln.de

Abstract. We study the first level of a conjunctive normal form (CNF) formula hierarchy with respect to the propositional satisfiability problem (SAT). This hierarchy is defined over a base formula that we call a *hypercube (formula)*. Such a hypercube simply consists of all 2^n possible n-clauses over a given set of n Boolean variables. The first level of the hierarchy are *1-hyperjoins*, meaning that arbitrary hypercubes are joined together via taking from each arbitrary many clauses for joining, i.e., set-union, such that each chosen clause occurs in at most one new clause of the 1-hyperjoin. We prove that arbitrary 1-hyperjoins can efficiently be recognized and solved w.r.t. SAT. To that end we introduce a simple closure concept on the set of the propositional variables of a formula.

Keywords: satisfiability, hypercube formula, hypergraph, transversal, variable closure.

1 Introduction

The Boolean conjunctive normal form (CNF) satisfiability problem, called SAT for short, gets as input a CNF formula and has to decide whether this formula admits a satisfying truth assignment. As is well known the remarkable result by S. Cook from 1971 established SAT as the first and genuine complete problem for the complexity class NP [4]. Thus SAT resides at the heart of the NP $\neq$ P conjecture of complexity theory.

A vital branch of satisfiability theory is concerned with finding subclasses of CNF for which SAT can be shown to be polynomial-time solvable. Of course, it is the specific structure shared by the members of such a class enabling one to design efficient algorithms for SAT. There are several polynomial-time SAT-testable classes known, as quadratic formulas, (q-)Horn formulas, matching formulas etc. [1,2,3,6].

Of special interest in this context are formula hierarchies that provide a certain dichotomy meaning that there is a definite level below of which corresponding formulas are polynomial time solvable and formulas on and above that level appear to be intractable. In the present paper taking an abstract point of view, we study the first level of a, as far as we know, new CNF formula hierarchy that is based on so-called *hypercube (for short* hc*) (formula)s.* A hypercube simply consists of all 2^n possible n-clauses over a set of n distinct variables. Formulas of the rth level of this hierachy, for $r \geq 1$, are called *r-hyperjoins*, where for $r = 1$ we simply speak of hyperjoins. Informally, a hyperjoin is composed out of finitely many hypercube formulas via taking an arbitrary set of clauses of each hypercube and joining them to new clauses such that each chosen clause occurs in exactly one new clause. An r-hyperjoin is built like a hyperjoin with the modification that each of the clauses chosen for joining from the underlying hypercubes can occur in at most r new clauses of the r-hyperjoin.

The paper is structured as follows: In the next Section 2 we provide some preliminaries and notational conventions. The next two sections are devoted to explain some tools that turn out to be useful for our hyperjoin investigations, namely the f-transversal concept in Section 3 and the variable closure in Section 4. In the main Section 5 we prove that hyperjoins can be recognized fast, and also are efficiently SAT-solvable. Finally, we collect some open problems in Section 6.

2 Notation and Preliminaries

Let CNF denote the set of conjunctive normal form formulas over propositional variables $x \in \{0, 1\}$. A *positive (negative)* literal is a (negated) variable. The *negation (complement)* of a literal l is $\bar{l}$. Each formula $C \in$ CNF is considered as a clause set, and each clause $c \in C$ is represented as a literal set. It is assumed that no clause contains a complemented pair like $\{x, \bar{x}\}$. For a formula C, clause c, by $V(C), V(c)$ we denote the set of variables contained in C, c, respectively (neglecting negations). The length of C is denoted as $\|C\|$. For $n \in \mathbb{N}$, we use $[n] := \{1, \ldots, n\}$ as a convenient abbreviation.

Recall that the satisfiability problem (SAT) asks whether an input formula $C \in$ CNF has a *model*, which, by defininition, is a truth assignment $t : V(C) \to \{0, 1\}$ assigning at least one literal in each clause of C to 1.

It turns out to be convenient to identify truth assignments with clauses, i.e. literal sets, in the following simple way: Let $x^0 := \bar{x}$, $x^1 := x$. Then we can identify a truth assignment $t : V \to \{0, 1\}$ with the literal set $\{x^{t(x)} : x \in V\}$. A crucial notion throughout is that of a *hypercube (formula)* W_V over variable set V. W_V is the collection of all clauses obtained in the way described previously via running through all total truth assignments $V \to \{0, 1\}$. Observe that the clauses of W_V are in 1:1-correspondence to the vertices of a hypercube of dimension $|V|$ explaining the name. In other words W_V is the set of all clauses c such that $V(c) = V$. For example, given $V = \{x, y\}$ we have $W_V = \{xy, \bar{x}y, x\bar{y}, \bar{x}\bar{y}\}$ as the hypercube over V writing clauses as literal strings.

For a clause c we denote by c^γ the clause in which all its literals are complemented. Similarly, let $t^\gamma = 1 - t : V \to \{0, 1\} \in W_V$ be the truth assignment obtained from t by complementing all assignments, and finally, set $C^\gamma := \{c^\gamma : c \in C\}$.

For $C \in$ CNF and $U \subseteq V(C)$, let $C(U) := \{c \in C : V(c) \cap U \neq \varnothing\}$ denote the *subformula (i.e. subset)* of C involving the variables in U. If $U = \{x\}$ we shall simply write $C(x)$. In contrast to $C(U)$, let $C_U := \{c|_U : c \in C\}$ be the *substructure* of C consisting of the restrictions $c|_U$ of the clauses $c \in C$ to the literals over U (which in general is no subformula of C). We call the formula C_U the *U-retraction* of C. For example, consider the formula

$$C = \{\bar{x}_1 x_2 x_3, \bar{x}_3 x_4 \bar{x}_5, x_1 x_3 \bar{x}_4 x_5, x_1 x_2 \bar{x}_4\}$$

then for $U = \{x_1, x_4\}$ the U-retraction of C is $C_U = \{\bar{x}_1, x_4, x_1\bar{x}_4\}$.

An *autark* set of variables can be removed from a formula without affecting its satisfiability status. More precisely, for $C \in$ CNF, we call $U \subseteq V(C)$ an *autark set*, if there exists a (partial) truth assignment $\alpha : U \to \{0, 1\}$ satisfying $C(U)$; in which case α is called an *autarky* [7]. Removing U, hence $C(U)$, from C therefore preserves the satisfiability status of the resulting formula. Obviously for a formula $C \in$ CNF, a set $U \subseteq V(C)$ is autark if and only if C_U is satisfiable.

3 Fibre-Transversals in Clause Sets

The notion of *fibre(f)-transversals* in CNF formulas (not to be mixed up with a hitting set) regards clauses as sets over a base hypergraph, and turns out to be useful later on. In essence we view clause sets via projection on the hypergraph that comes up when eliminating negations of literals in clauses. More precisely, a *(variable-) base hypergraph* $\mathcal{H} = (V, B)$ is a hypergraph whose vertices $x \in V$ are regarded as Boolean variables such that for each $x \in V$ there is a (hyper)edge $b \in B$ with $x \in b$. So B can be considered as a positive clause set. As above, for any $b \in B$, W_b is the hypercube formula over the set of variables in b. $K_{\mathcal{H}} := \bigcup_{b \in B} W_b$ denotes the set of all possible clauses over $\mathcal{H}$, and is called the *total clause set over $\mathcal{H}$*. For example, given $V = \{x_1, x_2, x_3\}$, and $B = \{b_1 := x_1 x_2, b_2 := x_1 x_3\}$, we have $K_{\mathcal{H}} = W_{b_1} \cup W_{b_2}$ where $W_{b_1} = \{x_1 x_2,\ \bar{x}_1 x_2,\ x_1 \bar{x}_2,\ \bar{x}_1 \bar{x}_2\}$, $W_{b_2} = \{x_1 x_3,\ \bar{x}_1 x_3,\ x_1 \bar{x}_3,\ \bar{x}_1 \bar{x}_3\}$ are the hc formulas over b_1, b_2.

Regarding each b as a *point* in the space B, we obtain the following mapping $\pi : K_{\mathcal{H}} \ni c \mapsto V(c) \in B$ recalling that $V(c)$ is the set of variables in clause c. We call $\pi^{-1}(b) = W_b$ the *fibre of $K_{\mathcal{H}}$* over b. Obviously the fibres are mutually disjoint and π is surjective, thus the map π can be regarded as a *projection*.

A *formula over $\mathcal{H}$ (or $\mathcal{H}$-formula)* is any subset $C \subseteq K_{\mathcal{H}}$ such that $C \cap W_b \neq \emptyset$ for each $b \in B$, implying that the restriction $\pi_C := \pi|_C$ yields a projection $\pi_C : C \to B$. Any formula $C \in$ CNF can be described in this way: Each $C \in$ CNF can be regarded as a formula over the base hypergraph $\mathcal{H}(C) := (V(C), B(C))$ with $B(C) := \{V(c) : c \in C\}$. Let $C \subset K_{\mathcal{H}}$ such that $W_b - C_b \neq \emptyset$, for all $b \in B$, then we define the *$\mathcal{H}$-based complement formula* of C as $\bar{C} := \bigcup_{b \in B}(W_b - C^b)$ where $C^b := \pi_C^{-1}(b)$. By construction $\bar{C}$ has the same base hypergraph as C. For example, for $\mathcal{H} = (V, B)$ with $V = \{x_1, x_2, x_3\}$, and $B = \{x_1 x_2, x_1 x_3\}$, let $C = \{x_1 \bar{x}_2, x_1 x_2, x_1 \bar{x}_3, \bar{x}_1 \bar{x}_3\}$ then $K_{\mathcal{H}} = C \cup \bar{C}$ where $\bar{C} = \{\bar{x}_1 x_2, \bar{x}_1 \bar{x}_2, x_1 x_3, \bar{x}_1 x_3\}$. A *fibre-transversal (f-transversal)* of $K_{\mathcal{H}}$ is a formula $F \subset K_{\mathcal{H}}$ such that $|F \cap W_b| = 1$, for each $b \in B$. Hence F is a formula exactly containing one clause of each fibre $\pi^{-1}(b)$ of $K_{\mathcal{H}}$, let that clause be refered to as $F(b)$. An important type of f-transversals F are those containing each variable of V as a *pure literal*, that is, occuring in F with a fixed polarity only. Such f-transversals are called *compatible* and have the specific property $\bigcup_{b \in B} F(b) \in W_V$. As a simple example for a compatible f-transversal consider the base hypergraph $\mathcal{H}$ with variable set $V := \{x_1, x_2, x_3\}$ and the base points $b_1 = x_1x_2$, $b_2 = x_1x_3$, $b_3 = x_2x_3$. Then e.g. the clauses $c_1 := \bar{x}_1x_2$, $c_2 := \bar{x}_1\bar{x}_3$, and $c_3 := x_2\bar{x}_3$ form a compatible f-transversal of the corresponding $K_{\mathcal{H}}$, because $c_1 \cup c_2 \cup c_3 = \bar{x}_1x_2\bar{x}_3 \in W_V$.

For convenience, let us derive a result that is used later on. It characterizes satisfiability of a formula C in terms of the existence of a compatible f-transversal in its based complement formula $\bar{C}$:

Theorem 1. *For $\mathcal{H} = (V, B)$, let $C \subset K_{\mathcal{H}}$ be a $\mathcal{H}$-formula such that $\bar{C} \subset K_{\mathcal{H}}$ also is a $\mathcal{H}$-formula (hence $B = B(C) = B(\bar{C})$). Then C is satisfiable if and only if $\bar{C}$ admits a compatible f-transversal.*

Proof. Suppose C is satisfiable and let t be one of its models. Then for each base point $b \in B = B(C) = B(\bar{C})$, the restriction $t|_b$ of t to b satisfies all clauses of the fibre $\pi^{-1}(b)$ of $K_{\mathcal{H}}$ except for the clause $t^\gamma|_b$ obtained from $t|_b$ via complementing all literals. Hence $F_t(b) := t^\gamma|_b$ is a member of $\bar{C}$ and therefore $F_t := \bigcup_{b \in B} t^\gamma|_b$ is a compatible f-transversal of $\bar{C}$.

Conversely, let F be a compatible f-transversal of $\bar{C}$. Then, by definition of compatibility $t := \bigcup_{b \in B} F(b)$ is a truth assignment. And we claim that via complementing all assignments we obtain a model t^γ of C. Indeed, suppose the contrary, meaning that there is a base point b and a clause c over b belonging to C that is not satisfied by t^γ. Then this clause must have the form $c = t|_b \in C$, but this means a contradiction to $F(b) = t|_b \in \bar{C}$ as F was assumed to be an f-transversal of $\bar{C}$, so the proof is finished. □

4 Variable Closures in Formulas

In this section we briefly study a certain closure concept over the variables in a CNF formula that will turn out to be useful in the next section.

Definition 1. *For $C \in$ CNF, and a variable subset $U \subseteq V(C)$, the* variable closure (also called variable hull) *$H_C(U) \subseteq V(C)$ of U in C is defined as $H_C(U) := V(C) - V(C - C(U))$, and thus is the set of all variables in C that do occur in $C(U)$ but not outside $C(U)$. For $U = \{x\}$, we simply write $H_C(x)$.*

For example, take $C = \{x_1x_2, \bar{x}_1x_3, x_2x_4, x_4\bar{x}_5\}$ and $U = \{x_1, x_5\}$ then $H_C(U) = \{x_1, x_3, x_5\}$. A nice property of $H_C(U)$, which is not hard to verify, is that if U is autark then also $H_C(U)$ is autark. Given $C \in$ CNF and $U = \{x_{i_1}, \ldots, x_{i_k}\} \subseteq V(C)$, for simplicity let $H(U) = H(x_{i_1}, \ldots, x_{i_k}) := H_C(U)$ (if C is fixed).

Lemma 1. *Let $x, y \in V(C)$ then $H(x) = H(y)$ if and only if for each $c \in C$ holds $x \in V(c) \Leftrightarrow y \in V(c)$. In that case holds $H(x) = H(x, y) = H(y)$.*

Proof. The if direction of the equivalence is obvious. For the reverse direction assume $H(x) = H(y)$ from which follows $y \in H(x)$ and $x \in H(y)$ implying that for each $c \in C$ holds $x \in V(c) \Leftrightarrow y \in V(c)$. The latter is equivalent to $C(x) = C(\{x, y\}) = C(y)$ implying $H(x) = H(x, y) = H(y)$. □

We call a variable hull in $C \in \mathrm{CNF}$ *free* if it does not contain a *subhull*, i.e. a non-empty subset which is a hull itself. A hull U is called *i-hull* if there is a set $Q \subseteq V(C)$ minimal for inclusion with $|Q| = i$ such that $H(Q) = U$.

Lemma 2. *(1) A hull $U \subseteq V(C)$ in C is free iff $H(x) = U$ for each $x \in U$. (2) Each free hull is a 1-hull, of which there exist at most $|V(C)|$.*

Proof. For each $x \in U$ we have $H(x) \subseteq H(U)$, so if U is free we have $H(x) = H(U) = U$. Conversely, assume there is a non-empty hull $Q \subsetneq U$ in C and $H(x) = U$ for each $x \in U$. Because $x \in Q$ implies $H(x) \subseteq H(Q) = Q \neq U$ we have $H(x) \neq U$ yielding a contradiction, and so part (1) is proved. Part (2) is true because we have $H(x) = U$ for a free hull U and each $x \in U$. $\square$

So we arrive at:

Theorem 2. *For $C \in \mathrm{CNF}$, we can check in polynomial time which hulls in C are free or whether there is none. Moreover, a free hull $U \subseteq V(C)$ in C can be checked for autarky in linear time $O(\|C\|)$.*

Proof. Since each free hull is a 1-hull we focus on 1-hulls and first compute $H(x)$ for all $x \in V(C)$, which can be done in time $O(n \cdot \|C\|)$. Since $|H(x)| \in O(n)$ for each $x \in V(C)$ where $n := |V(C)|$, in $O(n^2)$ we can check whether a given hull contains one of the $n-1$ other hulls, yielding an overall time of $O(n \cdot \|C\| + n^3)$. For the second assertion it suffices to show that if U is a free hull then the retraction C_U is a subset of the hypercube formula W_U. Indeed, in that case we have $C_U \notin \mathrm{SAT}$ if and only if $C_U = W_U$. So, testing a free hull for autarky simply means checking, whether $|C_U| = 2^{|U|}$ holds. Computing C_U from C can be achieved by inspecting C with appropriate data structures in linear time. It remains to verify $C_U \subseteq W_U$ for which we claim that $U \subseteq V(c)$, for each $c \in C(U)$. Suppose this does not hold, then there is $c \in C(U)$ and $x \in U$ not contained in $V(c)$. Hence $c \notin C(x)$ but $c \in C(U)$, therefore $C(x) \neq C(U)$. On the other hand, as U is free we must have $H(y) = U$, for each $y \in U$, thus $C(x) = C(U)$ yielding a contradiction. $\square$

5 Hyperjoin Formulas

Next we intend to characterize the class of (1-)hyperjoin formulas which are shown to behave efficiently w.r.t. SAT. Hyperjoins as defined next are in a certain way composed of a finite number of hc formulas which are defined in Section 2. Observe that hc formulas are largest formulas possessing *exactly one* free variable closure. Hyperjoin formulas may serve as the first level of a whole hierarchy of r-hyperjoins. This section also provides tools for efficiently recognizing 1-hyperjoins. Recall that $[k] := \{1, \ldots, k\}$.

Definition 2. *For $k \in \mathbb{N}$, $k \geq 2$, arbitrarily fixed, let $V_i, i \in [k]$, be sets of propositional variables with $V_i \cap V_j = \emptyset$, for each distinct $i, j \in [k]$, and $V := \bigcup_{i \in [k]} V_i$. Let $W_i := W_{V_i}$ denote the hc formula over V_i. A* (1-)hyperjoin (formula) $\mathfrak{H}$ *over $\{W_i : i \in [k]\}$ is the* CNF *formula defined as follows: Each clause $c_i \in W_i$ either* *equals a clause of $\mathfrak{H}$ or it is a proper subset of* exactly one *clause of $\mathfrak{H}$, for all $i \in [k]$ such that there is no clause in $\mathfrak{H}$ containing two clauses of the same W_i. In other words, to construct a $\mathfrak{H}$ out of $\{W_i : i \in [k]\}$ proceed as follows: Arbitrarily choose m_i clauses from W_i, where $0 \leq m_i \leq |W_i|$, for all $i \in [k]$, and by arbitrary unions compose new clauses, such that each chosen clause occurs in exactly one new clause, which are called the* joined *clauses in $\mathfrak{H}$, and such that each joined clause contains at most one member of W_i, for $i \in [k]$. Finally, adding all non-chosen clauses of the W_i's without modification yields the hyperjoin.*

For a hyperjoin $\mathfrak{H}$ over $\{W_i : i \in [k]\}$, we define for each W_i the joined part *$J_{\mathfrak{H}}(W_i)$ of W_i in $\mathfrak{H}$ by $J_{\mathfrak{H}}(W_i) := \{c \in W_i : c \notin \mathfrak{H}\}$.*

*An r-*hyperjoin *is defined analogously, except that each clause of a hc formula chosen for joining occurs in k distinct joined clauses for some $k \in \{1, \ldots, r\}$.*

As example for a hyperjoin, consider hc formulas W_1, for $V_1 = \{x, y, z\}$, and W_2, for $V_2 = \{s, t, u, v, w\}$, let $D_1 = \{x\bar{y}\bar{z}, xy\bar{z}\}$, $D_2 = \{s\bar{t}\bar{u}v\bar{w}, \bar{s}\bar{t}\bar{u}\bar{v}\bar{w}\}$ and as hyperjoin over W_1, W_2 let $(W_1 - D_1) \cup (W_2 - D_2) \cup \{x\bar{y}\bar{z}s\bar{t}\bar{u}v\bar{w}, xy\bar{z}\bar{s}\bar{t}\bar{u}\bar{v}\bar{w}\}$ with joined parts $J_{\mathfrak{H}}(W_i) = D_i$, $i = 1, 2$.

Clearly, a hyperjoin $\mathfrak{H}$ is unsatisfiable if there is $i \in [k]$ such that $J_{\mathfrak{H}}(W_i) = \varnothing$. From the general point of view, the relevant questions are:

(1) Given $C \in$ CNF, can we efficiently (and constructively) decide whether C is a hyperjoin, and in that case, can we efficiently determine its *construction rule*, that is, can we determine the corresponding hc formulas $W_i, i \in [k]$, and its joined parts in C?
(2) If we know that a hyperjoin $\mathfrak{H}$ is constructed over W_i with $J_{\mathfrak{H}}(W_i)$, $i \in [k]$, can we efficiently decide whether $\mathfrak{H}$ is satisfiable, and can we find a model in the positive case?

Regarding (1), a hyperjoin $\mathfrak{H}$ behaves somewhat as determined w.r.t. free variable hulls in the following sense:

Lemma 3. *For a hyperjoin $\mathfrak{H}$ over W_i, $i \in [k]$, let $x \in V := \bigcup_{i \in [k]} V_i$ be arbitrary, then exactly one of the following holds:*

(1) $H(x)$ is a free variable hull, and moreover has the property that either
 (a) there is $i \in [k]$ such that $x \in V_i$ and $H(x) = H(V_i) = V_i$, then the retraction $\mathfrak{H}_{H(x)} = W_i$, and $H(x)$ admits no autark assignment in $\mathfrak{H}$, or
 (b) there is $s \in \mathbb{N}$, $I \subseteq [k]$ such that $x \in V_I := \bigcup_{i \in I} V_i$, $|V_i| = s$, for each $i \in I$, and $H(x) = H(V_I) = V_I$, then the retraction $\mathfrak{H}_{H(x)}$ is a proper subset of W_{V_I}, and $H(x)$ admits an autark assignment in $\mathfrak{H}$.
 Moreover, each free hull U in $\mathfrak{H}$ is of type (a) or (b).
(2) $H(x) = V_I$ is no free hull, where $I \subseteq [k]$, and $x \in V_I$, then there are $y_1, \ldots, y_r \in V_I$, with $r \leq |I|$, such that $H(y_i)$ is a free hull in $\mathfrak{H}_i$ of type (1)(a) or (b), where $\mathfrak{H}_i := \mathfrak{H}_{i-1} - \mathfrak{H}_{H(y_{i-1})}$ are hyperjoins, for each $2 \leq i \leq r$, with $\mathfrak{H}_1 := \mathfrak{H}$.

Proof. Clearly, for each $x \in V$, $H(x)$ either is a free hull in $\mathfrak{H}$ or not. In the second case, $H(x)$ must contain a free hull U, which clearly is disjoint to other free hulls. Hence, assuming that (1) holds, it is not hard to see that removing the retraction $\mathfrak{H}_U$ from $\mathfrak{H}$ yields a smaller hyperjoin that, inductively, can be treated analogously implying that (2) holds. So it remains to verify (1). Suppose U is a free hull in $\mathfrak{H}$ then due to Lemma 2, we have $H(x) = U$, for each $x \in U$. Any clause of $\mathfrak{H}$ containing a variable $x \in V_i$, by construction, already contains the whole set V_i (disregarding negations). Therefore, for each hull U there exists a subset $I \subseteq [k]$ such that $U = V_I := \bigcup_{i \in I} V_i$. However, since U is free, we have the case $|I| = 1$ implying that U is of type (1)(a), of Lemma 3, and the retraction $\mathfrak{H}_{V_i} = W_i$ obviously is unsatisfiable. Or we have $|I| \geq 2$. Since U is free there cannot exist a clause in $\mathfrak{H}$ that does not contain all variables in V_I. So, each clause $c \in \mathfrak{H}$ such that $V_I \subseteq V(c)$ contains exactly one member of W_i, for all $i \in I$. It is not hard to see that this can happen if and only if $|V_i| = s$, for appropriate $s \in \mathbb{N}$. Moreover, then the retraction $\mathfrak{H}_{V_I}$ is a proper subset of W_{V_I}, and therefore admits a model, cf. the proof of Theorem 2 yielding (b). □

Now, we can answer the first question stated above:

Theorem 3. *Given $C \in$ CNF we can check in polynomial time whether C is a hyperjoin, and in positive case reveal its construction rule, except for satisfiable substructures corresponding to free hulls in C of type (1)(b) of Lemma 3.*

Proof. Take any $x \in V$ and compute $H(x)$ in linear time, then decide whether $H(x)$ is free which is possible in polynomial time, due to Theorem 2. In the positive case check whether (1)(a) or (b) of Lemma 3 holds. If neither is true, reject the input formula and stop. Otherwise, $H(x)$ is not a free hull in C, then repeat the last step for each variable in $H(x)$ distinct from x until either a rejection occured or a free hull is found. Then remove the corresponding retraction from C, and continue with the remaining formula as described above until a rejection is found or we have a decomposition of free hull retractions of C each of type (1)(a), or (b), due to Lemma 3.

Finally, computing the joined parts between the determined retractions, can be performed in polynomial time via comparing the retractions with C. □

As an example for the satisfiable substructures of type (1)(b) that can occur in a hyperjoin $\mathfrak{H}$, let $V_1 = \{x, y\}$, $V_2 = \{u, z\}$ be two sets among others composing $\mathfrak{H}$. Then joining all clauses of W_1 to those of W_2 in any 1:1-correspondence for building the joined part over V_1, V_2 yields a proper subset of $W_{V_1 \cup V_2}$ and therefore $V_1 \cup V_2$ is a free hull yielding a satisfiable subformula in $\mathfrak{H}$. No further clause over V_1, V_2 occurs in $\mathfrak{H}$ as $|V_1| = |V_2|$.

Addressing question (2) posed above, first observe that two clauses c_1, c_2 with the same base element $V(c_1) = V(c_2)$ either belong to the same hc formula W_1, or each is composed out of two or more clauses of the same set of hc formulas. Taking this into account we can prove the next unsatisfiability result for hyperjoins over two distinct hc formulas. It will later on serve as a kind of induction base.

Lemma 4. *A hyperjoin $\mathfrak{H}$ over W_1, W_2 is unsatisfiable if and only if $|J_{\mathfrak{H}}(W_1)| = |J_{\mathfrak{H}}(W_2)| \in \{0, 1\}$.*

Proof. Observe that $r := |J_{\mathfrak{H}}(W_1)| = |J_{\mathfrak{H}}(W_2)|$, since otherwise a clause of a hc formula was part of more than one clause of $\mathfrak{H}$. The base hypergraph $\mathcal{H}(\mathfrak{H})$, has at most three edges, namely $B := B(\mathfrak{H}) := \{V_1, V_2, V := V_1 \cup V_2\}$. $\mathfrak{H}$ is unsatisfiable iff there is no compatible f-transversal of the complement formula $\bar{\mathfrak{H}} = \bigcup_{b \in B}(W_b - \mathfrak{H}^b)$ due to Theorem 1. First, we show that there is no compatible f-transversal of $\bar{\mathfrak{H}}$ if $r \in \{0, 1\}$. In case $r = 0$, the assertion certainly is true. If $r = 1$, let $c = c_1 \cup c_2 \in \mathfrak{H}$ where $c_i \in J_{\mathfrak{H}}(W_i)$, $i = 1, 2$. Hence, $\pi_{\bar{\mathfrak{H}}}^{-1}(V_i) = \{c_i\}$, $i = 1, 2$, and each f-transversal of $\bar{\mathfrak{H}}$ must contain c_1, c_2, as well as any element clause $c' \in \pi_{\bar{\mathfrak{H}}}^{-1}(V) - \{c\}$. But the only clause of W_V that can yield a compatible f-transversal together with c_1, c_2 obviously is $c \in \mathfrak{H}$ itself. Hence there is no compatible f-transversal of $\bar{\mathfrak{H}}$ at all.

For the reverse direction, assume $r \geq 2$, and let $c = c_1 \cup c_2, d = d_1 \cup d_2 \in \mathfrak{H}$ with $c_i, d_i \in W_i, i = 1, 2$. Now we claim that $F := \{c_1, d_2, c_1 \cup d_2\}$ is a compatible f-transversal of $\bar{\mathfrak{H}}$. Indeed, compatibility is obvious, because the union of the elements in F yields $c_1 \cup d_2 \in W_V$. It remains to verify that F indeed is an f-transversal of $\bar{\mathfrak{H}}$, obviously it is an f-transversal of the total clause set over B. Obviously, $c_1, d_2 \in \bar{\mathfrak{H}}$, because these are members of joined parts. Moreover, because each element of $J_{\mathfrak{H}}(W_i), i = 1, 2$, is used only once to yield a joined clause in $\mathfrak{H}$ there can be no other joined clauses containing c_1 or d_2. Thus $c_1 \cup d_2 \in \bar{\mathfrak{H}}$ and therefore $c_1^\gamma \cup d_2^\gamma$ is a model of $\mathfrak{H}$. □

For example let $V_1 = \{x, y\}$, $V_2 = \{u, v\}$ and let the joined part of $\mathfrak{H}$ be $c_1 = x\bar{y}$ of W_1 and $c_2 = \bar{u}v$ of W_2. Then each f-transversal F of $\bar{\mathfrak{H}}$ clearly contains clauses c_1, c_2 and any clause of $\bar{W}_{V_1 \cup V_2}$ different from $c_1 \cup c_2$, hence F can never be compatible.

A formula C is *connected* if its base hypergraph $\mathcal{H}(C)$ is connected.

Proposition 1. *Let $\mathfrak{H}$ over W_i, $i \in [k]$, $k \geq 2$, be a connected hyperjoin. Then $\mathfrak{H}$ is satisfiable if there is a pair $(i, j) \in [k]^2$, $i \neq j$, such that for the retraction $\mathfrak{H}(i, j) := \mathfrak{H}_{V_i \cup V_j}$ holds $|J_{\mathfrak{H}(i,j)}(W_i)| = |J_{\mathfrak{H}(i,j)}(W_j)| \geq 2$.*

Proof. If $k = 2$ we are done because then $\mathfrak{H}$ is satisfiable iff $|J_{\mathfrak{H}}(W_1)| = |J_{\mathfrak{H}}(W_2)| \geq 2$ due to Lemma 4. Let $k \geq 3$. Clearly, since $\mathfrak{H}$ is connected there must exist a permutation π of $\{1, 2, 3, \ldots, k\}$ such that the hyperjoin $\mathfrak{H}(r) := \mathfrak{H}(1, \ldots, r)$, i.e., the retraction of $\mathfrak{H}$ constructed over the r first W_i w.r.t. π, is connected, for each fixed $2 \leq r \leq k$. In other words, it is possible to join the W_i's step by step until reaching $\mathfrak{H}$ such that each intermediate hyperjoin is connected.

Now let V_i, V_j, $i \neq j$, be such that $\mathfrak{H}(i, j)$ fulfills $|J_{\mathfrak{H}(i,j)}(W_i)| = |J_{\mathfrak{H}(i,j)}(W_j)| \geq 2$, and therefore is connected. Clearly, w.l.o.g. we can assume that $(i, j) = (1, 2)$ and moreover that $\{1, \ldots, k\}$ is a permutation as described above. Again, for $r = 2$ we are done by Lemma 4 meaning $\mathfrak{H}(2) \in \mathrm{SAT}$. Clearly, because $\mathfrak{H}(3)$ by construction is connected, at least one member of W_3 is joined to a clause of $\mathfrak{H}(2)$, hence $J_{\mathfrak{H}(3)}(W_3) \neq \varnothing$. Let t be a model of $\mathfrak{H}(2)$, and let $c \in J_{\mathfrak{H}(3)}(W_3)$ be arbitrarily chosen then we claim that $t \cup c^\gamma$ is a model of $\mathfrak{H}(3)$. From which the desired implication of the theorem follows inductively. To verify the claim, first observe that t specifically already satisfies all clauses containing the members of $J_{\mathfrak{H}(3)}(W_3)$ because they only are enlarged. Hence taking $c \in$

$J_{\mathfrak{H}(3)}(W_3)$ arbitrarily then c^γ satisfies all fragments of W_3 except for c but the corresponding joined clause in $\mathfrak{H}(3)$ is already satisfied, so the claim follows. □

Clearly, if a hyperjoin is disconnected we can consider each component independently as decribed previously. From the last proof we obtain immediately:

Corollary 1. *If there is a satisfiable retraction in a connected component of a hyperjoin formula, then the whole component is satisfiable.* □

To a hyperjoin $\mathfrak{H}$ over W_i, $i \in [k]$, construct an edge-weighted graph $G(\mathfrak{H})$ as follows: For each W_i build a vertex x_i and two distinct vertices x_i, x_j are joined by an edge iff for the retraction $\mathfrak{H}(i,j) := \mathfrak{H}_{V_i \cup V_j}$ holds $|J_{\mathfrak{H}(i,j)}(W_i)| = |J_{\mathfrak{H}(i,j)}(W_j)| > 0$, meaning that there exist $w \geq 1$ joined clauses in $\mathfrak{H}$ each containing a (distinct) member of W_i and of W_j, correspondingly. As weight assign to the edge the number w. Clearly, a hyperjoin $\mathfrak{H}$ is a connected formula iff $G(\mathfrak{H})$ is connected.

Proposition 2. *For $k \geq 2$, a connected hyperjoin $\mathfrak{H}$ over hc formulas $W_i := W_{V_i}$, $i \in [k]$, is satisfiable iff the sum of edge weights in $G(\mathfrak{H})$ is at least k.*

Proof. We proceed by induction on $k \geq 2$. The induction base is established due to Lemma 4. Now let the assertion hold, for each connected hyperjoin composed of at most k hc formulas, for a fixed $k \geq 2$, and assume that $\mathfrak{H}$ is connected and is composed over $k+1$ hc formulas; set $V := \bigcup_{i \in [k+1]} V_i$.

$\underline{\Rightarrow}$: So let $\mathfrak{H}$ be satisfiable, and suppose that the sum of edge weights in $G := G(\mathfrak{H})$ is at most k. Because of connectedness this implies that each edge weight is exactly one, so G has exactly k edges, and therefore is a tree because of connectedness. W.l.o.g. let W_{k+1} be a hc formula corresponding to a leaf in G having father vertex W_k then $\mathfrak{H}' := \mathfrak{H} - \mathfrak{H}_{V_{k+1}}$ is a hyperjoin over k hc formulas such that $G(\mathfrak{H}')$ is a tree having total edge weight $k-1$, and therefore, by induction hypothesis, $\mathfrak{H}'$ cannot be satisfiable. On the other hand, let t be a model of $\mathfrak{H}$, and let c correspond to the unique edge of weight 1 in G connecting W_{k+1} and W_k. Then clearly, $c = c_k \cup c_{k+1}$ where $c_k \in W_k$, $c_{k+1} \in W_{k+1}$ are fixed members, otherwise G would contain a cycle. Since W_k having at least two clauses is a leaf in G, at least one clause of W_{k+1} appears as a clause in $\mathfrak{H}$. Therefore, in order to satisfy these clauses, t has to contain the part c_{k+1}^γ. Hence, for satisfying c, t clearly has to satisfy c_k. It follows that the restriction of t to $V - V_{k+1}$ yields a model of $\mathfrak{H}'$ and therefore a contradiction.

$\underline{\Leftarrow}$: If $G(\mathfrak{H})$ which by assumption is connected has an edge of weight at least 2, then $\mathfrak{H}$ is satisfiable due to Prop. 1. In the remaining case $G(\mathfrak{H})$ has at least $k+1$ edges and each edge has weight exactly one. Then it contains a cycle having vertices $W_{i_0}, \ldots, W_{i_{r-1}}$, for appropriate $r \leq k+1$, corresponding to the retraction $\mathfrak{H}_{V(r)}$, where $V(r) := \bigcup_{j=0}^{r-1} V_{i_j}$. For all $j, l \in \{0, \ldots, r-1\}$, $j \neq l$, we then have $|J_{\mathfrak{H}_{V(r)}(i_j,i_l)}(W_{i_j})| = |J_{\mathfrak{H}_{V(r)}(i_j,i_l)}(W_{i_l})| = 1$ if $l = j+1 (\mod r)$ and 0 otherwise; let the corresponding joined clauses be $c_{i_0,i_1}, \ldots, c_{i_{r-1},i_0}$ meaning that $c_{i_j,i_{j+1}}$ contains a member $c_{i_j} \in W_{i_j}$ and a member $d_{i_{j+1}} \in W_{i_{j+1}} (\mod r)$, for $0 \leq j \leq r-1$. Now, it is not hard to see that a model for $\mathfrak{H}_{V(r)}$ is provided by $\bigcup_{j=0}^{r-1} d_{i_j}^\gamma (\mod r)$. Thus due to Cor. 1 $\mathfrak{H}$ is satisfiable completing the proof. □

The last result immediately yields a polynomial time algorithm for deciding SAT for hyperjoins and also to find a model, provided that the input formula is a hyperjoin and its construction rule is known. So we obtain the main result:

Theorem 4. *In polynomial time we can decide whether $C \in \text{CNF}$ is a hyperjoin; in that case, we can find a model for C or decide that none exists in linear time.*

Proof. Suppose we have found in polynomial time, due to Theorem 3, that C is a hyperjoin. Then we know its construction rule up to free hulls of type (1)(b), Lemma 3. We can build the graph $G(C)$ and compute its components in linear time. Clearly each component containing a detected free hull of type (1)(b), is satisfiable due to Cor. 1, and a model for each such component is given by c^γ, where c is an arbitrary clause in the based-complement of the corresponding hull due to Theorem 1 that can be found appropriately scanning the hull in linear time. For each remaining component proceed due to Prop. 2 for deciding its satisfiability, and in the positive case, a model can be provided in linear time depth-first traversing the graph and taking c^γ for determined fragments of hc formulas as executed in the proof of Prop. 1. □

6 Concluding Remarks and Open Problems

In the present paper we introduced and studied the hyperjoin formula class as the first level of the r-hyperjoin hierarchy over the hypercube. We have shown that this CNF formula class is polynomial time solvable. We leave it as open problems, resp., as future work tasks, to characterize the next level classes, namely the r-hyperjoin classes, $r \geq 2$, w.r.t. their SAT complexity. Further, can r-hyperjoin formulas be shown to be fixed-parameter tractable w.r.t. parameter r [5].

References

1. Aspvall, B., Plass, M.R., Tarjan, R.E.: A linear-time algorithm for testing the truth of certain quantified Boolean formulas. IPL 8, 121–123 (1979)
2. Boros, E., Crama, Y., Hammer, P.L.: Polynomial time inference of all valid implications for Horn and related formulae. AMAI 1, 21–32 (1990)
3. Boros, E., Hammer, P.L., Sun, X.: Recognition of q-Horn formulae in linear time. Discrete Appl. Math. 55, 1–13 (1994)
4. Cook, S.: The Complexity of Theorem Proving Procedures. In: Proceedings of the 3rd ACM STOC, pp. 151–158. ACM, New York (1971)
5. Downey, R.G., Fellows, M.R.: Parameterized Complexity. Springer, Heidelberg (1999)
6. Franco, J., Gelder, A.v.: A perspective on certain polynomial-time solvable classes of satisfiability. Discrete Appl. Math. 125, 177–214 (2003)
7. Monien, B., Speckenmeyer, E.: Solving satisfiability in less than 2^n steps. Discrete Appl. Math. 10, 287–295 (1985)

Planning as Satisfiability with Relaxed ∃-Step Plans

Martin Wehrle[1,⋆] and Jussi Rintanen[2,⋆⋆]

[1] Albert-Ludwigs-Universität, Institut für Informatik, Freiburg, Germany
[2] National ICT Australia Ltd and Australian National University, Canberra, Australia

Abstract. Planning as satisfiability is a powerful approach to solving domain independent planning problems. In this paper, we consider a relaxed semantics for plans with parallel operator application based on ∃-step semantics. Operators can be applied in parallel if there is at least one ordering in which they can be sequentially executed. Under certain conditions, we allow them to be executed simultaneously in a state s even if not all of them are applicable in s. In this case, we guarantee that they are enabled by other operators that are applied at the same time point. We formalize the semantics of parallel plans in this setting, and propose an effective translation for STRIPS problems into the propositional logic. We finally show that this relaxed semantics yields an approach to classical planning that is sometimes much more efficient than the existing SAT-based planners.

1 Introduction

Planning as satisfiability is a leading approach to solving domain independent planning problems. An important factor in its efficiency is the notion of parallel plans. In the definition of Kautz and Selman [1], operators can be applied in parallel as long as they are mutually non-interfering. This guarantees that any total ordering on them is a valid execution and leads in all cases to the same state. Therefore, non-interfering operators need not be considered in all possible orderings, and respective propositional encodings get more compact as many intermediate states do not have to be represented explicitly.

Based on an idea of Dimopoulos et al. [2], Rintanen et al. [3] present a relaxed semantics which they call ∃-step semantics. In contrast to the standard ∀-step semantics, operators are allowed to be executed in parallel if they are sequentially executable in at least *one* ordering. This is a less restrictive condition than non-interference. As a consequence, plans usually get shorter and can be found more efficiently. Rintanen et al. present propositional encodings for general ADL problems that lead to a subclass of general ∃-step plans with the additional constraint that simultaneously applied operators in a state s have to be already applicable in s.

⋆ Supported by the German Research Council (DFG) as part of the Transregional Collaborative Research Center "Automatic Verification and Analysis of Complex Systems" (SFB/TR 14 AVACS). See `http://www.avacs.org/` for more information.

⋆⋆ Supported by NICTA in the framework of the DPOLP project. NICTA is funded through the Australian Government's *Backing Australia's Ability* initiative, in part through the Australian National Research Council.

In this paper, we show that this concept can be further generalized for STRIPS problems. We present an encoding that allows operators to be executed in parallel in a state s if they are all either already applicable in s or enabled by some other operators that are applied at the same time point. This leads to a wider class of ∃-step plans in which more operators are simultaneously applicable. We call this semantics of parallelism the *relaxed* ∃*-step semantics*. In many planning domains, this leads to efficiency gains because of shorter parallel plan lengths. Furthermore, the formulae get more strongly constrained which can make them easier to solve. One main innovation in this context is the notion of the *disabling-enabling-graph* which is an extension of the disabling graph presented by Rintanen et al. [3].

The structure of this paper is as follows. Section 2 presents basic notations and the formal definition of the new semantics. In Section 3, we present a translation into the propositional logic. Section 4 shows its performance in comparison to the so far most efficient ∃-step encoding and to the encoding used by SATPLAN06 [4], one of the winners of the International Planning Competition 2006 for optimal deterministic planning. Finally, we discuss related work and conclude the paper.

2 Background

2.1 Notation

We consider planning in a setting where the states of the world are represented in terms of a set P of Boolean state variables that take the value *true* or *false*. Each *state* is a valuation on P, i.e. an assignment $s : P \to \{T, F\}$. A literal is a formula of the form a or $\neg a$ for all $a \in P$. For a literal l we define $\overline{l}$ by $\overline{a} = \neg a$ and $\overline{\neg a} = a$. The set of all literals is denoted by $\mathcal{L} = P \cup \{\neg a \,|\, a \in P\}$. We use *operators* for expressing how the state of the world can be changed. As this work restricts to STRIPS operators, we use the term operator for STRIPS operator. An *operator* on a set of state variables P is a tuple $o = \langle p, e\rangle$ where $p \subset \mathcal{L}$ is a set of literals, the *precondition*, and $e \subset \mathcal{L}$ is a set of literals, the *effect*. The operator is *applicable* in a state s if $s \models p$ (where we identify a set of literals with their conjunction). In this case, we define $\text{app}_o(s) = s'$ as the unique state that is obtained from s by making the effect literals of o true and retaining the truth-values of the state variables not occurring in the effect. For sequences of operators $o_1, \dots, o_n$ we define $\text{app}_{o_1;\dots;o_n}(s)$ as $\text{app}_{o_n}(\dots \text{app}_{o_2}(\text{app}_{o_1}(s)) \dots)$. We say that $\text{app}_{\langle p_1,e_1\rangle;\dots;\langle p_n,e_n\rangle}(s)$ is *defined* if and only if $\text{app}_{\langle p_1,e_1\rangle;\dots;\langle p_i,e_i\rangle}(s) \models p_{i+1}$ for $i = 0, \dots, n-1$.

Let $\pi = \langle P, I, O, G\rangle$ be a *planning instance*, consisting of a set P of state variables, a state I on P (the initial state), a set O of operators on P, and a formula G on P (the goal formula). A (sequential) plan for π is a sequence $\sigma = o_1; \dots; o_n$ of operators from O such that $\text{app}_\sigma(I) \models G$, i.e. applying the operators in the given order starting in the initial state is defined (the precondition of every operator is true when the operator is applied) and produces a state that satisfies the goal formula.

In the rest of this paper we also consider plans that are sequences of *sets of operators*, so that at each execution step all operators are simultaneously applied. In contrast to the ∃-step semantics given by Rintanen et al., operators that are applied simultaneously in a state s do not necessarily have to be applicable in s.

Example 1. Let $\pi = \langle P, I, O, G\rangle$ be a planning instance with state variables $P = \{A_i \,|\, i \in \{1,\dots,n\}\}$, initial state I such that $I \models \bigwedge_{i=1}^{n} \neg A_i$, goal state G such that $G = \bigwedge_{i=1}^{n} A_i$ and operators $o_1 = \langle \top, \{A_1\}\rangle$ and $o_i = \langle \{A_{i-1}\}, \{A_i\}\rangle$ for $i = 2,\dots,n$. As there is no state in which more than one precondition is satisfied, the shortest $\exists$-step plan consists of n steps, whereas the more relaxed $\exists$-step plans can reach the goal within only one step, as $o_1,\dots,o_n$ are applicable in parallel. This is because we allow simultaneously applied operators to be enabled by other operators, and o_{i-1} enables o_i for all $i \in \{2,\dots,n\}$.

2.2 Planning as Satisfiability

Planning can be performed by propositional satisfiability testing as follows. Produce formulae $\Phi^0, \Phi^1, \Phi^2, \dots$ such that Φ^i is satisfiable if there is a plan of length i. The formulae are tested for satisfiability in order of increasing plan length, and from the first satisfying assignment that is found a plan is constructed. Length i of a plan means that there are i time points in which a *set* of operators is applied simultaneously. Next we describe the general structure of such a formula Φ^i.

The state variables in a problem instance are $P = \{a_1,\dots,a_n\}$ and the operators are $O = \{o_1,\dots,o_m\}$. For a state variable a we have the propositional variable a^t that expresses the truth-value of a at time point t. Similarly, for an operator o we have o^t for expressing whether o is applied at t. For formulae Φ we denote the formula with all propositional variables superscribed with the index at time point t by Φ^t. The set of propositions where all variables are superscribed with index t is denoted by P^t.

For a planning instance π, a formula is generated to answer the following question. Is there an execution of a sequence of sets of operators taking k time points that reaches a state satisfying G from the initial state I? The structure of such a formula Φ^k is

$$I^0 \wedge R(P^0, P^1) \wedge \cdots \wedge R(P^{k-1}, P^k) \wedge G^k, \tag{1}$$

where I^0 and G^k are formulae describing the initial and goal state (the propositions are marked with 0 and k, respectively), and $R(P^t, P^{t+1})$ describes the transition relation leading from the state at time point t to the state at time point $t+1$ for all $t \in \{0,\dots,k-1\}$. We will introduce a corresponding propositional encoding of that relation according to our semantics.

2.3 Relaxed $\exists$-Step Semantics

Rintanen et al. [3] adapted the linearization idea of Dimopoulos et al. to planning as satisfiability and showed that corresponding encodings in the propositional logic yield efficiency gains in contrast to the standard definition. As they considered general operators (i.e. operators with arbitrary formulae as preconditions and conditional effects), they additionally restricted their semantics such that the operators' preconditions have to be already satisfied in s. In the following, we present a semantics that relaxes this requirement. In contrast to the approach of Rintanen et al., we allow operators to be executed in parallel in a state s even if not all of them are already applicable in s. Such operators have to be enabled by other operators applied at the same time point.

Definition 1 (Relaxed ∃-step plans). *For a set of operators O and an initial state I, a* relaxed ∃-step plan *is a sequence $T = S_1; \ldots; S_l$ of sets of operators such that there is a sequence of states $s_0, \ldots, s_l$ (the execution of T) such that $s_0 = I$ and*

1. *for all $i \in \{1, \ldots, l\}$ there is a total ordering $o_1 < \cdots < o_n$ on S_i such that $app_{o_1;\ldots;o_{j-1}}(s_{i-1}) \models p_j$ for all $o_j = \langle p_j, e_j\rangle \in S_i$, $s_i = app_{o_1;\ldots;o_n}(s_{i-1})$, and*
2. *the set $\bigcup_{\langle p,e\rangle \in S_i} e$ is consistent for all $i \in \{1, \ldots, l\}$.*

In the next section, we synthesize constraints that lead to relaxed ∃-step plans.

3 Propositional Encoding

For every $o = \langle p, e\rangle \in O$ there are the following axioms. First, if o is applied at a time point t, then its effects are true at the next time point. This is expressed by

$$o^t \rightarrow e^{t+1}, \tag{2}$$

where we identify the set e of literals with their conjunction. Second, the value of a state variable does not change if no operator that changes it is applied. Hence for every state variable a we have two formulae, one expressing the conditions for the change of a from false to true, and another from true to false. The formulae are analogous, and here we only give the one for change from true to false:

$$(a^t \wedge \neg a^{t+1}) \rightarrow \bigvee\{o^t | o = \langle p, e\rangle \in O, \neg a \in e\}, \tag{3}$$

where the empty disjunction is the constant $\perp$. Finally, the preconditions of applied operators must be true, and furthermore we need axioms for restricting the parallel application of operators. These *precondition* and *parallelism* axioms are described next. We define the notion of *disabling-enabling-graphs* in order to provide compact encodings in the propositional logic. It generalizes the concept of disabling graphs introduced by Rintanen et al. for the ∃-step semantics.

We first state when two operators *conflict* in their preconditions, i.e. we formulate a necessary condition for being applied simultaneously according to our semantics. Informally speaking, two operators o and o' conflict if there is no reachable state s such that they can be executed sequentially in s. Therefore, they cannot be applied simultaneously in our relaxed ∃-step semantics.

Definition 2 (Conflict). *Let $\pi = \langle P, I, O, G\rangle$ be a planning instance and $o = \langle p, e\rangle$, $o' = \langle p', e'\rangle \in O$. If there are literals $m_1 \in p$ and $m_2 \in p'$, and*

1. *there is no reachable state[1] s such that $s \models (m_1 \wedge m_2)$, and*
2. *there is no operator $o'' = \langle p'', e''\rangle$ such that*
   (a) $m_2 \in e''$,

[1] As testing exactly whether a given state is reachable in a planning instance is already PSPACE-complete (and therefore as hard as planning itself), we approximate this test by using a subclass of 2-literal-invariants as produced by the algorithm by Rintanen [5] which are computable in polynomial time.

(b) there is no literal m such that $m \in e''$ and $\overline{m} \in p'$, and
(c) o, o'' and o'', o' have consistent effects,

then o and o' conflict *in the ordering $o < o'$. They* conflict *if they conflict in both orderings $o < o'$ and $o' < o$.*

Example 2. Let $\pi = \langle P, I, O, G\rangle$ be a planning instance with $P = \{A, B\}$, initial state $I \models A \wedge \neg B$, and operators $O = \{o_1, o_2\}$, where $o_1 = \langle\{A\}, \{\neg A, B\}\rangle$ and $o_2 = \langle\{B\}, \{A, \neg B\}\rangle$. Then o_1 and o_2 conflict as there is no reachable state s such that $s \models A \wedge B$, and there is no operator $o'' = \langle p'', e''\rangle$ to resolve this conflict.

The motivation for using disabling-enabling-graphs is the following. The goal is to identify all sets of operators that "could be applied" simultaneously in our semantics (i.e. the operators do not conflict and have consistent effects), but there is a reachable state such that there is no ordering in which they can be sequentially applied. Such sets are subsets of a *strongly connected component (SCC)* of the disabling-enabling-graph. Strongly connected components can be identified in linear time [6].

Definition 3 (Disabling-Enabling-Graph). *Let $\pi = \langle P, I, O, G\rangle$ be a planning instance. A* disabling-enabling-graph *for π is a directed graph $G = \langle O, E\rangle$ with the following properties: For all $o = \langle p, e\rangle$ and $o' = \langle p', e'\rangle \in O$ there is an edge $(o, o') \in E$, if*

1. *there is a literal $m \in \mathcal{L}$ such that $m \in e$ and $m \in p'$ (*enabling*), or $m \in e'$ and $\overline{m} \in p$ (*disabling*),*
2. *o and o' do not conflict, and*
3. *there is a reachable state s such that $s \models e \wedge e'$ (parallel execution of o and o' leads to a reachable state).*

Example 3. Let $\pi = \langle P, I, O, G\rangle$ be a planning instance with $P = \{A, B, C\}$, initial state $I \models A \wedge B \wedge C$, and operators $O = \{o_1, o_2, o_3\}$, where $o_1 = \langle\{A\}, \{\neg B\}\rangle$, $o_2 = \langle\{B\}, \{\neg C\}\rangle$, and $o_3 = \langle\{C\}, \{\neg A\}\rangle$. In this case, the disabling-enabling-graph only contains disabling edges. There is a strongly connected component indicating that there is no ordering $<$ such that the operators can be executed according to $<$.

As the reachability of states is approximated, the disabling-enabling-graph that we use in our implementation is not necessarily minimal, but it can be computed in polynomial time in the size of the problem instance. This does not affect the correctness of our encoding as *less* operators are simultaneously applicable if the graph is not minimal. The number of edges typically increases in comparison to the disabling graph because enabling edges have to be considered as well. Moreover, the condition to be applied at some time point is relaxed, and we have to consider more operators that "could be applied" in parallel.

We impose a fixed ordering on all SCCs beforehand and allow operator execution only in that ordering. The resulting encoding is stricter than our formal definition of relaxed $\exists$-step semantics and does not always allow all the parallelism that is possible, but it leads to small formulae. For an SCC $S = \langle V, E\rangle$ with $V = \{o_1, \ldots, o_n\}$, fixed ordering $o_1 < \cdots < o_n$ and $i \in \{1, \ldots, n\}$, we denote the set of operators $\{o_{i+1}, \ldots, o_n\}$ that occur after o_i in this ordering by $Succ(S, o_i)$.

In the following, we describe the precondition axioms. In order to get a compact representation, we define for all operators and literals a corresponding set of *enabling operators*.

Definition 4 (Enabling Operators). *Let $o = \langle p, e\rangle$ be an operator, m a literal. The set $enOps(o; m)$ of* enabling operators *for o and m is defined such that $o' = \langle p', e'\rangle \in enOps(o; m)$ if and only if $o' \neq o$, $m \in e'$, o and o' have consistent effects and do not conflict in the ordering $o' < o$.*

Informally speaking, $enOps(o; m)$ contains operators that can be applied in parallel with o and have the literal m in their effect. In order to be applicable in a state s, we require for an operator $o = \langle p, e\rangle$ that for all literals $m \in p$ either $s \models m$ or there is an enabling operator $o' = \langle p', e'\rangle \in enOps(o; m)$ that is applied simultaneously. This is stated in the formula

$$o^t \rightarrow \bigwedge_{m \in p} (\overline{m}^t \rightarrow \bigvee (enOps^t(o; m) \setminus Succ^t(S, o))), \tag{4}$$

where S is the (unique) SCC in which o occurs and all propositions are labeled with time point t, and $enOps(o; m)^t$ is the set of enabling operators that are labeled with t. Operators occurring after o are ruled out because this allows us to use weaker parallelism axioms which are described next.

We guarantee that operators that are applied in parallel can be executed sequentially in at least one way. A sufficient but too strong constraint is the acyclicity of the corresponding disabling-enabling-graph. Instead, we only require that there is no cycle of operators that disable one another. This is sufficient because of our definition of the precondition axioms (4). Therefore, for SCC S and operator o in S, we require that if o is applied, then all operators in S that are disabled by o and occur after o in the fixed ordering are not applied at the same time point.

Let $S = \langle V, E\rangle$ be an SCC, $V = \{o_1, \ldots, o_n\}$, m a literal, and $o_1 < \cdots < o_n$ a fixed ordering. Let $E = \{o = \langle p, e\rangle \in V \mid \overline{m} \in e\}$ be the set of operators that falsify m and $R = \{o = \langle p, e\rangle \in V \mid m \in p\}$ the set of operators that require m to remain true. Similarly to Rintanen et al. [3], we introduce the chain-formula

$$\begin{aligned}
&\bigwedge \big\{o_i \rightarrow a_m^j \mid i < j, o_i \in E, o_j \in R, \{o_{i+1}, \ldots, o_{j-1}\} \cap R = \emptyset\big\} \\
&\cup \big\{a_m^i \rightarrow a_m^j \mid i < j, \{o_i, o_j\} \subseteq R, \{o_{i+1}, \ldots, o_{j-1}\} \cap R = \emptyset\big\} \\
&\cup \big\{a_m^j \rightarrow \neg o_j \mid o_j \in R\big\},
\end{aligned}$$

where the auxiliary variable a_m^j is true if there is an operator $o_i \in E_m$ with $o_i < o_j$ that is applied and falsifies m. The number of auxiliary variables is linear in the number of operators. The key idea behind this chain-formula is to ensure that if an operator o is applied that falsifies a literal m, then all operators occurring *after* o in the fixed ordering that require m to be true cannot be applied.

Example 4. Consider again the planning instance $\langle P, I, O, G\rangle$ with $P = \{A, B, C\}$, initial state $I \models A \wedge B \wedge C$, and operators $O = \{o_1, o_2, o_3\}$, where $o_1 = \langle\{A\}, \{\neg B\}\rangle$, $o_2 = \langle\{B\}, \{\neg C\}\rangle$, and $o_3 = \langle\{C\}, \{\neg A\}\rangle$. As we have seen in Example 3, these operators are not serializable. For a fixed ordering $o_1 < o_2 < o_3$, the chain-encoding leads to the following parallelism axioms. For variable A, the set of operators that falsify A is $E = \{o_3\}$, the set of operators that require A to remain true is $R = \{o_1\}$. As $o_1 < o_3$, we do not need any parallelism axioms for A. For B, we get $E = \{o_1\}$ and $R = \{o_2\}$. Therefore, we have to ensure that if o_1 is applied, then o_2 cannot be applied at the same time point. Hence, we get the parallelism axioms $o_1 \rightarrow a^2_B$ and $a^2_B \rightarrow \neg o_2$. For C, we get the axioms $o_2 \rightarrow a^3_C$ and $a^3_C \rightarrow \neg o_3$ in a similar way.

In our experiments, we have kept the operators' ordering as they have come out of our PDDL front end.

The conjunction of the effect, frame, precondition and parallelism axioms for all operators describes a transition relation $R(P^t, P^{t+1})$ leading from the state at time point t to the successor state. In a state s, if $s \models R(P^t, P^{t+1})$, there is a total ordering $o_1 < \cdots < o_n$ on the parallel operators $o_1, \ldots, o_n$ such that $\text{app}_{o_1;\ldots;o_n}(s)$ is defined. Let *Var(t)* be the set of all variables occurring in $R(P^t, P^{t+1})$.

Theorem 1. *Let $\pi = \langle P, I, O, G\rangle$ be a planning instance, and s a state reachable from I. Let ν be a valuation of Var(t) for some t such that $\nu(a^t) = s(a^t)$ for all $a^t \in P^t$ and $\nu \models R(P^t, P^{t+1})$. Let $\{o \,|\, \nu \models o^t\} = \{o_1, \ldots, o_n\}$ be the set of operators that is applied at time point t. Then there is a total ordering $o_1 < \cdots < o_n$ such that $app_{o_1;\ldots;o_n}(s)$ is defined.*

Proof. (sketch) There is an ordering $o_1 < \cdots < o_n$ such that for all $o, o' \in \{o_1, \ldots, o_n\}$: If o, o' are contained in the same SCC $G_i = \langle V_i, E_i\rangle$, then $o < o'$ iff $o <_i o'$ (where $<_i$ is the ordering imposed on V_i). If o, o' are contained in different SCCs, then $o < o'$ if there is an edge from o to o'. As o, o' do not conflict (the reader may verify this) and have consistent effects for all $o, o' \in \{o_1, \ldots, o_n\}$, it can be shown by induction that $app_{o_1;\ldots;o_n}(s)$ is defined: for all $o_i = \langle p_i, e_i\rangle \in \{o_1, \ldots, o_n\}$ and for all $m \in p_i$ the following holds. There is no $o_j = \langle p_j, e_j\rangle \in \{o_1, \ldots, o_{i-1}\}$ with $\overline{m} \in e_j$. If $s \not\models m$, then there is $o_j = \langle p_j, e_j\rangle \in \{o_1, \ldots, o_{i-1}\}$ with $m \in e_j$. Therefore $app_{o_1;\ldots;o_{i-1}}(s) \models p_i$.

3.1 Optimizations

In order to speed up planning, we additionally use invariants, which are formulae that are true in all states reachable from the initial state. A restricted class of invariants can be identified in polynomial time. In our experiments, we use formulae $l^t_1 \vee l^t_2$ for invariants $l_1 \vee l_2$ as produced by the algorithm by Rintanen [5].

For an operator $o = \langle p, e\rangle$ and a literal m, invariants can also be used to further reduce the number of enabling operators in $enOps(o; m)$. Consider $o' = \langle p', e'\rangle \in enOps(o; m)$. If there is an invariant $l_1 \vee l_2$ such that $\overline{l_1} \in e'$ and $\overline{l_2} \in p$, then o' cannot be used to enable o at the same time point. Hence, such operators o' are ruled out in our implementation of $enOps(o; m)$. Note that this does not affect correctness.

Furthermore, it is useful to keep the SCCs as small as possible because of the fixed ordering that we impose on their operators (the more operators we restrict to be executed in a fixed ordering, the more parallelism will possibly be lost). In order to tackle the problem of increasing SCCs, we add constraints $\neg o^t_1 \vee \neg o^t_2$ for operators o_1 and o_2 that disable each other (i.e. o_1 disables o_2, and vice versa), because such pairs cannot be executed in parallel anyway according to our semantics. The quadratical worst case size is almost never an issue in practice, but the size of the SCCs decreases because we need no edges between o_1 and o_2 in the disabling-enabling-graph.

4 Experiments

We evaluated our new relaxed ∃-step encoding in comparison to the so far most efficient ∃-step encoding [3] and to the encoding used in SATPLAN06 [4], winner of the IPC 2006 for optimal deterministic planning, on a number of benchmarks from the International Planning Competitions 2004 and 2006. We used the SAT Solver SIEGE [7] and averaged the runtimes and serial plan lengths over 50 runs. In cases where a run exceeded 1200 seconds, we took the average runtime of 5 runs. Cases in which a run exceeded 3600 seconds are indicated with a dash. The results for the SATPLAN06, the ∃-step and relaxed ∃-step encoding are compared in Table 1. We report the parallel and average serial plan length, as well as the total evaluation time of the formulae by a sequential evaluation (i.e. if Φ^{i-1} is found unsatisfiable, then test Φ^i for $i = 1, 2, 3, \dots$ until the first satisfiable formula is found). The runtime data only include the time for solving the formulae. The generation of disabling-enabling-graphs is possible almost as efficiently as the generation of disabling graphs [3]. The results were obtained on a PC running at 2.8 GHz with 2 GB main memory, and a LINUX operating system.

In the PSR, Airport and Pathways domains we get (sometimes significantly) better runtimes due to shorter parallel plan lengths. In particular, the PSR domain makes use of this more relaxed semantics. Storage is interesting because except for one instance (storage 17) we do not get shorter plans than with ∃-step semantics, but in that instance the last unsatisfiable formula gets much easier to solve. A more sophisticated heuristic ordering of the SCCs would possibly yield shorter plans and better runtimes in general. Furthermore, we implemented a domain structured similarly to Example 1 in which a hoist has to fill boxes with three objects in a predefined order[2]. We obtain a significant speedup in this setting as the more relaxed ∃-step plans get only half as long as the corresponding ∃-step plans. Moreover, we did experiments for other domains but give only a short overview of the results because of lack of space. In the tested instances, we can summarize the results as follows. Philosophers is easy for all the encodings, a fraction of a second is needed to solve the largest instances. In comparison to the SATPLAN06 encoding, the runtimes of the relaxed ∃-step encoding are (sometimes significantly) lower in the Openstacks and Pipesworld domain, and comparable (i.e. sometimes better, sometimes worse) in TPP. In comparison to the ∃-step encoding, the runtimes are comparable in Pipesworld and TPP, and slightly worse in Openstacks.

The parallel plan lengths of the relaxed ∃-step encoding usually decreases or is equal in comparison to the other encodings. The quality of the serial plans for ∃-step and relaxed ∃-step is comparable: The average serial plans produced by the relaxed ∃-step encoding are equally long or at most slightly longer than those produced by the ∃-step encoding in most of the cases. As we do no postprocessing step to eliminate redundant operators, the serial plans produced by SATPLAN06 are usually shorter than those produced by the (relaxed) ∃-step encodings.

[2] Where boxes n means that n boxes have to be filled.

Table 1. Experimental results with SATPLAN06, ∃-step and relaxed ∃-step encodings

	SP06 parallel length	*SP06 serial length*	*SP06 total time*	*∃-step parallel length*	*∃-step serial length*	*∃-step total time*	*relaxed parallel length*	*relaxed serial length*	*relaxed total time*
psr 46	29	34	246.9	28	42	16.4	16	43	1.1
psr 47	23	27	11.1	21	35	0.6	12	38	0.1
psr 48	26	37	42.7	24	42	2.3	15	43	0.3
psr 49	36	47	874	34	56	132.6	22	58	8.7
airport 17	28	88	0.5	28	88	1	25	88	0.3
airport 18	31	107	3.1	31	108	6.6	26	109	0.5
airport 19	30	90	0.9	30	90	0.8	25	90	0.3
airport 20	32	115	5.4	32	116	14.0	27	115	1.2
pathways 5	9	30	0.04	9	40	0.1	7	47	0.1
pathways 6	12	55	1.2	12	69	1.3	10	69	0.9
pathways 7	13	72	38.3	13	90	7.1	11	97	8.1
pathways 8	-	-	-	-	-	-	14	134	2339
storage 15	9	25	11.5	6	22	0.09	6	23	0.1
storage 16	-	-	-	7	28	0.5	7	29	0.5
storage 17	-	-	-	8	30	78.4	7	31	0.5
storage 18	-	-	-	9	36	50.2	9	38	40.4
boxes 5	21	25	9.8	20	30	2	10	29	0.1
boxes 6	25	30	149.7	24	37	28.3	12	34	0.2
boxes 7	-	-	-	28	45	354	14	52	0.9
boxes 8	-	-	-	-	-	-	16	63	2.6

5 Related Work

Planning as satisfiability was pioneered by Kautz and Selman. They presented a first translation of a planning problem into the propositional logic [8] as well as the standard encoding for parallel operator application [1]. In a further approach [9], they combined planning as satisfiability with the GraphPlan algorithm [10] resulting in the well known BLACKBOX planner. This planner has been further developed to SATPLAN06 [4].

Dimopoulos et al. [2] noticed that the notion of parallel plans used by Blum and Furst can be relaxed to what we have called ∃-step semantics. Cayrol et al. [11] implemented this idea in the GraphPlan framework. Rintanen et al. [3] proposed propositional encodings for the ∃-step semantics for arbitrary operators that restrict parallel operator application in a state s to operators that are already applicable in s.

Van den Briel et al. [12] used a concept of parallelism that is similar to our relaxed ∃-step semantics for an integer programming approach. They, however, presented a loosely constrained approach causing an exponential number of cycle elimination constraints, which their search algorithm has to take into account explicitly. Therefore, they considered it unfeasible to generate all the required constraints for guaranteeing a total ordering of parallel actions, and instead generate the necessary constraints during search. We avoided this problem by enforcing a fixed ordering on the number of operators of non-trivial SCCs. This may rule out some plans, but leads to small formulae which can be solved very efficiently. For our approach, we do not need specialized search algorithms, but can use arbitrary SAT solver to perform the planning task.

6 Conclusions

We have given a translation of a relaxed semantics for parallel planning into SAT and shown that it is efficient in domains of the recent International Planning Competitions if they exploit the relaxation of our semantics. In this case, we are often one order of magnitude faster than the so far most efficient encodings. Our encoding is restricted to STRIPS because the generalization does not seem to be possible for general ADL problems due to disjunctive preconditions of operators. As various planning domains and different semantics have been developed in the last years, it will be interesting for further research if there are general structures of planning domains that are well suited for certain types of semantics. In this context, there is also the question whether there are other classes of parallel plans that lead to efficient planning as satisfiability.

References

1. Kautz, H., Selman, B.: Pushing the Envelope: Planning, Propositional Logic, and Stochastic Search. In: Proceedings of the 13th National Conference on Artificial Intelligence, pp. 1194–1201 (1996)
2. Dimopoulos, Y., Nebel, B., Koehler, J.: Encoding Planning Problems in Nonmonotonic Logic Programs. In: Proceedings of the 4th European Conference on Planning, pp. 169–181 (1997)
3. Rintanen, J., Heljanko, K., Niemelä, I.: Planning as Satisfiability: Parallel Plans and Algorithms for Plan Search. Artificial Intelligence 170, 1031–1080 (2006)
4. Kautz, H., Selman, B., Hoffmann, J.: SatPlan: Planning as Satisfiability. In: Abstracts of the 5th International Planning Competition (2006)
5. Rintanen, J.: A Planning Algorithm not Based on Directional Search. In: Proceedings of the 6th International Conference on Principles of Knowledge Representation and Reasoning, pp. 617–624 (1998)
6. Tarjan, R.E.: Depth-First Search and Linear Graph Algorithms. SIAM Journal on Computing 1, 146–160 (1972)
7. Ryan, L.: Efficient Algorithms for Clause-Learning SAT Solvers. Master's thesis, Simon Fraser University (2004)
8. Kautz, H., Selman, B.: Planning as Satisfiability. In: Proceedings of the 10th European Conference on Artificial Intelligence, pp. 359–363 (1992)
9. Kautz, H., Selman, B.: Unifying SAT-based and Graph-based Planning. In: Proceedings of the 16th International Joint Conference on Artificial Intelligence, pp. 318–325 (1999)
10. Blum, A.L., Furst, M.L.: Fast Planning through Planning Graph Analysis. In: Proceedings of the 14th International Joint Conference on Artificial Intelligence, pp. 1636–1642 (1995)
11. Cayrol, M., Régnier, P., Vidal, V.: Least Commitment in Graphplan. Artificial Intelligence 130, 85–118 (2001)
12. van den Briel, M., Vossen, T., Kambhampati, S.: Reviving Integer Programming Approaches for AI Planning: A Branch-and-Cut Framework. In: Proceedings of the 15th International Conference on Automated Planning and Scheduling, pp. 310–319 (2005)

Forgetting in Logic Programs with Ordered Disjunction

Wu Chen[1,2], Norman Foo[3], and Mingyi Zhang[4]

[1] School of Computer Science and Engineering, Guizhou University, Guizhou, China, 550025
[2] School of Computer and Information Science, Southwest University, Chongqing, China, 400715
chenwu@swu.edu.cn
[3] School of Computer Science and Engineering,UNSW, NSW 2052, Australia
norman@cse.unsw.edu.au
[4] Guizhou Academy of Sciences, Guizhou, China, 550001
zhangmingyi045@yahoo.com.cn

Abstract. We present a new forgetting method in logic programs with ordered disjunction (LPODs). To forget a literal means to give it up from the answer sets of the program. There is a known method for doing this in an extended logic program (ELP) by transforming it into another such program whose answer sets have only lost the forgotten literal. However, a naive application of it to an LPOD can produce undesirable results. Our new method avoids these, and ensures that the answer sets of the LPOD, as specified by its so-called split programs, only lose as few literals as are necessary. To achieve this we introduce two new literals "$\top$" and "$\bot$" into the syntax of LPODs and extend the definition of an answer set accordingly.

1 Introduction

The theory of forgetting in classical proposition logic was introduced by Lin and Reiter [Lin and Reiter 1994]. Informally, to "forget" (or to "give up") a propositional symbol means to re-configure the language so that any formula that contains that symbol is transformed into one that does not while retaining as much information as possible. The proposal of Lin and Reiter was re-considered by Lang [Lang and Marquis 2002], [Lang et. al. 2003] who suggested ways of interpreting a number of contemporary notions in knowledge representation in terms of forgetting. Forgetting has since proved to be useful in algorithms for application domains such as knowledge update and revision, planning, and conflict solving for negotiation. As extended logic programs (ELP) [Gelfond and Lifschitz 91] are widely used to model many knowledge representation problems in AI it is natural to investigate how forgetting can be applied to such programs to realize those applications. In response to this the theory of forgetting in ELPs is now an active research area [Zhang et.al. 05, Wang et.al. 05, Eiter and Wang 06]. An important application of such techniques is to model negotiation between two ELPs [Foo et.al. 05]. Negotiation is a central concern of multi-agent theory in AI. Informally, an agent's stance is its ELP and its potential demands are the answer sets of its program. To negotiate, each agent selects one of its answer sets as an initial demand. It will revise its program by accepting part of its opponent's demand and giving up part of its own demand. The "giving up" phase is modeled as ELP forgetting, which guarantees that all of the agent's demand will be retained except those that it gives up. At some point the agent's preferences need to be considered for the simple reason that even within a given demand (answer) set the agent may value on some items more than others. Hence, models of negotiation with some notion of preference relation among literals of an ELP would be advantageous. As a step in this direction we consider logic programs with ordered disjunctions (LPODs) [Brewka et.al. 04] under answer set semantics [Gelfond and Lifschitz 91]. LPODs can express preference among potential knowledge representation problem solutions qualitatively rather than numerically. The formal relationship of an LPOD to ELPs is the following. An LPOD splits into many ELPs. Every consistent answer set of any split program is an answer set of the LPOD. The semantics of LPODs thus relies on that of its split ELPs, but overlays them with the preference ordering on literals specified by the ordering of the disjunctions. The details will be explained later in this paper when the examples are dicussed.

There are of course realistic aspects of negotiation that we do not address here. Among them are these: *strategies*, *beliefs* and *bounded resources*. No real-world agent enters a negotiation without some kind of strategy, and it has beliefs about the likely behavior of the other agent. Any reasoning it does has to contend with limited resources. These issues are currently being investigated. Here we only consider the essential aspect of preferences by an agent, which must precede the other aspects.

In negotiation between two LPODs, if we simply adopt the known forgetting methods in ELPs to give up part (some literals) of the selected demand, there is an unfortunate side-effect that causes every literals in the demand to be given up. Thus, those methods are not suitable, and that is the motivation for investigating how the intended effect (of only forgetting the literals to be given up) may be implemented as an LPOD transformation.

In this paper, we will consider how to obtain such a method. First of all, it should guarantee that the number of each agent's split programs is unchanged. Furthermore it should ensure each agent keeps its own selected demand unchanged except those literals (demand) to be given up. In order to achieve the aim, two new symbols "$\top$"$(true)$ and "$\bot$"$(false)$ are introduced in LPODs. In the new forgetting method, the literals given up will be replaced with "$\top$" or "$\bot$" under different considerations. Based on the semantics of "$\top$" and "$\bot$" in ELPs, some new rules will be added to the Gelfond-Lifschitz reduction.

The rest of the paper is organized as follows. Section 2 briefly recapitulates some notations of LPODs and ELPs. Then in section 3 we present an extension of LPODs with "$\top$" and "$\bot$", and provide the semantics of these extended LPODs. Based on the semantics of "$\top$" and "$\bot$", some rules containing "$\top$" and "$\bot$" are added into the Gelfond-Lifschitz reduction to obtain a basic program from an ELP. In section 4 our new forgetting method is described and properties of the method are captured. The concluding section summarizes what has been achieved and further work.

2 Preliminaries

In this paper, we will work in a propositional language $\mathcal{L}$. In this section, we recall some basic notations of logic programs following the conventions of [Brewka et.al. 04, Gelfond and Lifschitz 91, Lifschitz 96, Lloyd 87].

LPODs are an extension of ELPs introduced by Brewka and his colleagues [Brewka et.al. 04]. The extension is a new connective $\times$ in LPODs that represents ordered disjunction, and is allowed in the head of rules only. Formally an LPOD is a finite set of rules of the following form:

$$L_0 \times L_1 \times ... \times L_k \leftarrow L_{k+1}, ..., L_m, not\ L_{m+1}, ..., not\ L_n (0 \leq k < m < n) \quad (1)$$

where each $L_i (0 \leq i \leq n)$ is a literal of an underlying propositional language $\mathcal{L}$, and a *literal* is a positive atom a or negated atom $\neg a$. For any atom a, the literals a and $\neg a$ are *complementary* literals. This notation is standard for complementary pairs: for an atom a if the literal L is a then $[\neg]L$ represents $\neg a$, and if L is $\neg a$ then $[\neg]L$ is a. Let r be a rule of the above form. The intuitive reading of the head of r is : if possible L_0, if L_0 is not possible, then L_1,..., if all of $L_0, ..., L_{k-1}$ are not possible, then L_k. The literal $L_i (0 \leq i \leq k)$ are called choices of the rule. We denote its head, positive body and negative body by $Head(r) = \{L_0, ..., L_k\}$, $Pos(r) = \{L_1, \ldots, L_m\}$ and $Neg(r) = \{L_{m+1}, \ldots, L_n\}$ respectively. r is called a *fact* if $Pos(r) = \emptyset$ and $Neg(r) = \emptyset$ when $Head(r) \neq \emptyset$. The rule r is called a *constraint* if $Head(r) = \emptyset$ when $Pos(r) \neq \emptyset$ or $Neg(r) \neq \emptyset$.

ELPs are a special case of LPOD if k=0 for all the rules. The semantics of LPODs is based on the idea that an LPOD can be split into a set of ELPs which reflect the ordering on the literals encoded by the new $\times$ connective. Let Π be an LPOD. For $r \in \Pi$, we call r^j the *jth* option of r, where $r = L_0 \times L_1 \times ... \times L_k \leftarrow body$, $r^j = L_j \leftarrow body$, not L_0, not $L_1, ...,$ not $L_{j-1} (0 \leq j \leq k)$. Π' is a split program of Π if it is obtained from Π by replacing each rule in Π with one of its options.

Example 1. *Let Π consists of the following rules*

$$A \times B \leftarrow not\ E$$
$$C \times D \leftarrow not\ F$$

We obtain 4 split ELPs:

$1) A \leftarrow not\ E$	$2) A \leftarrow not\ E$	$3) B \leftarrow not\ E, not\ A$	$4) B \leftarrow not\ E, not\ A$
$C \leftarrow not\ F$	$D \leftarrow not\ F,\ not\ C$	$C \leftarrow not\ F$	$D \leftarrow not\ F,\ not\ C$

We will now review as much of the machinery as is necessary to keep our exposition self-contained. A *basic program* is an ELP Π such that $Neg(r) = \emptyset$ for any $r \in \Pi$. It resembles what is informally known as a pure logic program or a definite clause program, i.e. no occurrence of negation-as-failure ("not"), but classical negation ($\neg$) can of course appear in literals. By regarding opposite literals as distinct propositional symbols we can use the least model (or equivalently the least fixed point) of the program to specify its semantics. This is essentially the idea behind the following definition of the answer set of a basic program. If Π is a basic program, let Lit be the set of literals in the underlying langauge of Π again. A set of literals is the *answer set* of Π if it is the smallest subset S of Lit such that:

1. for any $r \in \Pi$, if $Pos(r) \subseteq S$, then $Head(r) \subseteq S$;
2. if S contains a pair of complementary of literals, then $S = Lit$.

We will denote the answer set of a basic program Π by $Cn(\Pi)$. An answer set is *consistent* if it contains no complementary literals, otherwise it is *inconsistent*.

An answer set of a (possibly non-basic) ELP is defined via the Gelfond-Lifshitz reduction [Gelfond and Lifschitz 91] which is recapitulated as follows. If Π is an ELP, let Lit be the set of literals of the underlying langauge of Π. For any set $S \subset Lit$, Π can be reduced to a basic program Π^S, which is an ELP with $Neg(r) = \emptyset$ for every $r \in \Pi$, by removing

1. each $r \in \Pi$ if $Neg(r) \cap S \neq \emptyset$;
2. all formulas of the form $not\ L$ in the remaining rules.

Then S is an *answer set* of Π whenever S is an answer set of Π^S.

A set of literals is an answer set of an LPOD Π if it is also a consistent answer set of a split ELP Π' of Π. Thus in example 1, Π has 4 answer sets: $S_1 = \{A, C\}$, $S_2 = \{A, D\}$, $S_3 = \{B, C\}$ and $S_4 = \{B, D\}$. We use $ANS(\Pi)$ to denote the set of answer sets of an LPOD Π. The intended ordering of literals can be used to (partially) order these answer sets, as suggested by [Brewka et.al. 04].

3 LPOD with "$\top$" and "$\bot$"

LPODs with "$\top$" and "$\bot$" are an extension of LPODs. Formally, the new programs have the additional literals "$\top$" and "$\bot$" than LPODs. Before defining the semantics of LPODs with "$\top$" and "$\bot$" an example will illustrate the motivation for extending LPODs in this manner.

Example 2. *In example 1, suppose we give up literal A. We can obtain the new LPOD Π' through a naive application of the forgetting method of [Zhang et.al. 05].*

$$\begin{array}{l} B \leftarrow \textit{not}\ E \\ C \times D \leftarrow \textit{not}\ F \end{array}$$

$$\begin{array}{ll} 1) B \leftarrow \textit{not}\ E & 2) B \leftarrow \textit{not}\ E \\ \quad C \leftarrow \textit{not}\ F & \quad D \leftarrow \textit{not}\ F,\ \textit{not}\ C \end{array}$$

There are at least two undesirable side-effects of this naive application in the above example. One is two split programs are given up, and so are their answer sets — we call this *"program collapse"*. The other is all the answer sets that include A are given up, which departs from the idea of keeping them as unaltered as possible except for the literals given up.

To overcome these limitations we suggest the following criteria. First, in order to avoid program collapse, the literal given up should be replaced with some new literal rather than be deleted. In addition, the new literal should not affect the other literals of the answer set except the literal given up, and yet not introduce new literals into the answer set. It turns out that the two special literals "$\top$" and "$\bot$" mentioned above suffice as the new literals. Their intended meanings are $true$ and $false$, respectively.

The semantics of the new LPODs with "$\top$" and "$\bot$" is only slightly different from the standard LPODs. Intuitively, "$\top$" in the head of a rule means the rule does not convey any information. This is most easily justified if one re-writes a rule $\top \leftarrow \textit{body}$

as a disjunction $\top \vee \neg body$. By the interpretation of $\top$ as *true* this is equivalent to simply *true* which carries no information. On the other hand if the head of a rule has $\bot$, the interpretation of it as *false* says that the rule is a constraint. Moreover, the new literals "$\top$" and "$\bot$" with negation-as-failure *not* will be dealt with in a classical way. In other words, "not $\top$" and "not $\bot$" represent $false$ and $true$, respectively or equivalently "not $\top$" is "$\bot$" and "not $\bot$" is "$\top$". According to the new semantics, the Gelfond-Lifschitz reduction will have the following additional steps to remove the redundant rules and formulas:

1. delete rules r with Head(r)=$\top$;
2. delete rules r with $\bot \in Pos(r)$ or $\top \in Neg(r)$;
3. delete all formulas of the forms $\top$, $\bot$ and *not* $\bot$ in the remaining rules except $r = \bot \leftarrow$.

These new steps can be justified in similar ways to the discussion above, using the intended interpretation of $\top$ and $\bot$, e.g., deleting a rule with $\top \in Neg(r)$ because the body of rules are conjunctions and this *not* $\top$ conjunct is *false* so the rule is never applied.

Observation 1. *Why not delete the formula "$\bot$" in a rule of the form "$\bot \leftarrow$ "? Such a rule in an ELP Π means it has no answer set, as is intuitively seen by the interpretation of it as a fact* false.[1] *Thus we need not delete "$\bot$" in such rules.*

4 Forgetting in LPODs with "$\top$" and "$\bot$"

In forgetting (giving up) some literals from an LPOD, in order to avoid the side-effects mentioned above we replace these literals with "$\top$" or "$\bot$" on depending on the behavior of the forgetting that is required. Because any answer set of an LPOD is consistent there are no complementary literals in any answer set. Thus when forgetting the literal L, the complementary $[\neg]L$ is irrelevant and need not be replaced by the new literals $\top$ or $\bot$. As in [Zhang et.al. 05], we have also two forgetting ways in LPODs.

Definition 1. *Let Π be an LPOD. Suppose literal L is forgotten from Π. We call it $\top$-$forgetting$($\bot$-$forgetting$,resp.) if we replace all occurrences of the literal L in Π with $\top$($\bot$, resp.) except the L in $[\neg]L$.*

The new programs obtained through the above method are denoted by $\top$-$reduce(\Pi, \{L\})$ and $\bot$-$reduce(\Pi, \{L\})$, respectively.

Let us illustrate the above definition in an example.

Example 3. *Let an LPOD Π consists of the following rules*

$$A \times B \leftarrow not\ E$$
$$D \times B \leftarrow not\ F$$

[1] This interpretation can be formally justified as in pure logic programs. Indeed the rule corresponds to one that has both head and body empty, which is not permitted in ELPs.

which has 4 split ELPs:

$1) A \leftarrow not\ E$	$2) A \leftarrow not\ E$	$3) B \leftarrow not\ E, not\ A$	$4) B \leftarrow not\ E, not\ A$
$D \leftarrow not\ F$	$B \leftarrow not\ F,\ not\ D$	$D \leftarrow not\ F$	$B \leftarrow not\ F,\ not\ D$

Π has four answer sets $S_1 = \{A, D\}$, $S_2 = \{A, B\}$, $S_3 = \{B, D\}$ and $S_4 = \{B\}$. Suppose we will forget literal A through $\top$-$forgetting$ and $\bot$-$forgetting$, respectively. First, we check the $\top$-$forgetting$. The new program $\top$-$reduce(\Pi, \{A\})$ is as follows:

$$\begin{aligned} \top \times B &\leftarrow not\ E \\ D \times B &\leftarrow not\ F \end{aligned}$$

We obtain 4 split ELPs:

$1) \top \leftarrow not\ E$	$2) \top \leftarrow not\ E$	$3) B \leftarrow not\ E, not\ \top$	$4) B \leftarrow not\ E, not\ \top$
$D \leftarrow not\ F$	$B \leftarrow not\ F,\ not\ D$	$D \leftarrow not\ F$	$B \leftarrow not\ F,\ not\ D$

Based on the extended Gelfond-Lifschitz reduction, split programs 1) and 3) will be reduced to the same basic program $\{D \leftarrow\}$, thus the two split programs have the same answer set $\{D\}$. Similarly, split programs 2) and 4) will be reduced to the same basic program $\{B \leftarrow\}$, thus the two split programs have the same answer set $\{B\}$. These answer sets that arise from the application of $\top$-forgetting exhibit the following properties: $S_1 \setminus \{A\} \in ANS(\top\text{-}reduce(\Pi, \{A\}))$, $S_2 \setminus \{A\} \in ANS(\top\text{-}reduce(\Pi, \{A\}))$ and $S_4 \setminus \{A\} \in ANS(\top\text{-}reduce(\Pi, \{A\}))$ but $S_3 \setminus \{A\} \notin ANS(\top\text{-}reduce(\Pi, \{A\}))$. In other words, if A belongs to an answer set S of an LPOD Π, then $S \setminus \{A\}$ will be an answer set of the new LPOD Π' through $\top$-$forgetting$; Otherwise, $S \setminus \{A\}$ is not always an answer set of Π'.

Next we examine $\bot$-$reduce(\Pi, \{A\})$. The $\bot$-$reduce(\Pi, \{A\})$ LPOD is:

$$\begin{aligned} \bot \times B &\leftarrow not\ E \\ D \times B &\leftarrow not\ F \end{aligned}$$

We obtain 4 split ELPs:

$1) \bot \leftarrow not\ E$	$2) \bot \leftarrow not\ E$	$3) B \leftarrow not\ E, not\ \bot$	$4) B \leftarrow not\ E, not\ \bot$
$D \leftarrow not\ F$	$B \leftarrow not\ F,\ not\ D$	$D \leftarrow not\ F$	$B \leftarrow not\ F,\ not\ D$

Similarly, $\bot$-$reduce(\Pi, \{A\})$ has two answer sets $\{B, D\}$ and $\{B\}$. This suggests the conclusion: literal A does not belong to an answer set S of Π if and only if S is an answer set of the new LPOD Π' through $\bot$-$forgetting$.

The above examples motivate the main results of the paper. The proofs are included merely for completeness as the examples above should be sufficiently persuasive for an intuitive reading.

Theorem 1. *Let Π be an LPOD. Suppose $S \in ANS(\Pi)$ and $S \setminus \{L\} \neq \emptyset$.*

1. *if $L \in S$, then $S \setminus \{L\} \in ANS(\top\text{-}reduce(\Pi, \{L\}))$;*
2. *$L \notin S$ iff $S \in ANS(\bot\text{-}reduce(\Pi, \{L\}))$.*

Proof 1. *1. According to the definition of answer set of LPODs, S is also a consistent answer set of a split program SP of Π. For convenience, we use Π' to represent $\top$-$reduce(\Pi, \{L\})$ and S' to represent $S \setminus \{L\}$. Π' has a split program SP', which is a transform of SP through $\top$-$forgetting$. We need to show S' is an answer* *set of SP',i.e., $S' = Cn(SP'^{S'})$. Based on the definition of the extended Gelfond-Lifschitz reduction, we know that $SP'^{S'}$ is obtained from SP^S through the following ways:(1)delete all rules $r \in SP^S$ with $Head(r) = \{L\}$; (2) delete all formulas of the form L in the remaining rules in SP^S. For $r \in SP'^{S'}$, if $Pos(r) \subseteq S'$, then $Pos(r) \subset S$. Based on the definition of answer set of basic program, it further follows that $Head(r) \subseteq S$. Because $Head(r) \cap \{L\} = \emptyset$ we conclude that $Head(r) \subseteq S'$.*

Now we need to show that S' is the smallest subset of Lit that satisfies the above property in $SP'^{S'}$. Suppose there is a proper set S'' of S' that satisfies the above property in $SP'^{S'}$. It follows that $S'' \cup \{L\}$ satisfies the above property in SP^S. But then $S'' \cup \{L\} \subset S$, contradicting the answer set definition that S is the smallest set which satisfies the above property in SP^S.

2. ($\Rightarrow$) Suppose S is a consistent answer set of a split program SP of Π and Π' represents $\bot$-reduce$(\Pi, \{L\})$. Π' has a split program SP', which is a transform of SP through $\bot$-$forgetting$. We need to show S is also an answer set of SP',i.e., $S = Cn(SP'^S)$. From the definition of the extended Gelfond-Lifschitz reduction we know that SP'^S is also a transform of SP^S by the following: (1) delete all rules $r \in SP^S$ with $Pos(r) \cap \{L\} \neq \emptyset$; (2) delete all formulas of the form L in the remaining rules in SP^S. Obviously, since $L \notin S$, for any $r \in SP'^S$, if $Pos(r) \subseteq S$ then $Head(r) \subseteq S$.

Now we need to show that S is the smallest subset of Lit that satisfies the above property in SP'^S. S is the smallest subset of Lit satisfying the above property in SP^S. Based on the reduction from SP^S to SP'^S, we know that all the rules and formulas related with S are kept in SP'^S. It follows that S is also the smallest subset of Lit satisfying the above property in SP'^S.

($\Leftarrow$) Since the contrapositive of a proposition is logically equivalent to the proposition, we only need to show: if $L \in S$, S is not an answer set of $\bot$-reduce$(\Pi, \{L\})$. By assumption S is a consistent answer set of a split program SP of Π. Let Π' be $\bot$-reduce$(\Pi, \{L\})$. Π' has a split program SP', which is a transform of SP through $\bot$-$forgetting$. We know that SP'^S is also a transform of SP^S by the extended Gelfond-Lifshitz reduction as above. There is a rule $r \in SP^S$ such that $Head(r) = \{L\}$ and $Pos(r) \subseteq S$. If $Pos(r) = \emptyset$, SP' has no answer set, $S \neq Cn(SP'^S)$. Otherwise, it follows that there is a rule $r' \in SP'^S$ related to r such that $Head(r') = \emptyset$ and $Pos(r') \subseteq S$, i.e., r' is a constraint. Thus $S \notin ANS(\bot$-reduce$(\Pi, \{L\}))$.

From the second result of the above theorem, we know that if a literal L belongs to an answer set S of an LPOD Π, then S will not be an answer set of new LOPD $\bot$-*reduce*$(\Pi, \{L\})$. Now, it would be pleasing if $S \setminus \{L\}$ is also not an answer set of the new program. Unfortunately, example 3 provides a counter-example. But under some conditions the result will hold.

Corollary 1. *Let Π be an LPOD. Suppose S is an answer set of Π, $S \setminus \{L\} \neq \emptyset$ and $S \setminus \{L\} \notin ANS(\Pi)$. If $L \in S$, then $S \setminus \{L\} \notin ANS(\bot$-reduce$(\Pi, \{L\}))$.*

Proof 2. *Suppose Π has $n (n \geq 1)$ answer sets $S_1, ..., S_n$, $L \in \cap_{i=1}^m S_i$ and $L \notin \cup_{i=m+1}^n S_i$ $(1 \leq m \leq n)$. Let S be $S_i (1 \leq i \leq m)$. S is also a consistent answer set of a split program SP of Π. Let Π' be $\bot$-reduce$(\Pi, \{L\})$ and S' be $S \setminus \{L\}$. Π' has a split program SP', which is a transform of SP through $\bot$-$forgetting$. Similar to proof 1, we can conclude that $S' \notin ANS(\bot$-reduce$(\Pi, \{L\}))$. Based on the second* *conclusion of theorem 1, for* $(m+1) \leq i \leq n$*,* $S_i \in ANS(\bot\text{-}reduce(\Pi, \{L\}))$*. But since* $S \setminus \{L\} \notin ANS(\Pi)$ *we conclude that* $S \setminus \{L\} \neq S_i((m+1) \leq i \leq n)$*. Thus* $S \setminus \{L\} \notin ANS(\bot\text{-}reduce(\Pi, \{L\}))$ *if* $L \in S$ *and* $S \setminus \{L\} \notin ANS(\Pi)$*.*

Now we extend the definition 1 to giving up a set of literals X from an LPOD Π. The corresponding expressions are $\top\text{-}reduce(\Pi, X)$ and $\bot\text{-}reduce(\Pi, X)$, respectively. The following is an easy extension of theorem 1 using essentially the same idea in the known methods for forgetting in ELPs, viz., there is no difference in forgetting first a literal L and then a literal K, or vice-versa.

Theorem 2. *Let* Π *be an LPOD. Suppose* $S \in ANS(\Pi)$*.*

1. *if* $X \subset S$*, then* $S \setminus X \in ANS(\top\text{-}reduce(\Pi, X))$*;*
2. $X \cap S = \emptyset$ *iff* $S \in ANS(\bot\text{-}reduce(\Pi, X))$*.*

5 Conclusion and Further Work

In this paper we asked the question of how the notion of forgetting a propositional symbol can be syntactically performed in LPODs, in ways that are analogous to those for standard ELPs. We argued that this is best done by extending the syntax of LPODs to include two new literals "$\top$" and "$\bot$". Based on the intended semantics of these literals, we added new rules to extend the Gelfond-Lifschitz reduction. Then we introduced the new notions of $\top\text{-}forgetting$ and $\bot\text{-}forgetting$ in LPODs and showed by examples that they do exhibit the intuitive behavior of how answer sets are modified when literals are forgotten. These examples suggested properties that faithfully mirror prior work on literal forgetting in ELPs. The main theorems formalize these intuitions. In ongoing work we are considering methods to resolve conflicts in negotiation between two LPODs that model agents with ordered disjunctive demands. As in preceding work, literal forgetting is used to model an agent concession. We are also investigating strategies, beliefs (epistemic literals) and bounded reasoning.

Acknowledgements. We should thank Gerhard Brewka for his insightful discussion and suggestions, and the anonymous referees for their comments. This work was supported in part by the national Natural Science Foundation under grant NSF 60573009 and the first author was also supported in part by the stadholder foundation of Guizhou Province under grant 2005(212).

References

[Brewka et.al. 04] Brewka, G., Niemelá, I., Syrjánen, T.: Logic Programs with Ordered Disjunction. Computational Intelligence 20(2), 335–357 (2004)

[Eiter and Wang 06] Eiter, T., Wang, K.: Forgetting and conflict resolving in disjunctive logic programming. In: AAAI 2006. Proceedings of the 21st National Conference on Artificial Intelligence, AAAI Press, Stanford (2006)

[Foo et.al. 05] Foo, N., Meyer, T., Zhang, Y., Zhang, D.: Negotiating logic programs. In: Proceedings of NRAC05, Sixth Workshop on Nonmonotonic Reasoning, Action and Change, Edinburgh (2005)

[Foo et.al. 04] Foo, N., Meyer, T., Brewka, G.: LPOD Answer Sets and Nash Equilibria. In: Maher, M.J. (ed.) ASIAN 2004. LNCS, vol. 3321, pp. 343–351. Springer, Heidelberg (2004)

[Gelfond and Lifschitz 91] Gelfond, Lifschitz, V.: Classical negation in logic programs and disjunctive databases, New Generation Computing, pp. 365–385 (1991)

[Lang and Marquis 2002] Lang, J., Marquis, P.: Resolving inconsistencies by variable forgetting. In: Proceedings of KR2002, pp. 239–250 (2002)

[Lang et. al. 2003] Lang, J., Liberatore, P., Marquis, P.: Propositional independence - formula-variable independence and forgetting. Journal of Artificial Intelligence Research 18, 391–443 (2003)

[Lifschitz 96] Lifschitz, V.: Foundations of logic programming, in Principles of Knowledge Representation, pp. 69–127. CSLI Publications (1996)

[Lin and Reiter 1994] Lin, F., Reiter, R.: Forget it! In Working Notes of AAAI Fall Symposium on Relevance, pp. 154–159 (1994)

[Lloyd 87] Lloyd, J.W.: Foundation of logic programming, 2nd edn. Springer, New York (1987)

[Wang et.al. 05] Wang, K., Sattar, A., Su, K.: A theory of forgetting in logic programming. In: Proceedings of the 20th National Conference on Artificial Intelligence (AAAI-2005), pp. 682–687. AAAI Press, Stanford, California, USA (2005)

[Zhang et.al. 05] Zhang, Y., Foo, N., Wang, K.: Solving logic program conflicts through strong and weak forgettings. In: Proceedings of the 19th International Joint Conference on Artificial Intelligence (IJCAI-2005), Edinburgh, pp. 627–632 (2005)

The ‘Majority’ and ‘by Default’ Modalities

Victor Jauregui

The School of Computer Science and Engineering, and
National ICT Australia Ltd,
The University of New South Wales,
Sydney, Australia
vicj@cse.unsw.edu.au

Abstract. This paper investigates the modality ‘true by default’, associating it with those propositions which hold over a ‘majority’ of possible worlds. A semantic account of ‘majorities’ as the ‘large subsets’ of a set is furnished, and a modal logic of ‘majority default reasoning’ based on this semantics is axiomatised and proved sound and complete.

1 Introduction

Like the notions of being ‘necessarily true’, and ‘possibly true’; the notion of being ‘mostly true’, or ‘majority true’, can be regarded as a ‘mode’ of truth: a modality. Moreover, this modality can be employed to characterise the modality ‘true by default’, which underpins notions of default reasoning, as the modality associated with those propositions which hold in a ‘majority’ of possible worlds. Consider a set $\mathcal{W}$ denoting a set of ‘possible worlds’, and suppose the set $A \subseteq \mathcal{W}$ represents an intuitively ‘large subset’ of $\mathcal{W}$. Moreover, for some ‘proposition’ φ suppose the set of φ-worlds, denoted $[\![\varphi]\!]$, includes A; i.e., φ holds over A. In this respect φ is regarded to be ‘true by default’ over $\mathcal{W}$ as it is true over a ‘majority’ (a large subset: A) of possible worlds. One of the key notions in this paper is that of a ‘large subset’, from which ‘majorities’ are defined.

For example, consider the set of worlds $\mathcal{W} = \{1, 2, 3, 4, 5\}$, where each is identified by its index under some enumeration. Suppose only incomplete knowledge about which of these worlds was actually the case was available (assuming one of these was the actual world). If one knew, however, that φ was true in four out of the five worlds, then one might infer/assume that φ was ‘true by default’, as, all things being equal, there would be a fairly good chance that the actual world was one of these five worlds; that is, four out of five can reasonably be said to constitute a ‘majority’. Consequently, one natural way (among others) to define majorities (for this finite $\mathcal{W}$) so as to respect the intuition of ‘true by default’ would be to define the collection of ‘large subsets’ $\mathcal{K}$ of $\mathcal{W}$ as the sets of cardinality greater than or equal to four, as shown in Fig. 1; where ‘_’ represents the explicit omission of the corresponding entry.

$$\mathcal{K} = \left\{ \begin{array}{l} \{\_\,, 2\,, 3\,, 4\,, 5\}, \\ \{1\,, \_\,, 3\,, 4\,, 5\}, \\ \{1\,, 2\,, \_\,, 4\,, 5\}, \\ \{1\,, 2\,, 3\,, \_\,, 5\}, \\ \{1\,, 2\,, 3\,, 4\,, \_\}, \\ \{1\,, 2\,, 3\,, 4\,, 5\} \end{array} \right\}$$

Fig. 1. The collection $\mathcal{K}$ of ‘large subsets’ of the ‘space’ of possible worlds $\mathcal{W}$

If for some φ, there is a large subset $A \in \mathcal{K}$ such that $A \subseteq [\![\varphi]\!]$, then intuitively one might reason (i.e., assume) by default that φ holds. Observe that $\{\_, 2, 3, 4, 5\}$, and $\{1, \_, 3, 4, 5\}$ are both in $\mathcal{K}$—they both represent 'majorities'—however, their intersection $\{\_, 2, 3, 4, 5\} \cap \{1, \_, 3, 4, 5\} = \{\_, \_, 3, 4, 5\}$, being of cardinality three, is not. So the collection of majorities $\mathcal{K}$ might be expected to not be closed under intersection. Moreover, note that the intersection closure for $\mathcal{K}$ yields the empty set, $\varnothing$, which, in any event, would be hard to justify as a majority of $\mathcal{W}$.

More generally, the objective is to define those propositions which are 'true by default', by interpreting them as those which hold over a 'majority' of possible worlds. This perspective is to be contrasted with those of, for example, Kraus *et al.* [6], Shoham [13], Delgrande [4], etc., whose semantics for propositions which are 'true by default' is based on the notion of 'preferred', or 'most normal' worlds. The view put forward here is that there may be a number of complementary, rather than competing, semantics for the modality 'true by default'. This paper explores one based on the notion of 'majority'. As will be seen, these two perspectives are not purely philosophical/ontological distinctions; they reveal a number of technical differences which will be discussed in Section 5.

Accounts of 'majority'. It should be stressed that the notion of 'majority' can itself be construed in a number of ways and/or to a number of degrees; from 'majority' as "half or more", through to 'majority' as "an overwhelming proportion". These different interpretations may sanction very different inferences. As already intimated, the 'default' interpretation intended here typically lies somewhere between the two extremes. In view of this degree of freedom, the basic definitions given are deliberately weak, with the intention that it is always easier to strengthen them in preference to losing generality at the outset.

Ben-David and Ben-Eliyahu-Zohary [1] employ *filters*: non-empty collections of sets closed under supersets and (finite) intersections, to capture the notion of a 'majority'. Closure under intersection has been questioned by some (e.g., Schlechta [11,12] and Pacuit and Salame [10]), especially for finite sets (see also the discussion of Makinson [9] on probabilistic approaches of nonmonotonic reasoning). Moreover, among the families of filters are the 'improper filters'—those filters which comprise the entire power-set—which contain the empty set. Intuitively it is hard to justify the empty set being a majority of any set except possibly itself. Schlechta [11] obviates this problem by requiring that any two large sets must have non-empty intersection; which led to his notion of a 'weak filter' to characterise 'majorities'. The notion of 'majority' in this paper is shown to be closely related to that of Schlechta.

A modal account of 'majority' has been given by Pacuit and Salame [10]. Their definition of a 'majority' differs from the one presented here in a number of ways, which can be accounted for by their respective motivations; here majorities are interpreted as 'significant majorities': sufficient to sanction 'default inferences'; whereas theirs is intended to capture 'voting majorities', where the relevant notion corresponds to "half or more". The intended notion in this paper lies somewhere between “half or more”, at the weaker extreme, and, at the other, “in the overwhelming majority”.

Overview. The presentation below is an attempt to give an alternative semantics to the modal notion of ‘majority’ adequate for the purposes of defining propositions which are ‘true by default’. Section 2 describes a basic modal language. In Section 3 a definition of the ‘large subsets’ of a set is given, which, in turn, is used to define ‘majorities’ and hence ‘majority validity’: the central concept of this paper. Section 4 presents a semantics in the generalised modal style of Chellas [3]. An investigation of the properties of this notion of ‘majority’ is conducted in Section 5, in which a comparison with the account of Ben-David and Ben-Eliyahu-Zohary [1] is conducted. It is shown that the notion presented here differs from, and arguably improves upon, that of Ben-David and Ben-Eliyahu-Zohary. A sound and complete axiomatisation is presented in Section 6. Section 7 reveals the relationship between the modality furnished here and the property of ‘normality’ which represents logically well-behaved modalities. Notably, ‘normality’ is a feature of the filter-based modality of Ben-David and Ben-Eliyahu-Zohary which is absent in the approach presented in this paper. The condition of ‘maximal coherence’, introduced in Section 8, shows how the present modality can be ‘re-normalised’.

2 The ‘Majority’ Modality

This section introduces a simple language incorporating a modality which will receive the interpretation ‘majority true’. For this purpose, the language $\mathcal{L}$, based on some countable set of propositional variables P, is defined as follows:

$$P \subseteq \mathcal{L}, \qquad \bot \in \mathcal{L}, \qquad \text{if } \varphi, \psi \in \mathcal{L}, \text{ then } \varphi \to \psi, \Box\varphi \in \mathcal{L}.$$

The other connectives are taken to be defined from these: e.g., $\neg\varphi$ is $\varphi \to \bot$, $\top$ is $\neg\bot$, and $\Diamond\varphi$ denotes $\neg\Box\neg\varphi$, and so on. This will be the language which will be considered in this paper, and to which all subsequent discussion of syntax will refer.

In this language a modal logic of ‘majority’ can be defined. For this logic, the modal connective $\Box$, is interpreted as a ‘majority’ modality, and the expression $\Box\varphi$, for some proposition $\varphi \in \mathcal{L}$, reads: “φ is *mostly true*”, or “φ is *majority true*”. More specifically, the notion of ‘majority’ defined here is intended to reflect one which agrees with the interpretation ‘true by default’. So the expression $\Box\varphi$ could equally receive the semantics “φ is *true by default*”. In this respect the logic furnished subsequently may also be regarded as a ‘majority default logic’: a logic for default reasoning based on majorities.

3 Large Sets; Majorities

Commonly, in models based on a set of ‘possible worlds’ $\mathcal{W}$, the proposition φ is regarded *valid* if it holds in all possible worlds; i.e., over all of $\mathcal{W}$: given any possible world $w \in \mathcal{W}$, it is *certain* that φ will hold at w. Extending this argument, in this paper a formula φ is deemed 'true by default' if it holds over a 'majority' of possible worlds: given any w it is 'reasonably certain' that φ will hold at w.[1]

This section defines the 'large subsets' of a given set, from which 'majorities' are derived. A formula is deemed 'true by default' if it holds (or is valid) to a 'large degree', or 'in a majority of possible worlds'; i.e., over some 'large subset' of the set $\mathcal{W}$.

Large subsets. Let $\mathcal{K} \subseteq 2^{\mathcal{W}}$ be a non-empty collection of subsets of $\mathcal{W}$. $\mathcal{K}$ represents a collection of *large subsets* of $\mathcal{W}$ if it satisfies the following axioms:

1. if $S \in \mathcal{K}$ is 'large', then any $T \subseteq \mathcal{W}$ such that $S \subseteq T$, must also be 'large' (i.e., $T \in \mathcal{K}$): $\mathcal{K}$ is closed under supersets;
2. it cannot be that both a set S and its complement (w.r.t. $\mathcal{W}$) $\overline{S}$ are 'large': $S \notin \mathcal{K}$ or $\overline{S} \notin \mathcal{K}$;

These conditions are reasonably intuitive: any set which includes a 'large' set must itself be large; and, if a set is deemed 'large', then its complement cannot be large. Note, however, that large sets need not be closed under intersection; this ensures that the possibility of two large sets having 'non-large' overlap is not ruled out *a priori*—for justification, consider the combinatorial example sketched earlier.

While not incontrovertible, these intuitions represent a plausible basis for the interpretation of majority as a "significant proportion", adequate to capture 'default inferences'. It follows from the definition that:

Proposition 1. *For $\mathcal{K} \subseteq 2^{\mathcal{W}}$ a collection of large subsets of $\mathcal{W}$:*

- *$\mathcal{W}$ is a large set: i.e., $\mathcal{W} \in \mathcal{K}$;*
- *$\varnothing$ is not large: $\varnothing \notin \mathcal{K}$;*
- *if $S, T \in \mathcal{K}$, then $S \cap T \neq \varnothing$.* $(\star)$

These properties are fairly intuitive for an interpretation of 'majority', lending further support to the feasibility of the notion as defined above.

Schlechta's [11] definition of large subsets uses the last of these three (labelled $\star$ above) in place of Axiom 2 above in the definition of large sets. In conjunction with upward-closure (Axiom 1), the two are provably equivalent.

Proposition 2. *Replacing Axiom 2 by $(\star)$, entails the same collections of large sets.*

It should be pointed out that, as the different form of the axioms suggests, the definition presented in this paper was conceived independently of that of Schlechta. In view of this it is worthy of note that the two formulations turn out to be equivalent.

[1] This suggests that worlds occur with equal likelihood.

4 Majority Semantics

Following Chellas [3], a *generalised modal frame* is a pair $(\mathcal{W}, \rho)$, such that $\rho : \mathcal{W} \to 2^{2^{\mathcal{W}}}$ maps a world w into a set of 'propositions' $\rho(w) \subseteq 2^{\mathcal{W}}$; where the term 'proposition', as used here in a semantical context, assumes the technical meaning of a set of worlds.[2]

A *majority frame* is defined to be a generalised modal frame $(\mathcal{W}, \rho)$, such that for each $w \in \mathcal{W}$, $\rho(w)$, satisfies the axioms of large sets; i.e., at each w, $\rho(w)$ is a collection of large sets. The members of $\rho(w)$, represent the 'majorities' of $\mathcal{W}$ considered at world w. Note that this definition entertains the possibility that different collections of sets are considered large at different worlds. Ben-David and Ben-Eliyahu-Zohary [1] argue for this 'subjectiveness' in their approach; e.g., a greater degree of confidence might be required to assume that a particular nut and bolt hold fast for an astronaut than for a person operating a lawn-mower.

A *majority structure* is a tuple $\mathcal{M} = (\mathcal{W}, \rho, \nu)$, with $\nu : P \to 2^{\mathcal{W}}$; i.e., a majority structure is a majority frame with a valuation superimposed over it. The class of all majority frames/structures is denoted N.

The relation of 'truth' between a formula φ, in a majority structure $\mathcal{M}$, at a world w, is denoted $w \models_{\mathcal{M}} \varphi$, and is defined, inductively, as follows:

$$\begin{array}{rll} w \models_{\mathcal{M}} p & \text{iff} & w \in \nu(p); \qquad \text{for } p \in P \\ w \not\models_{\mathcal{M}} \bot; & & \\ w \models_{\mathcal{M}} \varphi \to \psi & \text{iff} & w \not\models_{\mathcal{M}} \varphi \text{ or } w \models_{\mathcal{M}} \psi; \\ w \models_{\mathcal{M}} \Box\varphi & \text{iff} & \text{there is some } S \in \rho(w), \text{ such that } S \subseteq [\![\varphi]\!]. \end{array}$$

Note that, as for each $w \in \mathcal{W}$, $\rho(w)$ is upward closed, then equivalently:

$$w \models_{\mathcal{M}} \Box\varphi \quad \text{iff} \quad [\![\varphi]\!] \in \rho(w).$$

Majority validity. A formula φ is valid in a majority structure $\mathcal{M}$, denoted $\models_{\mathcal{M}} \varphi$, iff $w \models_{\mathcal{M}} \varphi$, for every $w \in \mathcal{W}$. Moreover, φ is said to be *majority valid*, or N-*valid*, denoted $\models_{\mathsf{N}} \varphi$, iff it is valid in every majority frame; i.e., $\forall \mathcal{M} \in \mathsf{N}$, $\models_{\mathcal{M}} \varphi$.

5 The Properties of $\Box$

This section investigates the properties of the 'logic' furnished by N; i.e., of the set of all formulae valid in all majority frames: $\mathcal{L}_{\mathsf{N}} = \{\varphi \in \mathcal{L} \mid \ \models_{\mathsf{N}} \varphi\}$. In particular, it is shown that the logic which results is not normal; a result which will be of importance in the investigation of 'coherence' in Section 7.

The remainder of this section is devoted to revealing the properties of $\mathcal{L}_{\mathsf{N}}$, with a view to constructing a deductive system which axiomatises it.

[2] Chellas calls these *minimal models/frames*. To avoid confusion with the same term in the nonmonotonic reasoning community, in this paper they are renamed 'generalised frames/structures'.

Proposition 3. *The following properties can be verified to hold for any given majority frame $\mathcal{M}$ (i.e., w.r.t. $\models_{\mathcal{M}}$):*

$$\begin{array}{rl|rl}
N: & \Box\top & RM: & \dfrac{\varphi \to \psi}{\Box\varphi \to \Box\psi} \\
D: & \Box\varphi \to \Diamond\varphi & & \\
M: & \Box(\varphi \wedge \psi) \to (\Box\varphi \wedge \Box\psi) & RN: & \dfrac{\varphi}{\Box\varphi} \\
Q: & \neg\Box\bot & &
\end{array}$$

Corollary 1. *The properties N, D, M, Q, RM, RN hold w.r.t. $\models_{\mathsf{N}}$; i.e., they are all N-valid.*

These properties express, respectively, that: absolute truth holds in the majority of cases (N); if something holds in a majority of cases, then its negation does not (D); if in conjunction two properties comprise a majority, then each is a majority on its own (M); Q expresses that the empty set (the set of worlds for which $\bot$ holds) is not a majority. The last two rules express that a proposition entailed by another which holds in a majority of cases, itself must hold in a majority of cases; and that anything that holds universally holds in a majority of cases.

Counter properties. It is also worthwhile pointing out some important principles which are not N-valid. Consider:

$$C: \quad (\Box\varphi \wedge \Box\psi) \to \Box(\varphi \wedge \psi);$$

which is the converse of M and equates to the statement that majorities are closed under intersection. Intuitively, it should be clear that this axiom won't hold from the discussion of how large sets were defined earlier: it may be that φ is holds in a majority of cases, as does ψ (albeit in different majorities), but the cases in which both hold do not comprise a majority.

Normal logics. 'Normal logics' have traditionally held a prominent place in studies on modal logic, because, in some respects, they comprise accounts of modalities which are logically 'well behaved'.

A modal logic is *normal* iff it satisfies:[3]

$$\begin{array}{rl}
N: & \Box\top \\
C: & (\Box\varphi \wedge \Box\psi) \to \Box(\varphi \wedge \psi) \\
RM: & \dfrac{\varphi \to \psi}{\Box\varphi \to \Box\psi}
\end{array}$$

From this it can be seen (as is shown by Chellas in [3]) that:

Theorem 1 (Chellas [3]). *Normal logics can be characterised as those logics determined by classes of generalised frames for which, for each $w \in \mathcal{W}$, $\rho(w)$ is a filter.*

[3] See, for example, Chellas [3].

In particular, N ensures $\rho(w)$ is non-empty, M/RM yields superset closure, and C gives closure under intersection. Note that $\Box\bot$ is consistent with the smallest normal logic (denoted K); i.e., $\nvdash_K \neg\Box\bot$; hence K admits improper filters. This is in contrast with the property D which rules them out.

In fact, Theorem 1 indicates that the filter-based semantics given by Ben-David and Ben-Eliyahu-Zohary reflects a 'normal' account of the 'majority' modality (hence the 'by default' modality). Their basic system can be shown to employ to the 'simplest' normal modality; i.e., the smallest normal logic K.[4]

From the characterisations above, it is interesting to note that, in conjunction with N and RM, C is necessary for a logic to be normal. It therefore follows, from the earlier observation showing that C is not validated in N, that, in contrast to Ben-David and Ben-Eliyahu-Zohary's filter-based semantics, the logic $\mathcal{L}_\mathsf{N}$ is not normal, and thus cannot be characterised by the class of standard modal frames, nor of generalised filter frames.

Moreover, 'preferential' accounts of default reasoning; where φ is 'true by default' in some subset of 'preferred worlds', imply that these propositions comprise a principal filter, necessarily making the modality: 'true by default', a normal one. This makes precise the earlier claim that the 'preferential' and 'majority' perspectives to default reasoning involve tangible differences. Moreover, the implication is that *approaches to default reasoning based on majorities should not employ filters, nor any other semantics which comprises a normal modality.*

6 Axiomatisation: The Logic Θ

In this section an axiomatisation of the logic expressed by $\mathcal{L}_\mathsf{N}$ will be compiled. The subsequent logic, called Θ, will be shown to coincide with $\mathcal{L}_\mathsf{N}$; i.e., soundness and completeness will be shown for Θ with respect to N.

Let $\Theta \subseteq \mathcal{L}$ be the smallest logic containing axioms N and D, and closed under the rule RM:[5]

$$\boxed{\begin{array}{rc} N: & \Box\top \\ D: & \Box\varphi \to \Diamond\varphi \\ RM: & \dfrac{\varphi \to \psi}{\Box\varphi \to \Box\psi} \end{array}}$$

This nomenclature, once more, is borrowed from Chellas [3].

The choice of axioms above is designed to correspond naturally to the definition of large sets given in Section 3. In particular, N reflects that the collection of majorities, being non-empty, must contain $\mathcal{W}$; D reflects that if a set is a majority, its complement cannot be; and RM reflects closure under supersets. Note also that this axiomatisation almost coincides with that given previously for normal logics in Section 5 except that in Θ the axiom D replaces axiom C.

[4] When the basic modality behind their semantics is considered.

[5] Note that a logic is already implied to contain all tautologies, and be closed under Detachment.

Soundness. In the previous section it was shown that each of N and D is valid in the class of all majority frames N, and that the rule RM preserves N-validity. Consequently, Θ is sound w.r.t. N; i.e., $\models_{\mathsf{N}} \varphi$ when $\vdash_{\Theta} \varphi$.

Completeness. To show that Θ is complete w.r.t. N: i.e., that if $\models_{\mathsf{N}} \varphi$, then $\vdash_{\Theta} \varphi$, a canonical majority frame for Θ will be constructed. For background on canonical frames and completeness see, for example, Chellas [3]. This construction, which is omitted due to limitations on space, shows:

Theorem 2 (Completeness of Θ for N). *The axiomatisation above, for Θ, is complete for the class of majority frames N. That is, for any $\varphi \in \mathcal{L}$, if $\models_{\mathsf{N}} \varphi$, then $\vdash_{\Theta} \varphi$.*

7 Normality and Coherence

In Section 5 it was established that the modal logic Θ is not normal. Normal logics traditionally have been the most celebrated modal logics, largely as they were those logics characterised by classes of standard frames: the original semantics proposed by Kripke [7]. As the name suggests, normal modalities can be viewed as being 'well behaved'. Significantly, normality turns out to be one property which distinguishes the modality of majority introduced here with the corresponding filter-based notion of Ben-David and Ben-Eliyahu-Zohary [1].

It turns out that the condition of normality is closely related to the notion of 'maximal coherence' defined in this section. This section introduces the notion of 'coherence', and how it relates to the 'normality' property of modal logics. More precisely, 'maximal coherence' is shown to 're-normalise' the majority semantics presented above; forging the link between this semantics and that of Ben-David and Ben-Eliyahu-Zohary. Informally, 'coherence' is a property which characterises how 'concentrated' the consequences of a theory are. It is used as a generalisation of 'consistency': consistency concerns the existence of non-empty overlap of worlds/models; 'coherence' generalises this to the existence of 'significant overlap'.

Coherent theories. A collection of propositions is called a *theory*. Each member of the theory is a *consequence* of the theory: i.e., a theory comprises the collection of its consequences. Classically, the set of logical consequences of a theory comprises a filter; i.e., a filter represents a theory closed under logical consequence. For a theory $\mathcal{U}$ let $Cn(\mathcal{U})$ denote the 'logical closure' of $\mathcal{U}$; i.e., the smallest filter containing $\mathcal{U}$. Note that, Cn is a *closure operator*; i.e.,

– $\mathcal{U} \subseteq Cn(\mathcal{U})$; (expansion)
– if $\mathcal{U} \subseteq \mathcal{V}$, then $Cn(\mathcal{U}) \subseteq Cn(\mathcal{V})$; (monotonicity)
– $Cn(Cn(\mathcal{U})) \subseteq Cn(\mathcal{U})$; (idempotence)

A theory $\mathcal{U}$ is said to be *coherent* if for every tuple $S_1, \ldots, S_k \in \mathcal{U}$, $(1 \leq k \leq \omega)$, there exists a large set $A \in \mathcal{K}$, such that $A \subseteq S_1 \cap \cdots \cap S_k$.

Proposition 4. *A theory $\mathcal{U}$ is coherent iff $Cn(\mathcal{U}) \subseteq \mathcal{K}$.*

This characterises a coherent theory as one which admits only classical consequences which are 'weak'; i.e., consequences which are 'true by default'.

Theories comprising collections of large sets are theories of majority valid propositions. However, only coherent theories exclude classical consequences which are not majority valid; as $Cn(\mathcal{U}) \subseteq \mathcal{K}$ if $\mathcal{U}$ is coherent. In this respect, only coherent theories should be considered to truly consist solely of majority valid consequences. These represent 'weak' theories, in the sense that they sanction only majority valid consequences.

Maximal coherence; proper filters. The notion of coherence is intended to help shed some insight into the definition of large sets presented above, and its relationship to, e.g., the filter-based approach of Ben-David and Ben-Eliyahu-Zohary. In the investigation that follows, an arbitrary collection of large sets $\mathcal{K}$, is used; the generalisation to systems of large sets—as is the case in the frames considered here—is straight-forward.

A coherent theory $\mathcal{U}$ is said to be *maximal* if there's no coherent theory $\mathcal{V}$ which properly includes $\mathcal{U}$. Maximal coherent collections are those sub-collections of $\mathcal{K}$ which include as many members of $\mathcal{K}$ as possible while retaining coherence. They comprise the largest sub-theories of $\mathcal{K}$ which remain weak.

The main result of this section is the following characterisation of filters in terms of coherent theories:

Lemma 1. *Every maximal coherent theory is a proper filter over $\mathcal{K}$.*

This establishes the link between coherence and filters. In view of Theorem 1, in Section 5, it also reveals the connection between coherence, the semantics of large sets (majority frames) introduced earlier, and normality: the maximal coherent collections of subsets of $\mathcal{K}$ correspond to the filters of Ben-David and Ben-Eliyahu-Zohary.

8 Discussion and Conclusions

This paper introduced an account of 'majorities' with the intention of associating the modality 'true by default' with those propositions which hold over a 'majority' of possible worlds. This notion of majority, in particular, differs from, and arguably improves upon, that of Ben-David and Ben-Eliyahu-Zohary [1] in that it is not closed under intersection—the conjunction of two separate majorities may not be a majority—and prohibits the counterintuitive property of allowing the empty set to comprise a majority.

These concepts were formalised to furnish a modal semantics of 'majority' which was axiomatised to give a non-normal modal logic of 'majority default reasoning': Θ, that was proved sound and complete w.r.t. these semantic notions.

The notion of 'coherence' formulated in Section 7 can be viewed as a property that returns the modality of 'majority' presented to 'normality' (in the modal sense), making it compatible with Ben-David and Ben-Eliyahu-Zohary's account. Normal modalities tend to be 'well behaved', making it a notable property for a logic to have.

'Normality', in the modal sense, is a feature present in approaches in which a unique set of 'preferred' worlds is used as a semantics for the 'true by default' modality; e.g., Kraus *et al.* [6], Shoham [13], Delgrande [4]. This makes precise the claim that an account of default reasoning based on majority considerations (at least one in the spirit of that which appears here and in [11]) is irreconcilable with those accounts based on 'preferred' worlds. However, as was argued in this paper, 'normal' modalities are not well suited to capture 'majority' based default reasoning without introducing counterintuitive anomalies (e.g., intersection closure).

References

1. Ben-David, S., Ben-Eliyahu-Zohary, R.: A modal logic for subjective default reasoning. Artificial intelligence 116, 217–236 (2000)
2. Chellas, B.F.: Basic conditional logic. Journal of philosophical logic 4, 133–153 (1975)
3. Chellas, B.F.: Modal logic: an introduction. Cambridge University press, Cambridge (1980)
4. Delgrande, J.P.: An approach to default reasoning based on a first-order conditional logic. In: Proceedings, AAAI-87, pp. 340–345 (1987)
5. Gabbay, D.: Theoretical foundations for non-monotonic reasoning in expert systems. In: Apt, K.R. (ed.) NATO advanced study institute on logics and models of concurrent systems, pp. 439–457. Springer, New York (1985)
6. Kraus, S., Lehmann, D., Magidor, M.: Nonmonotonic reasoning, preferential models and cumulative logics. Artificial intelligence 44, 167–207 (1990)
7. Kripke, S.: Semantical considerations on modal logic. philosophica fennica 16, 83–94 (1963)
8. Makinson, D.: General theory of cumulative inference. In: Reinfrank, M., Ginsberg, M.L., de Kleer, J., Sandewall, E. (eds.) Non-Monotonic Reasoning. LNCS, vol. 346, pp. 1–18. Springer, Heidelberg (1988)
9. Makinson, D.: Bridges from classical to nonmonotonic logic. Kings college (2005)
10. Pacuit, E., Salame, S.: Majority logic. In: Dubois, D., Welty, C.A., Williams, M.-A. (eds.) KR, pp. 598–605. AAAI Press, Stanford, California, USA (2004)
11. Schlechta, K.: Filters and partial orders. Logic Journal of the IGPL 5(5), 753–772 (1997)
12. Schlechta, K.: Coherent systems. Elsevier, Amsterdam (2004)
13. Shoham, Y.: Reasoning about change. MIT press, Cambridge (1988)
14. Sikorski, R.: Boolean algebras, 2nd edn. Springer, Heidelberg (1964)

Restricted Higher-Order Anti-Unification for Analogy Making

Ulf Krumnack, Angela Schwering, Helmar Gust, and Kai-Uwe Kühnberger

Institute of Cognitive Science, Osnabrück
{krumnack,aschweri,hgust,kkuehnbe}@uos.de

Abstract. Anti-unification has often be used as a tool for analogy making. But while first-order anti-unification is too simple for many applications, general higher-order anti-unification is too complex and leads into theoretical difficulties. In this paper we present a restricted framework for higher-order substitutions and show that anti-unification is well-defined in this setting. A complexity measure for generalizations can be introduced in a quite natural way, which allows for selecting preferred generalizations. An algorithm for computing such generalizations is presented and the utility of complexity for anti-unifying sets of terms is discussed by an extended example.

1 Introduction

Analogies are a central construct of human thinking [1,2,3] and are considered to be an indispensable tool for scientific progress [4]. Analogical reasoning is a high-level cognitive process in which two conceptualizations from a source and a target domain are compared and analyzed for common patterns. The purpose of analogies is to use old information (typically from the source domain) to explain new situations in the target domain. Experimental research supply evidence that structural commonalities between domains are the main driver for the construction of analogies. There exist many approaches for analogy models which apply different mechanisms to analyze and extract commonalities between two domains [5,6]. When the domains are specified formally, the theory of anti-unification can be used for a structural comparison and for representing the commonalities at a general level. Heuristic-Driven Theory Projection (HDTP) is such a symbolic analogy model using anti-unification to detect analogies between different domains.

This paper discusses anti-unification in the context of analogies and presents a spelled-out approach for computing analogies between domain theories. The remainder of the paper is structured as follows: Section 2 explains the theory of anti-unification in the context of analogies. After examining the related work (section 3), we present our approach for restricted higher-order anti-unification to compute structural commonalities (section 4). Section 5 shows how HDTP uses our approach and illustrates the results using the heat-flow example: the analogy between fluid dynamics (water flow) and thermodynamics (heat flow). Section 6 summarizes the paper and gives directions for future work.

2 Anti-Unification for Analogies

Anti-unification, originally introduced in the context of induction [7], is a formal counterpart of unification. While a unifier for a given set of terms T is a term u that is an instance[1] of every term of T, an anti-unifier (or generalization) is a term g that is an anti-instance of every term of T. While unification aims to find the most general unifier (mgu), anti-unification searches for the most specific anti-unifier, normally referred to as least general generalization (lgg).

$$\begin{tikzcd} & X \arrow[dl] \arrow[dr] & \\ a & & b \end{tikzcd}$$
(a)

$$\begin{tikzcd} & f(X,c) \arrow[dl] \arrow[dr] & \\ f(a,c) & & f(b,c) \end{tikzcd}$$
(b)

$$\begin{tikzcd} & Y \arrow[dl] \arrow[dr] & \\ f(a,b) & & g(a,b) \end{tikzcd}$$
(c)

$$\begin{tikzcd} & F(a,b) \arrow[dl] \arrow[dr] & \\ f(a,b) & & g(a,b) \end{tikzcd}$$
(d)

$$\begin{tikzcd} & F(h(X,Y)) \arrow[dl] \arrow[dr] & \\ f(h(x,h(a,b))) & & g(h(a,b)) \end{tikzcd}$$
(e)

Fig. 1. Anti-Unification of terms

Fig. 1 gives several examples for anti-unification. Terms are generalized to an anti-instance where differing constants are replaced by a variable. The terms in (c) and (d) differ w.r.t. the function symbols. While first-order anti-unification fails to detect commonalities when function symbols differ, higher-order anti-unification generalizes the function symbols to a variable F and retains the structural commonality. Example (d) is formally of higher-order, but can be reified into first-order introducing a "meta" function *eval* with $eval(f,(a,b)) = f(a,b)$. Example (e), however, affects the argument number and therefore cannot be reified: the left/right term F is substituted by f/g, $X \rightarrow x/a$ and $Y \rightarrow h(a,b)/b$. Higher-order anti-unification is necessary to anti-unify complex structures, but we loose the uniqueness of lgg and they are in general not well-defined. However, in the context of analogies, anti-instances have to meet several restrictions: generalized terms shall not be more complex than the domain terms and they shall contain elements that are present in both domains. This paper presents a solution for computing anti-instances with these restrictions for analogies. In section 4 we explain the role of anti-unification in analogy-making with HDTP.

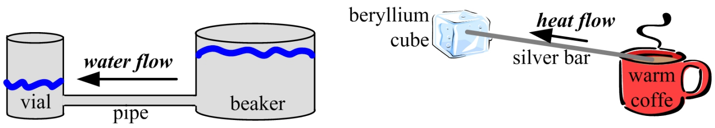


Fig. 2. Analogy between fluid dynamics (water flow) and thermodynamics (heat flow)

Heuristic-Driven Theory Projection (HDTP) is an analogy model using the theory of anti-unification to detect structural commonalities. Due to limited

[1] In this paper a term s is called an *instance* of t, and t an *anti-instance* of s if there is a substitution σ such that σ applied to t produces s. In this case we write $t \xrightarrow{\sigma} s$ or simply $t \rightarrow s$.

space we refer to [8] for the specification of the syntactic, semantic, and algorithmic properties of HDTP. Here we explain its functionality only with an example: The analogy between fluid dynamics and thermodynamics (Fig. 2). There are different possibilities of associating concepts of the target domain with concepts of the source domain. E.g. the bar and the pipe play equivalent roles as bearers of "flowing things" (water resp. heat). The task is to find plausible assignments to roles having some explanatory power. In the source domain one can observe water flowing. Although there cannot be observed anything flowing in the target domain, one can infer by analogy that there must exist something like "heat" that "flows" from the hot coffee to the beryllium cube.

Fluid dynamics (Th_S)	Thermodynamics (Th_T)
sorts $real,\ massterm, object,\ time$	**sorts** $real,\ massterm,\ object,\ time$
entities $vial: object, beaker: object, water: massterm,\ pipe: object$	**entities** $coffee: massterm,\ b\_cube: object,$ $cup: object,\ bar: object$
functions $height : object \times time \to real \times \{cm\}$ $footprint : object \times time \to real \times \{cm^2\}$ $in : object \times massterm \to object$ $vol : object \times time \to real \times \{cm^3\}$	**functions** $temp : object \times time \to real \times \{C\}$ $in : object \times massterm \to object$
facts $connected(beaker, vial, pipe)$ $\forall t_1 : time, t_2 : time :$ $footprint(beaker, t_1) > footprint(vial, t_1) \wedge$ $footprint(beaker, t_1) = footprint(beaker, t_2) \wedge$ $footprint(vial, t_1) > footprint(vial, t_2)$	**facts** $connected(in(coffee, cup), b\_cube, bar)$
laws $\forall t_1 : time, t_2 : time : t_2 > t_1 \wedge$ $height(in(water, beaker), t_1) > height(in(water, vial), t_1)$ $\to height(in(water, beaker), t_1) >$ $height(in(water, beaker), t_2) \wedge$ $vol(in(water, beaker), t_1) - vol(in(water, beaker), t_2)$ $= vol(in(water, vial), t_2) - vol(in(water, vial), t_1)$	**laws** $\forall t_1 : time,\ t_2 : time : t_2 > t_1$ $temp(in(coffee, cup), t_1) >$ $temp(b\_cube, t_1)$ $\to$ $temp(in(coffee, cup), t_2) <$ $temp(in(coffee, cup), t_1) \wedge$ $temp(b\_cube, t_2) >$ $temp(b\_cube, t_1)$

The source and target domain are specified by a set of formulas represented in many-sorted first-order logic. Given two theories Th_S and Th_T modeling source and target as input, the HDTP algorithm computes the analogy by selecting successively a formula from Th_S and Th_T (according to a heuristic) and constructing a generalization together with the corresponding substitutions.

source domain Th_S	target domain Th_T	generalization Th_G
$connected(beaker, vial, pipe)$	$connected(in(coffe, cup), b\_cube, bar)$	$connected(A, B, C)$
$height(in(water, beaker), t_1) >$ $height(in(water, vial), t_1)$	$temp(in(coffee, cup), t_1) >$ $temp(b\_cube, t_1)$	$T(A, t_1) >$ $T(B, t_1)$
$height(in(water, beaker), t_1) >$ $height(in(water, beaker), t_2)$	$temp(in(coffee, cup), t_1) >$ $temp(in(coffee, cup), t_2)$	$T(A, t_1) >$ $T(A, t_2)$
$height(in(water, vial), t_2) >$ $height(in(water, vial), t_1)$	$temp(b\_cube, t_2) >$ $temp(b\_cube, t_1)$	$T(B, t_2) >$ $T(B, t_1)$

Th_G contains four variables: A and B play the role of the container for the flowing "thing"; C connects A and B. The function symbol T stands for the function measuring the energy (height of the water in the container/temperature of the heat). The resulting substitutions describe the analogical relation:

$$A \to beaker/in(coffee, cup) \qquad B \to vial/b\_cube$$
$$C \to pipe/bar \qquad T \to \lambda x, t : height(in(water, x), t)/temp(x, t)$$

3 Related Work

Compared to unification, there is few work especially dedicated to anti-unification. However, many results on unification apparently have a counterpart in anti-unification. First of all, it should be noticed that higher-order anti-unification suffers from similar problems as higher-order unification: the *lgg* is no longer unique and without restrictions the notion is not even well-defined as demonstrated by [9]: one can construct infinite chains of ever more and more general generalizations. Therefore different strategies to restrict anti-unification have been proposed.

[10] uses higher-order patterns to restrict the possible generalizations. A λ-term (in β-normal form) is called a higher-order pattern if every free occurrence of a (higher-order) variable F appears as $F(X_1, \dots, X_n)$ with bound variables X_i all being distinct. It is shown that the substitution ordering on higher-order patterns gives a preorder with unique maximally specific generalization for any pair of terms. Thus the pattern restriction leads to well-defined generalizations. A well-known problem of higher-order patterns is overgeneralization. Given $f(a)$ and $g(a)$, the least generalization would be X instead of $F(a)$ or $F(X)$, since the latter ones are no higher-order patterns.

[9] claims that a major problem of λ-terms is the missing control of how function arguments are used in terms: they can be discarded or doubled. He therefore proposes to use combinator terms instead of λ-terms. Combinator terms are built as composition of basic functions, called combinators, that allow to control how arguments are passed from one function to the next. By restricting the valid combinators to so called relevant combinators, it is shown that a useful notion of generalization can be developed and an algorithm to compute such generalizations is presented. This formalism is also used by [11] in the context of analogical programming.

4 Restricted Higher-Order Anti-Unification

In the context of analogies we aim at computing generalizations that preserve as much of the structure of both domain terms as possible. However, the generalization must not be structurally more complex than the original terms. In this section we define a formal framework that is guided by this requirements.

4.1 Extended Substitutions

We will extend classical first-order terms by introducing variables that can take arguments: for every natural number n we assume an infinite set $\mathcal{V}_n$ of variables with arity n and a finite sets of n-ary function symbols $\mathcal{C}_n$.[2] Here we explicitly allow the case $n = 0$ with $\mathcal{V}_0$ being the set of first-order variables and $\mathcal{C}_0$ being 0-ary function symbols (constants). Variables will be written as uppercase letters

[2] One could, of course, use a more elaborated system for typing, but for the sake of simplicity we will limit to arity here, as this suffices to support the arguments of the paper.

while function symbols are lowercase. A *term* is either a first-order or a higher-order term, i.e. an expression of the form $F(t_1, \ldots, t_n)$ with $F \in \mathcal{V}_n$ and terms $t_1, \ldots, t_n$. In this setting we will redefine the notion of substitution.

Definition 1 (Basic Substitutions). *We define the following set of basic substitutions:*

1. *A* renaming $\rho^{F,F'}$ *replaces a variable* $F \in \mathcal{V}_n$ *by another variable* $F' \in \mathcal{V}_n$ *of the same arity:*
$$F(t_1, \ldots, t_n) \xrightarrow{\rho^{F,F'}} F'(t_1, \ldots, t_n).$$
2. *A* fixation ϕ_c^V *replaces a variable* $F \in \mathcal{V}_n$ *by a function symbol* $f \in \mathcal{C}_n$ *of the same arity:*
$$F(t_1, \ldots, t_n) \xrightarrow{\phi_f^F} f(t_1, \ldots, t_n).$$
3. *An* argument insertion $\iota_{V,i}^{F,F'}$ *with* $0 \leq i \leq n$, $F \in \mathcal{V}_n$, $V \in \mathcal{V}_k$ *with* $k \leq n-i$, *and* $F' \in \mathcal{V}_{n-k+1}$ *is defined by*
$$F(t_1, \ldots, t_n) \xrightarrow{\iota_{V,i}^{F,F'}} F'(t_1, \ldots, t_{i-1}, V(t_i, \ldots, t_{i+k-1}), t_{i+k}, \ldots, t_n).$$
4. *A* permutation $\pi_\alpha^{F,F'}$ *with* $F, F' \in \mathcal{V}_n$ *and* $\alpha : \{1, \ldots, n\} \to \{1, \ldots, n\}$ *bijective, rearanges the arguments of a term:*
$$F(t_1, \ldots, t_n) \xrightarrow{\pi_\alpha^{F,F'}} F'(t_{\alpha(1)}, \ldots, t_{\alpha(n)}).$$

Argument fixation can be used to replace a variable by a symbol of the same arity, e.g. $f(X) \xrightarrow{\phi_a^X} f(a)$ with $X \in \mathcal{V}_0$ and $F(a) \xrightarrow{\phi_f^F} f(a)$ with $F \in \mathcal{V}_1$. Argument insertion is a bit more complicated: inserting a 0-ary variable V increases the arity of the embedding term by 1, e.g. $F(a) \xrightarrow{\iota_{V,1}^{F,F'}} F'(a, V)$. But inserting a variable $G \in \mathcal{V}_n$ with $n \geq 2$ reduces the arity: $F(a, b, c, d) \xrightarrow{\iota_{G,1}^{F,F'}} F'(a, G(b, c), d)$. Notice that we allow argument insertion to be applied to first-order (i.e. 0-ary) variables, e.g. $X \xrightarrow{\iota_{c,0}^{X,X'}} X'(c)$.

In what follows a *substitution* is always meant to be a composition of basic substitutions. We will write $s \to t$ if there exists a substitution that transforms s into t, and $s \xrightarrow{\sigma} t$ if σ is such a substitution. Notice that every first-order substitution is indeed a substitution, since it can be described as a composition of renaming, argument insertion and fixation, e.g.

$$X \xrightarrow{\iota_{V,0}^{X,X'}} X'(V) \xrightarrow{\phi_a^V} X'(a) \xrightarrow{\phi_f^{X'}} f(a)$$

Hence this framework is a real extension of the first-order case. We will now show that it is still sufficiently restricted to be of practical use. To make this more precise, we define a measure for the structural complexity of terms. This is achieved by counting the number of symbols the term is composed of.

Definition 2 (Information Load). *A term t is assigned the* information load $\mathfrak{il}(t)$ *recursively defined as follows:*

1. $\mathfrak{il}(F(t_1,\ldots,t_n)) = n + \mathfrak{il}(t_1) + \ldots + \mathfrak{il}(t_n)$ *for a n-ary variable $F \in \mathcal{V}_n$ and terms $t_1,\ldots,t_n$.*
2. $\mathfrak{il}(f(t_1,\ldots,t_n)) = 1 + n + \mathfrak{il}(t_1) + \ldots + \mathfrak{il}(t_n)$ *for a function symbol $f \in \mathcal{C}_n$ and terms $t_1,\ldots,t_n$.*

Here we also allow the case $n=0$, i.e. 0-ary variables and constants, e. g. $\mathfrak{il}(X)=0$ for $X\in\mathcal{V}_0$, $\mathfrak{il}(c)=1$ for $c\in\mathcal{C}_0$, and $\mathfrak{il}(f(c,X))=4$ with $f\in\mathcal{C}_2$. Therefore applying substitutions will never reduce the information load of a term:

Lemma 1. *Let s and t be terms. If $s \to t$, then $\mathfrak{il}(s) \leq \mathfrak{il}(t)$.*

Proof. This can be proven by inductive decomposition of the substitution: renaming and permutation do not change the information load. Fixation increases the information load by 1 and so does argument insertion: applying $\iota_{G,i}$ to $F(t_n,\ldots,t_n)$ with $G \in \mathcal{V}_k$ leads to $F'(t_1,\ldots,t_{i-1},G(t_i,\ldots,t_{i+k-1}),t_{i+k},\ldots,t_n)$ with information load $n-k+1+\sum_{j=1}^{i-1}\mathfrak{il}(t_j)+\mathfrak{il}(G(t_i,\ldots,t_{i+k}))+\sum_{j=i+k}^{n}\mathfrak{il}(t_j)$ and $\mathfrak{il}(G(t_i,\ldots,t_{i+k}))=k+\sum_{j=i}^{i+k-1}\mathfrak{il}(t_j)$, summing up to $n+1+\sum_{j=1}^{n}\mathfrak{il}(t_j)$. q.e.d.

As the information load of terms is always finite and there are only finitely many permutations of arguments for a given arity, it follows:

Corollary 1. *For a given term t there are only finitely many (up to renaming) anti-instances (i.e. terms s with $s \to t$).*

4.2 Preferred Generalizations

Based on our extended notions of terms and substitutions, generalization can be defined as usual:

Definition 3 (Generalization). *A generalization for a pair of terms $\langle s,t\rangle$ is a triple $\langle g,\sigma,\tau\rangle$ with a term g and substitutions σ,τ such that $s \xleftarrow{\sigma} g \xrightarrow{\tau} t$.*

Since for every term t there is a substitution $X \to t$, generalizations are guaranteed to exist. As a direct consequence of lemma 1 we get:

Proposition 1. *A pair of terms $\langle s,t\rangle$ has only finitely many (up to renaming) generalizations. For every generalization $\langle g,\sigma,\tau\rangle$ it holds $\mathfrak{il}(g) \leq \mathfrak{il}(s)$ and $\mathfrak{il}(g) \leq \mathfrak{il}(t)$.*

This means that least general generalizations are well-defined in this setting. However, they do not have to be unique:

Having multiple possible generalizations is not necessarily bad, especially in the context of analogies, where normally several mappings with different degree of plausibility may coexist. Nevertheless, it would be useful to have a criterion to rank the alternatives. A plausible approach could be based on the information load of the generalization. However, since we are aiming at anti-unifying sets of terms and want to promote the reuse of substitutions, we will introduce a ordering that is based on their complexity:

$$\begin{tikzcd}
& F(d, G(a)) \arrow[dl] \arrow[dr] & & & f(X, Y) \arrow[dl] \arrow[dr] & \\
f(g(a, b, c), d) & & f(d, \phi(a)) & f(g(a, b, c), d) & & f(d, \phi(a))
\end{tikzcd}$$

Fig. 3. Example with multiple least general generalizations

Definition 4 (Complexity of Substitution). *The* complexity *of a basic substitution σ is defined as*

$$\mathcal{C}(\sigma) = \begin{cases} 0 & \textit{if } \sigma = \rho \\ 1 & \textit{if } \sigma = \phi_c \\ k+1 & \textit{if } \sigma = \iota_{V,i} \textit{ and } V \in \mathcal{V}_k \\ 1 & \textit{if } \sigma = \pi_\alpha \end{cases}$$

For a composition of basic substitutions we define $\mathcal{C}(\sigma_1 \cdots \sigma_m) = \sum \mathcal{C}(\sigma_i)$ and for an arbitrary substitution σ is $\mathcal{C}(\sigma) = \min\{\mathcal{C}(\sigma_1 \cdots \sigma_m) \mid \sigma_1 \cdots \sigma_m = \sigma\}$.[3]

The complexity of a substitution is meant to reflect its processing effort. Therefore permutations have a non-zero complexity even though they do not change the information load of a term. The argument insertion restructures the term, and the higher the arity of the inserted variable, the more arguments are moved and therefore the more complexity is assigned to that operation.[4]

Based on complexity of substitutions, complexity of generalizations can be defined straight forward:

Definition 5 (Complexity of Generalization). *Let $\langle g, \sigma, \tau \rangle$ be a generalization for a pair of terms $\langle s, t \rangle$. Define the complexity of the generalization by $\mathcal{C}(\langle g, \sigma, \tau \rangle) = \mathcal{C}(\sigma) + \mathcal{C}(\tau)$.*

With this complexity measure we can select *preferred generalization* by minimizing their complexity. Obviously, preferred generalizations are always least general, while the contrary is not always the case as demonstrated by Fig. 3.

4.3 Computing Preferred Generalizations

A simple algorithm to compute preferred generalizations is given in Fig. 4. It uses a bottom-up breadth-first strategy. A priority queue of anti-instances is initialized with the *left* and *right* terms that shall be anti-unified. Now the first element of that queue is taken. If there is a compatible anti-instance for that term

[3] The minimum construction is needed, as there exist multiple decompositions of σ with different complexity. With a bit more effort one can define a normal decomposition which can be shown to have minimal complexity. This is left out in this paper due to space limitation.

[4] The complexity values for basic substitutions have proven to be useful in practice. Here, the analysis of different values is subject of future work.

```
function $anti\_unify(left\_term, right\_term)$ returns $generalization$
  variables:
    $Open$: priority queue of $anti\_instance$
    $Closed$: set of $anti\_instance$
    // $anti\_instance$ is a quadruple $\langle$complexity,substitution,generalized_term,index$\rangle$
    // with index $\in \{$“left”, “right”$\}$ and $op($“left”$) =$ “right”,$op($“right”$) =$ “left”
  initialize:
    $Open = \{\langle 0, id, left\_term, \text{“left”}\rangle, \langle 0, id, right\_term, \text{“right”}\rangle\}$
    $Closed = \{\}$
  while $Open \neq \{\}$ do
    $\langle c, \sigma, g, i\rangle = \texttt{first}(Open)$; $Open = \texttt{rest}(Open)$
    if $\langle \_, \theta, g, op(i)\rangle \in Open \cup Closed$ return $\langle g, \sigma, \theta\rangle$
    $G = \{\langle c', \sigma', g', i\rangle \mid \exists \tau \in basic\_substitutions : g' \xrightarrow{\tau} g, c' = c + \mathcal{C}(\tau), \sigma' = \sigma \circ \tau\}$
    $Open = \texttt{merge}(Open, G)$
    $Closed = Closed \cup \{\langle c, \sigma, t, i\rangle\}$
  end while
end function
```

Fig. 4. Algorithm for computing preferred generalizations

from the other domain, a preferred generalization has been found. Otherwise all anti-instances are computed that can be reached by a single (inverse) application of a basic substitution. These anti-instances are annotated with their complexity and merged back into queue and the whole process starts again.

The anti-instances produced have a decreasing information load and therefore the algorithm is guaranteed to terminate. However, the preferred generalization that is computed may depend on the complexity measure $\mathcal{C}$ that is chosen.

5 Example

This section explains our approach step-by-step with the heat flow analogy. Fig. 5 shows the anti-unification of the terms $connects(beaker, vial, pipe)$ from Th_S and $connected(in(coffee, cup), b\_cube, bar)$ from Th_T with required substitutions. The remaining generalizations are analogous to the anti-unification of $height(in(water, beaker), t_1) > height(in(water, vial), t_1)$ and $temp(in(coffee, cup), t_1) > temp(b\_cube, t_1)$.

Three basic substitutions are needed to transform $connected(A, B, C)$ to the domain-specific term in the source: the fixation ϕ^{A}_{beaker} substitutes the variable A to the constant $beaker$ (the same for B and C). The substitution the target side is more complex, because A maps on the complex structure $in(coffee, cup)$. $\iota^{A,A'}_{X,0}$ replaces variable A by the complex structure $A'(X)$ and $\iota^{A',A''}_{Y,0}$ inserts the second variable Y. Afterwards the variables are fixated to the required constants. The substitutions have the complexity $\mathcal{C} = 3$ on the source and $\mathcal{C} = 7$ on the target side. To transform $T(A, t_1) > T(B, t_2)$ in domain specific terms, 7 basic substitutions are required on the source and 7 on the target side. However, many

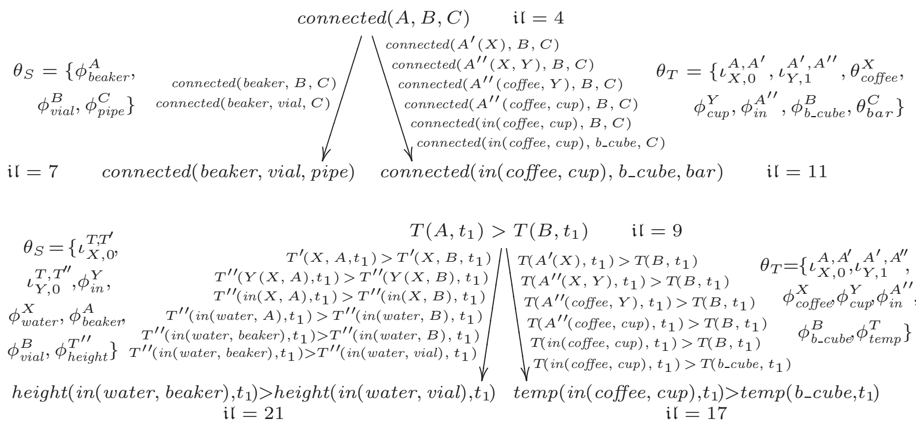

connected(A, B, C) il = 4
θ_S = {φ^A_beaker, φ^B_vial, φ^C_pipe}
connected(beaker, B, C)
connected(beaker, vial, C)
connected(A′(X), B, C)
connected(A″(X, Y), B, C)
connected(A″(coffee, Y), B, C)
connected(A″(coffee, cup), B, C)
connected(in(coffee, cup), B, C)
connected(in(coffee, cup), b_cube, C)
θ_T = {ι^{A,A′}_{X,0}, ι^{A′,A″}_{Y,1}, θ^X_coffee, φ^Y_cup, φ^{A″}_in, φ^B_b_cube, θ^C_bar}
il = 7 connected(beaker, vial, pipe)
connected(in(coffee, cup), b_cube, bar) il = 11
T(A, t1) > T(B, t1) il = 9
θ_S = {ι^{T,T′}_{X,0}, ι^{T,T″}_{Y,0}, φ^Y_in, φ^X_water, φ^A_beaker, φ^B_vial, φ^{T″}_height}
T′(X, A, t1) > T′(X, B, t1)
T″(Y(X, A), t1) > T″(Y(X, B), t1)
T″(in(X, A), t1) > T″(in(X, B), t1)
T″(in(water, A), t1) > T″(in(water, B), t1)
T″(in(water, beaker), t1) > T″(in(water, B), t1)
T″(in(water, beaker), t1) > T″(in(water, vial), t1)
T(A′(X), t1) > T(B, t1)
T(A″(X, Y), t1) > T(B, t1)
T(A″(coffee, Y), t1) > T(B, t1)
T(A″(coffee, cup), t1) > T(B, t1)
T(in(coffee, cup), t1) > T(B, t1)
T(in(coffee, cup), t1) > T(b_cube, t1)
θ_T = {ι^{A,A′}_{X,0}, ι^{A′,A″}_{Y,1}, φ^X_coffee, φ^Y_cup, φ^{A″}_in, φ^B_b_cube, φ^T_temp}
height(in(water, beaker), t1) > height(in(water, vial), t1) il = 21
temp(in(coffee, cup), t1) > temp(b_cube, t1) il = 17

Fig. 5. Computing the generalizations for the heat flow analogy

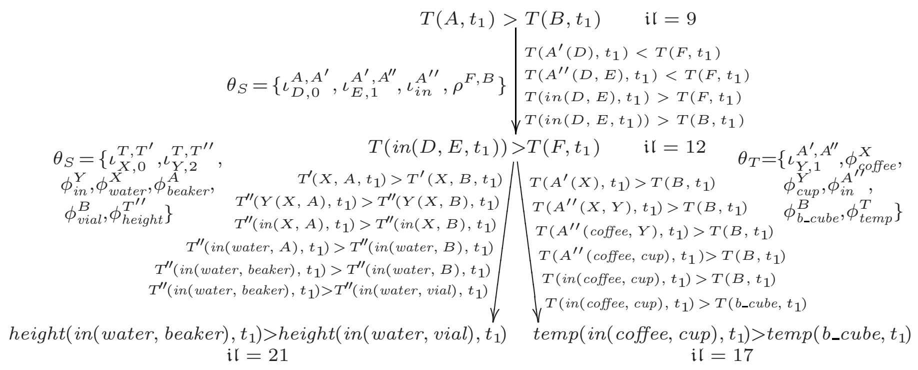

T(A, t1) > T(B, t1) il = 9
θ_S = {ι^{A,A′}_{D,0}, ι^{A′,A″}_{E,1}, ι^{A″}_in, ρ^{F,B}}
T(A′(D), t1) < T(F, t1)
T(A″(D, E), t1) < T(F, t1)
T(in(D, E), t1) > T(F, t1)
T(in(D, E, t1)) > T(B, t1)
T(in(D, E, t1)) > T(F, t1) il = 12
θ_S = {ι^{T,T′}_{X,0}, ι^{T,T″}_{Y,2}, φ^Y_in, φ^X_water, φ^A_beaker, φ^B_vial, φ^{T″}_height}
T′(X, A, t1) > T′(X, B, t1)
T″(Y(X, A), t1) > T″(Y(X, B), t1)
T″(in(X, A), t1) > T″(in(X, B), t1)
T″(in(water, A), t1) > T″(in(water, B), t1)
T″(in(water, beaker), t1) > T″(in(water, B), t1)
T″(in(water, beaker), t1) > T″(in(water, vial), t1)
T(A′(X), t1) > T(B, t1)
T(A″(X, Y), t1) > T(B, t1)
T(A″(coffee, Y), t1) > T(B, t1)
T(A″(coffee, cup), t1) > T(B, t1)
T(in(coffee, cup), t1) > T(B, t1)
T(in(coffee, cup), t1) > T(b_cube, t1)
θ_T = {ι^{A′,A″}_{Y,1}, φ^X_coffee, φ^Y_cup, φ^{A″}_in, φ^B_b_cube, φ^T_temp}
height(in(water, beaker), t1) > height(in(water, vial), t1) il = 21
temp(in(coffee, cup), t1) > temp(b_cube, t1) il = 17

Fig. 6. Different generalizations with different information load and complexity of substitutions

substitutions are already available from the anti-unification of *connects*: only 5 new substitutions are required for the source and 1 new for the target.

The generalized term $T(A, t_1) > T(B, t_2)$ has the information load $\mathfrak{il} = 9$ and is actually not the *lgg* of the domain specific terms. Fig. 6 shows that the generalization $T(in(D, E), t_1) > T(F, t_1)$ with the $\mathfrak{il} = 12$ is more specific and an anti-instance of the domain-specific terms. However, none of the analogical mappings received by the anti-unification of *connects* is reused and therefore also none of the substitutions. The complexity of the required substitutions on the source side is $\mathcal{C} = 8$ and on the target side $\mathcal{C} = 4$. Since HDTP aims to anti-unify whole theories and reduce the complexity of substitutions across the whole domain-specific substitution, the solution presented in Fig. 5 is preferred.

6 Conclusion and Future Work

We have presented a restricted version of higher-order anti-unification and have proven that in this framework least general generalizations can be defined and are indeed more specific than in the first-order case. We proposed a measure for the complexity of substitutions and presented an algorithm that allows for computing preferred generalizations with respect to this complexity. We further showed that complexity can be useful in the context of analogy making, when not only pairs of terms, but sets of terms shall be anti-unified, as it can be a means to encourage the reuse of substitutions.

The complexity measure for substitutions has been successfully applied in practice. However, alternative definitions are possible and the impact of different modification is the subject of current investigations. The application of complexity to anti-unification of sets of formulas as sketched in section 5 will be further examined and an algorithmic framework will be developed.

References

1. Chalmers, D.J., French, R.M., Hofstadter, D.R.: High-level perception, representation, and analogy: A critique of artificial intelligence methodology. Journal of Experimental and Theoretical Artificial Intelligence 4(3), 185–211 (1992)
2. Forbus, K.D., Gentner, D., Markman, A.B., Ferguson, R.W.: Analogy just looks like high-level perception: why a domain-general approach to analogical mapping is right. Journal of Experimental & Theoretical Artificial Intelligence 10, 231–257 (1998)
3. Holyoak, K., Morrison, R. (eds.): The cambridge handbook on thinking and reasoning. Cambridge University Press, Cambridge (2005)
4. Oppenheimer: Analogy in science. American Psychologist 11, 127–135 (1956)
5. Gentner, D.: The mechanism of analogical learning. In: Vosniadou, S., Ortony, A. (eds.) Similarity and analogical reasoning, pp. 199–241. Cambridge University Press, Cambridge (1989)
6. Hofstadter, D.R., Mitchell, J.: The copycat project: A model of mental fluidity and analogy-making. In: Hofstadter, D.R., group, F.A.R. (eds.) Fluid Concepts and Creative Analogies. Basic Books, pp. 205–267 (1995)
7. Plotkin, G.D.: A note on inductive gneralization. Machine Intelligence 5, 153–163 (1970)
8. Gust, H., Kühnberger, K.U., Schmid, U.: Metaphors and heuristic-driven theory projection (hdtp). Theoretical Computer Science 354(1), 98–117 (2006)
9. Hasker, R.W.: The replay of program derivations. PhD thesis, Champaign, IL, USA (1995)
10. Pfenning, F.: Unification and anti-unification in the Calculus of Constructions. In: Sixth Annual IEEE Symposium on Logic in Computer Science, Amsterdam, The Netherlands, pp. 74–85 (1991)
11. Wagner, U.: Combinatorically restricted higher order anti-unification. An application to programming by analogy. Master's thesis, Technische Universität Berlin (2002)

A Template Matching Table for Speeding-Up Game-Tree Searches for Hex

Rune Rasmussen, Frederic Maire, and Ross Hayward

Faculty of Information Technology,
Queensland University of Technology,
Gardens Point Campus,
GPO Box 2434,
Brisbane QLD 4001,
Australia
r.rasmussen@qut.edu.au, f.maire@qut.edu.au,
r.hayward@qut.edu.au

Abstract. Transposition tables have long been a viable tool in the pruning mechanisms of game-tree search algorithms. In such applications, a transposition table can reduce a game-tree to a game-graph with unique board positions at the nodes. This paper proposes a transposition table extension, called a *template matching* table, where templates that prove winning positions are used to map features of board positions to board values. This paper demonstrates that a game-tree search for the game of Hex can have a more effective pruning mechanism using a template matching table than it does using a transposition table.

1 Introduction

The family of *combinatorial games* consists of those two player games that do not involve chance, that do not hide information from the players, where the players take alternate turns and the outcome at the end of a game is either a win for one of the players or a draw [1]. The complete information of a combinatorial game is given by a structure known as a *game-tree*, which is a rooted tree whose nodes are the valid board positions and whose edges are the legal moves [2].

To solve a combinatorial game, one can employ a search procedure known as a *minimax* search. In short, a minimax search is an exhaustive search of a game-tree that returns a value called a *game theoretic* value. The game theoretic value represents the outcome if both players play perfectly. Minimax search times can be improved if the search employs an alpha-beta pruning algorithm. Such a search is known as an *alpha-beta search* [3]. The alpha-beta search is able to use the game theoretic value to prune subtrees from the game-tree.

The alpha-beta pruning algorithm is not the only kind of pruning available. A board position in a game tree maybe reached via more than one path. [4]. A *game-graph* is the reduction of a game tree such that every board position is unique. The multiple paths that can lead from the root to a particular board position is called a set of *transpositions* [5]. A minimax search procedure can use transposition sets to prune a game-tree, because the board positions at the end of any given set of transpositions have the same game theoretic value. A search procedure can store board positions and their respective game theoretic values in a cache called a *transposition table* and can access those game theoretic values from the table at reoccurring board positions, so as to prune the search. That is, a transposition table can reduce a game tree search to a game graph search.

In this paper, we investigate the game-tree of a combinatorial game called Hex. Two papers have prompted this investigation. Ashelevich's pioneering work in [6], on a system called *H-Search* that automatically deduces sub-games for the game of Hex and a search algorithm called *Pattern Search* by van Rijswijck in [7] that harnesses sub-game analysis to efficiently solve small Hex boards. From their work we can infer that the game-graph for Hex is not minimum and is reducible to a more compact graph. In this paper, we show that a sub-game representation, called a *template*, can be used to map Hex board positions to game theoretic values and this mapping implies that the game-graph for Hex is indeed reducible. We present a variation on a transposition table, called a *template matching* table, that uses templates in the same way that a transposition table uses transpositions. Finally, we compare a template matching table against a standard transposition table and show that the template matching table is a more effective pruning procedure for searches that solve small Hex boards.

Section 2, gives an overview of the game of Hex and describes various concepts of a sub-game in Hex. Section 3, introduces multi-target sub-games and gives a definition for templates. The template matching table is defined in Section 4. In Section 5, a comparison is made between our template matching table and a standard transposition table in solving a set of small Hex boards. A conclusion and further work is discussed in Section 6.

2 The Game of Hex

The game of Hex is a two-player game played on a tessellation of hexagonal cells that cover a rhombic board (see Figure 1)[8]. Each player has a cache of coloured stones. The goal for the player with the black stones is to connect the black sides of the board. Similarly, the goal for the player with the white stones is to connect the white sides of the board. The initial board position is empty. Players take turns and place a single stone, from their respective cache, on an empty cell. The first player to connect their sides of the board with an unbroken chain of their stones is the winner. The game of Hex never ends in a draw [8].

If a player forms an unbroken chain of stones, not necessarily between that player's sides, then any two stones in that chain are said to be *connected*. A *group* is a maximal connected component of stones [9,6]. Figure 1 shows seven groups where three of the seven groups have the labels 'a', 'b' and 'c', respectively. The four sides of the board also constitute four distinct groups. A player wins a game, when the opposite sides for that player connect.

For the analysis of board positions, Anshelevich introduced the concept of a *sub-game* [9,6]. In a sub-game, the players are called *Cut* and *Connect*. Both *Cut*

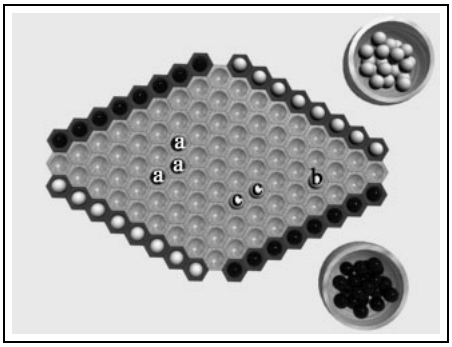


Fig. 1. A 9x9 Hex board

and *Connect* play on a subset of the empty cells between two disjoint targets, where a *target* is either an empty cell or one of *Connect's* groups. The player's roles are not symmetric, as *Connect* moves to form a chain of stones connecting the two targets, while *Cut* moves to prevent *Connect* from forming any such chain of stones. In this paper, we generalize sub-games to also include a subset of *Connect's* stones.

Definition 1 (sub-game). *A sub-game is a four-tuple* (x, S, C, y) *where* x *and* y *are targets. The set* S*, is a set of cells with Connect's stones and the set* C *is a set of empty cells. Finally,* x*,* y*,* S *and* C *are all disjoint.*

The set S is called the *support* and the set C is called the *carrier*. The sub-game just defined is equivalent to the sub-game Anshelevich defines in [9,6] when S is empty. A sub-game is a *virtual connection* if *Connect* can win this sub-game against a perfect *Cut* player. A virtual connection is *weak* if *Connect* must play first to win the sub-game. A virtual connection is *strong* if *Connect* can play second and still win the sub-game.

3 Multi-target Sub-Games and Templates

The sub-game in Definition 1, is a game that involves two distinct targets. However, more complex sub-games can occur on arbitrary board positions, where player *Connect* has the option of connecting one of many target pairs. A *multi-target sub-game* has two disjoint sets of cells X and Y, called *target sets*. The players play on a subset of the empty cells, where the aim for *Connect* is to form a chain of stones connecting at least one cell in X to at least one cell in Y, while *Cut* moves to prevent *Connect* from forming any such chain of stones.

Definition 2 (multi-target sub-game). *A multi-target sub-game is a four-tuple* (X, S, C, Y) *where* X *and* Y *are target sets. The set* S*, is a set of cells with Connect's stones and the set* C *is a set of empty cells. Finally,* X*,* Y*,* S *and* C *are all disjoint.*

A multi-target sub-game is a *virtual connection*, if *Connect* can win this sub-game against a perfect *Cut* player. In addition, the terms *weak* and *strong* apply. A multi-target sub-game that is a virtual connection is called a *template*.

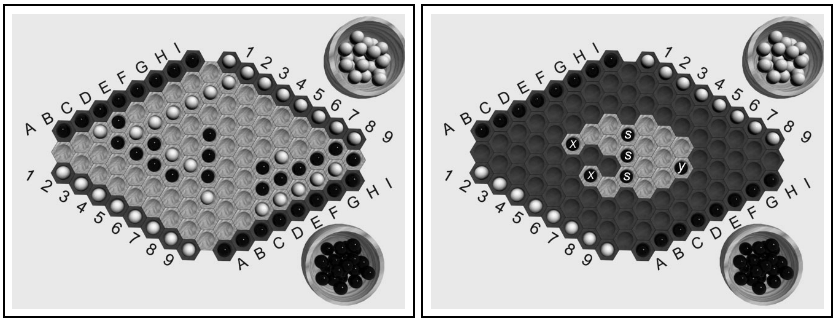


Fig. 2. Left: An arbitrary board position b. Right: A template that proves b is winning for Black.

A template can be used to prove a board position is winning for one of the players. For example the left of Figure 2, shows an arbitrary board position b. For board position b, let B_X be the set of stones that form Black's top-left side group and let B_Y be the set of stones that form Black's bottom-right side group. In addition, let B_S be the remaining set of Black stones and let B_C be the set of empty cells. The right of Figure 2 shows a template (X, S, C, Y), where Black is *Connect*, $X = \{C5, D3\}$, $Y = \{F7\}$, $S = \{D6, E5, F4\}$ and the carrier C is represented by the set of empty cells with a lighter shade. This template *proves* b is winning for Black because, $X \subseteq B_X$, $Y \subseteq B_Y$, $S \subseteq B_S$ and $C \subseteq B_C$, which means that every chain of stones that Black can form to win this template will also connect Black's sides on b.

Proposition 1 (Template Matching). *Let (X, S, C, Y) be a template where player P is Connect. Given a board position b, let B_X be the set of stones in one side group for player P, B_Y be the set of stones in the other side group for player P, B_S be the set of P stones not in B_X or B_Y and B_C be the set of empty cells. If $(X \subseteq B_X) \wedge (Y \subseteq B_Y) \wedge (S \subseteq B_S) \wedge (C \subseteq B_C)$, template (X, S, C, Y) proves b is winning for player P.*

4 A Template Matching Table

A transposition table is a mapping $T : b \rightarrow v$, from a set of board positions b to a set of game theoretic values v. In practice, a game-tree search uses a transposition table as a cache of board values that trigger a pruning mechanism. For example in Figure 3, a minimax search attempts to solve the Hex board position b_0. Game-theoretic values returned by this search are (-1) if a board position has a Black winning strategy and (+1) if a board position has a White winning strategy. In the left of Figure 3, the search traverses to board position b_i and proceeds to search the subtree rooted at b_i. After searching this subtree, the search returns (-1) for b_i and the transposition table T is updated with the mapping $(b_i, -1)$. In the right of Figure 3, the search traverses a different path to board position b_i. Since the transposition table T has a mapping for b_i, the search uses the value $T(b_i)$ instead of searching the subtree rooted at this b_i. The transposition table is used to prune the search, because the search can immediately backtrack from position b_i with value $T(b_i)$.

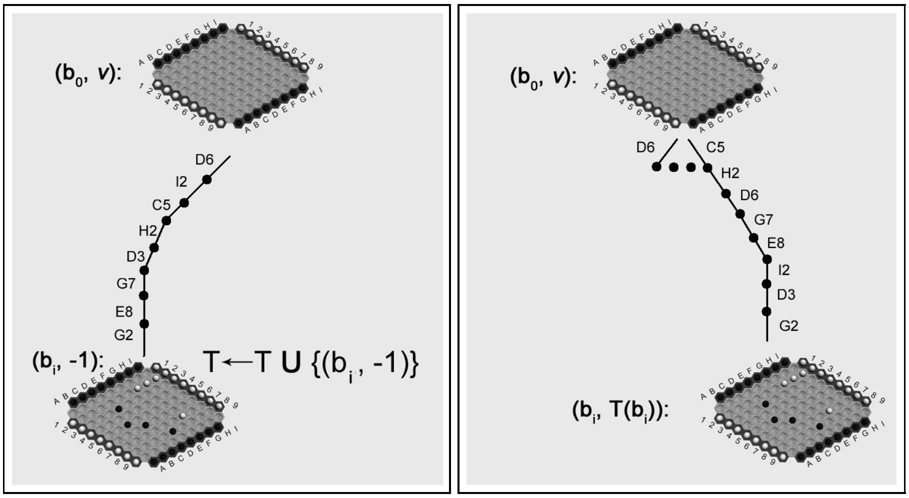


Fig. 3. Left: After searching the subtree under board positions b_i, the minimax search finds that Black has a winning strategy for b_i. Since the game theoretic value for a Black winning strategy is (-1), the search adds the mapping $(b_i, -1)$ to the transposition table T. Right: Following a different path, the same minimax search again arrives at board position b_i. The game theoretic value for b_i can be returned by the transposition table, in place of a search in the subtree of b_i.

For the game of Hex, a *template matching* table is a mapping $T_T : t \rightarrow v$, from a set of templates t to a set of game theoretic values v. In a game-tree search where a procedure is employed to generate templates, the application of a template matching table is similar to the application of a transposition table[1]. In Figure 4, a game-tree search that employs a template generating algorithm attempts to solve the Hex board position b_0. The game-theoretic values are (-1)

[1] For the game of Hex, there are a number of algorithms that can be modified to generate templates. The *H-Search* algorithm defined in [9,6], can generate virtual connections for end-of-game board positions. The *Pattern Search* defined in [7] and extended to a Hex solving algorithm in [10], is a game-tree search that solves a virtual connection at each search node. The *Connection Utility* search defined in [11] is an extension of the Pattern Search with a strong move generating algorithm. This search will provide the base search for experiments in this paper.

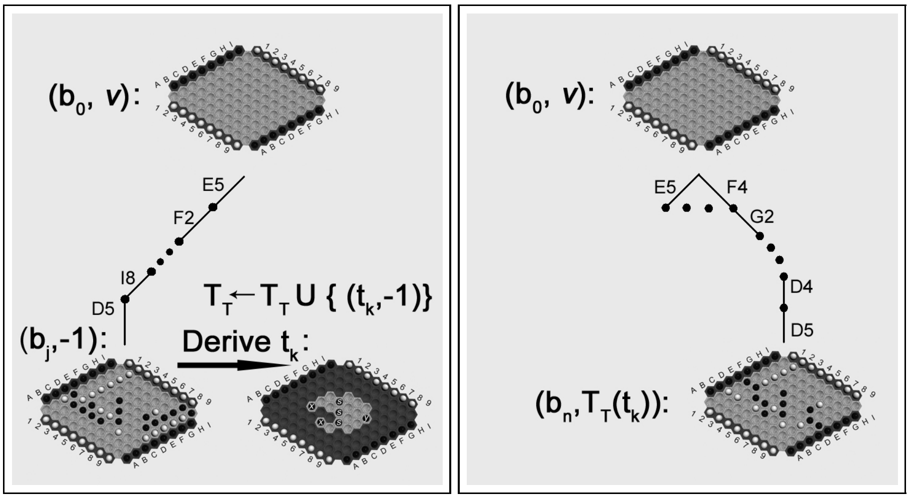


Fig. 4. Left: A game-tree search finds that Black has a winning strategy for b_i represented by template t_k. The search adds the mapping $(t_k, -1)$ to the template matching table T_T. Right: On a different path, the same search arrives at board position b_n. An application of Proposition 1 shows that template t_k matches b_n. The game theoretic value for b_n can be returned by the template matching table, in place of a search in the subtree of b_n.

and (+1) for Black and White's win, respectively. In the left of Figure 4, the search traverses to board position b_j where a template generating search derives template t_k. Since the Black player is the *Connect* player in t_k, the template matching table T_T is updated with the mapping $(t_k, -1)$. In the right of Figure 4, the search traverses a different path to board position b_n. An application of Proposition 1 finds template t_k is a match for position b_n. The search uses the value $T_T(t_k)$ instead of searching the subtree rooted at b_n.

5 Experiments with the Template Matching Table

The use of a transposition table can prune repeating board positions from a game-tree search. In this paper, we show that a template matching table can be used to prune a game-tree search more effectively for the game of Hex. In this experiment, a game-tree search called *Connection Utility* search, which was defined in [11], will be the base search algorithm. One version of this search will make use of a transposition table and a second version will make use of a template matching table. In the appendix, Algorithm 1 shows pseudo code for the Connection Utility search. Code fragments 1, 2 and 3 can be added to Algorithm 1 to define a transposition driven search. Alternatively, code fragments 4, 5 and 6 can be added to Algorithm 1 to define a search with template matching. These searches deal with virtual connections. The game-theoretic values to be stored

Table 1. The number of nodes visited for each search in the order of best-to-worst

Search using	Nodes Visited 3x3	Nodes Visited 4x4
Template Matching Table	1380	198512
Transposition Table	1795	546832

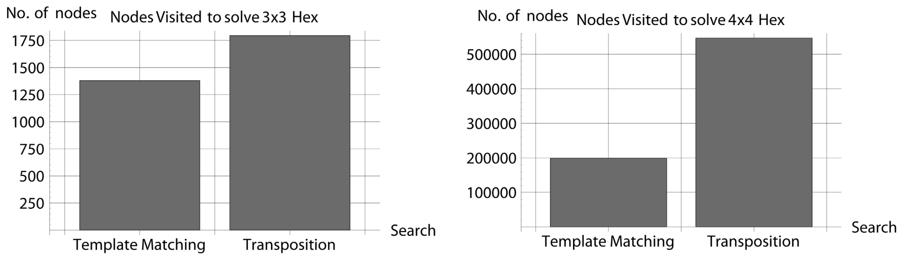


Fig. 5. A comparison of nodes visited by searches solving 3x3 and 4x4 Hex

in the transposition table are indices into a list of strong carriers. This is valid, because the list of strong carriers can be partitioned, so that one sublist of the carriers represent Black's winning positions and the other sublist of the carriers represent White's winning positions.

5.1 Results

Figure 5 and Table 1 show that the Connection Utility search can solve Hex for the 3x3 and 4x4 boards in fewer search nodes using a template matching table, than it can using a standard transposition table. Given that the search algorithms are simplified so that particular enhancements be measured, a comparative experiment beyond the 4x4 board is impractical.

6 Conclusion and Further Work

This paper shows that pruning with a template matching table is more effective than pruning with a transposition table, for the game of Hex. Similar template matching tables can be defined for other combinatorial games, where the concepts of sub-game and sub-game deduction are applicable. The search described herein can be extended with the H-Search algorithm described in [9,6], to solve the 5x5 and 6x6 Hex boards, completely. Through personal discussions with Ryan Hayward and his colleagues at the University of Alberta, concerning some finer details of move domination and the move ordering algorithms used in their Hex Solver program described in [10], we have devised several Hex solving programs that employ our template matching table and are each capable of completely solving 7x7 Hex in less than half a million nodes. These search times are a vast improvement over the search times that Hayward et al. report for their Solver program. Given these promising results, our aim in future work is to completely solve the 8x8 Hex board, a board size that has so far resisted the efforts of game researchers.

References

1. Fraenkel, A.S.: Combinatorial games: Selected bibliography with a succinct gourmet introduction. Electronic Journal of Combinatorics (1994)
2. van Rijswijck, J.: Computer Hex: Are Bees Better than Fruitflies? Thesis: Master of science, University of Alberta (2000)
3. Brudno, A.L.: Bounds and valuations for abridging the search of estimates. Problems of Cybernetics 10, 225–241 (1963)
4. Bruin, A.D., Pijls, W.: Trends in game tree search. In: Conference on Current Trends in Theory and Practice of Informatics, pp. 255–274 (1996)
5. Kishimoto, A., Müller, M.: A general solution to the graph history interaction problem. In: McGuinness, D.L., Ferguson, G. (eds.) AAAI, pp. 644–649. AAAI Press / The MIT Press (2004)
6. Anshelevich, V.V.: A hierarchical approach to computer hex. Artificial Intelligence 134, 101–120 (2002)
7. van Rijswijck, J.: Search and Evaluation in Hex. Thesis: Master of science, University of Alberta (2002)
8. Gardener, M.: The game of hex. In: The Scientific American Book of Mathematical Puzzles and Diversions, Simon and Schuster, New York (1959)
9. Anshelevich, V.V.: An automatic theorem proving approach to game programming. In: Proceedings of the Seventh National Conference of Artificial Intelligence, pp. 194–198. AAAI Press, California (2000)
10. Hayward, R., Björnsson, Y., Johanson, M., Kan, M., Po, N., Rijswijck, J.V. (eds.): Advances in Computer Games: Solving 7x7 HEX: Virtual Connections and Game-State Reduction. IFIP International Federation of Information Processing, vol. 263. Kluwer Achedemic Publishers, Boston (2003)
11. Rasmussen, R., Maire, F., Hayward, R.: A move generating algorithm for hex solvers. In: Sattar, A., Kang, B.-H. (eds.) AI 2006. LNCS (LNAI), vol. 4304, pp. 637–646. Springer, Heidelberg (2006)

Appendix

Code Fragment 1.

```
1: $T \leftarrow \emptyset$  {The transposition table}
2: $C \leftarrow \emptyset$  {The carrier list}; $i \leftarrow 0$  {The list index}
```

Code Fragment 2.

```
1: if $T(board)$ then
2:    $carrier \leftarrow C[T(board)]$  {Look in the transposition table for a board-value pair}
3:    for all $j \in carrier$ do
4:       $utilities[j] \leftarrow 1$ {Code to calculate the connection utilities}
5:    end for
6:    $modeColour \leftarrow board.PlayerOfLastMove$  {The Connect player's colour. The
      carrier was deduced on a board position where Connect made the last move.}
7:    return$(carrier, utilities, modeColour)$
8: end if
```

Code Fragment 3.

```
1: $T \leftarrow T \cup \{(board, i)\}$  {Update the transposition with a carrier index}
2: $C[i] \leftarrow carrier$  {Update the carrier list}
3: $i \leftarrow i + 1$
```

Code Fragment 4.

```
1: $T_T \leftarrow \emptyset$  {The template matching table}
2: $C \leftarrow \emptyset$  {The carrier list}; $i \leftarrow 0$  {The list index}
```

Code Fragment 5.

```
1: if $T_T(board)$ then
2:    $carrier \leftarrow C[T_T(board)]$ {Proposition 1 is applied to match a template to the
      board position and return an index.}
3:    for all $j \in carrier$ do
4:       $utilities[j] \leftarrow 1$  {Code to calculate the connection utilities}
5:    end for
6:    $modeColour \leftarrow board.PlayerOfLastMove$ {The Connect player's colour. A Cut
      move cannot trigger a match before a Connect move does.}
7:    return$(carrier, utilities, modeColour)$
8: end if
```

Code Fragment 6.

```
1: $t \leftarrow constructTemplate(carrier, board)$  {Construct template t from the carrier
   and board position.}
2: $T_T \leftarrow T_T \cup \{(t, i)\}$  {Update the template matching table.}
3: $C[i] \leftarrow carrier$  {Update the carrier list}
4: $i \leftarrow i + 1$
```

Algorithm 1. Connection_Utility_Search(*board*) **Returns** (Carrier, vector of Connection Utilities, Winner Colour)

```
1: $utilities[1 : board.size] \leftarrow 0$; $carrier \leftarrow \emptyset$
2: $modeColour \leftarrow Black$ {Start the search in Black mode}
3: >Code Fragments 1 for transposition table or 4 for template matching table.
4: if $board.isTerminal$ then
5:    $modeColour \leftarrow board.winningPlayer$
6:    for all $i \in board.winnersStones$ do
7:       $utilities[i] \leftarrow 1$
8:    end for
9:    return$(carrier, utilities, modeColour)$
10: end if
11: >Code Fragments 2 for transposition table or 5 for template matching table.
12: $moveList \leftarrow aStaticMoveGenerator(board)$
13: while $moveList \neq \emptyset$ do
14:    $m \leftarrow popFirst(moveList)$
15:    $board.playMove(m)$
16:    $(C, Util, winColour) \leftarrow Connection\_Utility\_Search(board)$
17:    {The mode colour changes if $winColour \neq modeColour$.}
18:    $modeColour \leftarrow winColour$
19:    $board.undoMove(m)$
20:
21:    if $modeColour = board.turn$ then
22:       {$m$ was a winning move.}
23:       $carrier \leftarrow \{m\} \cup C$ {$carrier$, is a weak threat pattern carrier.}
24:       return$(carrier, Util, modeColour)$
25:    else
26:       {$m$ was not winning, try a remaining move.}
27:       $carrier \leftarrow carrier \cup C$ {OR deduction rule}
28:       $utilities \leftarrow utilities + Util$ {vector operation}
29:       $moveList \leftarrow moveList \cap C$ {the must-play region}
30:       $moveList \leftarrow SortDescending(moveList, utilities)$
31:    end if
32: end while
33: >Code Fragments 3 for transposition table or 6 for template matching table.
34: return$(carrier, utilities, modeColour)$ {Successful OR deduction.}
```

Customizing Qualitative Spatial and Temporal Calculi

Jochen Renz[1] and Falko Schmid[2]

[1] RSISE, The Australian National University, Canberra, ACT 0200, Australia
[2] SFB/TR8 Spatial Cognition, Universität Bremen, 28334 Bremen, Germany

Abstract. Qualitative spatial and temporal calculi are usually formulated on a particular level of granularity and with a particular domain of spatial or temporal entities. If the granularity or the domain of an existing calculus doesn't match the requirements of an application, it is either possible to express all information using the given calculus or to customize the calculus. In this paper we distinguish the possible ways of customizing a spatial and temporal calculus and analyze when and how computational properties can be inherited from the original calculus. We present different algorithms for customizing calculi and proof techniques for analyzing their computational properties. We demonstrate our algorithms and techniques on the Interval Algebra for which we obtain some interesting results and observations. We close our paper with results from an empirical analysis which shows that customizing a calculus can lead to a considerably better reasoning performance than using the non-customized calculus.

1 Introduction

Qualitative Spatial and Temporal Representation and Reasoning (QSTR) [1] is often useful when exact properties of spatial or temporal entities (e.g. exact location) are not important, but when we are interested in the relationships between different entities. A qualitative spatial or temporal calculus typically takes a particular set of spatial or temporal entities as its domain and defines relations over it to represent an aspect of space or time on a particular level of granularity.

Reasoning over these relations is usually done by exploiting *composition* of relations and the main reasoning problem is the *consistency problem* which asks if a given set of constraints over a set of relations has an instantiation which satisfies all the constraints. The consistency problem is NP-hard for most calculi if all relations are allowed, but in some cases tractable if only the base relations are used. Many spatial and temporal calculi have been defined in the literature [1] and they are all based on the same principles of having a particular domain and base relations over a particular aspect of space and time on a particular level of granularity. Some of these calculi have been intensively analyzed and computational properties are known for many of them.

Let us assume an application for which we want to represent qualitative spatial or temporal information and want to efficiently reason about this information. One problem we might face when selecting an existing calculus is that our application has a different domain than the existing calculus or requires relations on a different level of granularity. One way of dealing with this problem is to use the existing calculus and to ignore the differences. This is not very useful as the computational properties of the existing calculus only hold for the given domain and granularity. Another possibility is to develop a new calculus which is tailored towards our application. In this paper we look at a third possibility, namely, to customize an existing spatial or temporal calculus to a given application or to some given requirements. The result will also be a tailored calculus, but in many cases it will be possible to simply inherit known computational properties from the existing calculus.

Customizing calculi is not new and there are several examples where customizations of spatial or temporal calculi have been suggested. The best known example is probably RCC5 which is a subcalculus of RCC8 on a coarser level of granularity [3]. In this paper we go a step further and analyze customization in general. We distinguish different ways of customizing calculi, discuss properties which customized calculi have to satisfy, present algorithms for generating coarser calculi which satisfy these properties, and show when and how computational properties of coarser calculi can be inherited from existing calculi. We demonstrate our algorithms and methods on the Interval Algebra (IA) and derive some interesting results and observations. The advantage of using the IA is that we know all maximal tractable subsets of the IA [6] and can therefore easily test our general methods.

2 Qualitative Spatial and Temporal Calculi

A binary qualitative spatial or temporal calculus takes a domain of spatial or temporal entities $\mathcal{D}$ and defines a set of base relations $\mathcal{B}$ that partition $\mathcal{D} \times \mathcal{D}$ into jointly exhaustive and pairwise disjoint sets. Between any two values of the domain, exactly one of the base relations holds. Indefinite information can be expressed by using the union ($\cup$) of base relations, one base relation of the union must hold, but it is not yet known which of them. The set of all relations is therefore the powerset of the base relations $2^{\mathcal{B}}$. Usually the operators converse ($\smile$, or conv()), intersection ($\cap$), complement ($\neg$), and most importantly composition ($\circ$) are defined. Composition of two relations R, S is the relation defined as follows: $R \circ S = \{(a,c) | \exists b.(a,b) \in R \text{ and } (b,c) \in S\}$. Special relations are the universal relation U which is the union of all base relations, the empty relation $\emptyset$ and the identity relation id. A spatial or temporal calculus is a set of relations $2^{\mathcal{B}}$ which is closed under the operators, i.e., applying the operators to all relations always results in relations of the same set. Due to the definition of the relations, it is clear that $2^{\mathcal{B}}$ is always closed under converse, union, intersection, and complement, but it might not be closed under composition. For some calculi it is therefore necessary to use weak composition ($\circ_w$) instead of composition, which is defined as follows: $R \circ_w S = \{T \in \mathcal{B} | T \cap (R \circ S) \neq \emptyset\}$. $2^{\mathcal{B}}$ is always closed under weak composition.

Spatial and temporal information is usually represented using constraints over the relations, e.g., the constraint xRy, where x, y are variables over the domain $\mathcal{D}$ and $R \in 2^{\mathcal{B}}$, is satisfied if there is an instantiation of x and y with values $a, b \in \mathcal{D}$ such that $(a,b) \in R$. Given a set Θ of such constraints, an important reasoning problem is whether Θ is consistent, i.e., whether there are instantiations of all variables in Θ such that all constraints are satisfied. We write the consistency problem as $\mathsf{CSPSAT}(\mathcal{S})$ to indicate that only relations of the set $\mathcal{S}$ are used in Θ. This is a constraint satisfaction problem which is NP-hard in general. In order to enable efficient solutions to the consistency problem, the minimum requirement is that $\mathsf{CSPSAT}(\mathcal{B})$ is tractable. Ideally,

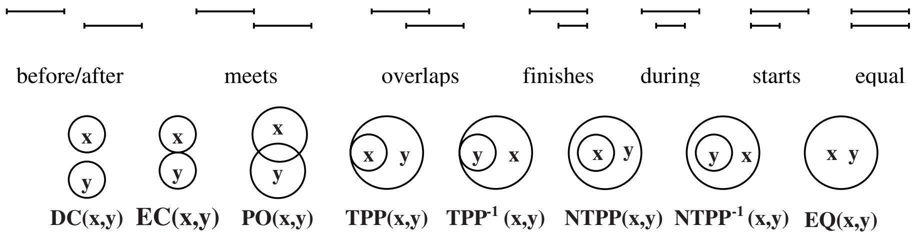


Fig. 1. Illustrations of the base relations of the Interval Algebra and of RCC8

we can identify large tractable subsets $\mathcal{T} \subseteq 2^{\mathcal{B}}$ for which $\mathsf{CSPSAT}(\mathcal{T})$ is tractable[18]. A maximal tractable subset is a tractable subset $\mathcal{T}$ such that CSPSAT is NP-hard for each superset of $\mathcal{T}$. If a tractable subset contains all base relations, it can be used to considerably speed up solving $\mathsf{CSPSAT}(2^{\mathcal{B}})$ [4].

A simple way of approximating consistency of a set Θ is to use the path-consistency algorithm which makes Θ path-consistent by applying the following operation to each triple of variables x, y, z of Θ until either a fixed point is reached or the empty relation is obtained (R_{xy} is the relation between x and y): $\forall x, y, z : R_{xy} := R_{xy} \cap (R_{xz} \circ R_{zy})$. If a calculus uses weak composition instead of composition, we have to use weak composition ($\circ_w$) in the given operation, and the corresponding algorithm is called *algebraic-closure algorithm* [5]. If weak composition is equal to composition, then algebraic-closure is equal to path-consistency. This operation can be performed in cubic time in the number of variables. If the empty relation occurs, then Θ is inconsistent, otherwise the resulting set Θ' is path-consistent/algebraically closed.

The best known spatial and temporal calculi are RCC8 [3] and the IA [7] (see Figure 1). The domain of RCC8 are extended spatial regions, defined as regular subsets of a topological space. RCC8 consists of eight base relations that distinguish topological relationships between spatial regions: DC (disconnected), EC (externally connected), PO (partial overlap), TPP (tangential proper part), NTPP (non-tangential proper part), their converses TPP^{-1} and NTPP^{-1}, and the identity relation EQ (equal).

The domain of the IA are intervals, defined as convex sets of a one-dimensional directed space. The IA consists of thirteen base relations: before ($<$), meets (m), overlaps (o), during (d), starts (s), finishes (f), their converse relations mi, oi, di, si, fi, and the identity relation equal ($=$). The IA is closed under composition and there is one maximal tractable subset (ORD-Horn) that contains all base relations [8] and 17 maximal tractable subsets which do not contain all base relations [6].

3 Customizing Spatial and Temporal Calculi

Qualitative spatial and temporal calculi use a particular domain $\mathcal{D}$ and partition $\mathcal{D} \times \mathcal{D}$ in a particular way into a set of base relations. The computational properties of a calculus hold only for the given choice of domain and base relations. If we have an application for which we want to use a spatial or temporal calculus it is possible that our application uses a different domain or requires different distinctions than those made by an existing calculus. This means that the computational properties of the existing calculus might not apply to our application; we have to develop a new calculus and prove its computational properties. In many cases, however, it is possible to customize an existing calculus to meet some more restricted requirements on domain or distinguished relationships, and to inherit computational properties from the existing calculus.

Assume that we have an application where we want to represent relations between intervals but where all the interval endpoints have to be distinct. As Golumbic and Shamir point out [9], this assumption is frequently made in combinatorics and has also been proved useful in applications of temporal reasoning. Under this assumption, we do not need the granularity offered by the Interval Algebra and would like to use coarser relations. There are two ways of how we can customize the Interval Algebra:

1. We use what Golumbic and Shamir call *macro relations* [9], i.e., unions of base relations. We combine IA base relations and use these macro relations as base relations of our customized calculus. For example, we could combine $\{m, o\}$, $\{mi, oi\}$, $\{s, d, f\}$, $\{si, di, fi\}$ and use them together with the relations $\{<\}$, $\{>\}$, and $\{=\}$ as our new base relations. This corresponds to the algebra $\mathcal{A}_7$ defined in [9].
2. We use only the relations we need, namely, the interval relations $<$, $>$, d, di, o, oi, $=$ and do not use m, mi, s, si, f, fi which correspond to intervals with common endpoints. This is similar to the algebra $\mathcal{A}_6$ defined in [9] with the exception of the identity relation which we include in order to be able to use the standard constraint satisfaction algorithms.

In both cases we use only seven "base relations" and their unions (i.e., 2^7 relations). But in both cases we get unwanted relations if we close these sets under composition, intersection and converse which happens if we compute path-consistency or apply other reasoning methods for the given set of constraints. In the first case, the composition of $\{<\}$ and $\{s, d, f\}$, for example, gives the relation $\{<, d, o, m, s\}$ which is outside the relations we wanted to use as f is not included. In the second case we get the same behavior, the composition of $<$ and d gives the relation $\{<, d, o, m, s\}$ which includes the base relations m and s which we did not want to use. In both cases, the relations are not closed under the operators and we end up having to use the full Interval Algebra again. This shows that even though it is straightforward to use a finer calculus for representing coarser information, we need to do a proper customization in order to benefit from having to represent only coarser information.

A good example is the RCC5 calculus consisting of five base relations: DR (discrete), PO (partial overlap), PP (proper part), its converse PP^{-1} and the identity relation EQ. RCC5 is closed under weak-composition and also under the other operators. Two of the RCC5 relations, PO and EQ are the same as the RCC8 relations, DR, PP, and PP^{-1} are macro relations: $\mathsf{DR} = \mathsf{DC} \cup \mathsf{EC}$, $\mathsf{PP} = \mathsf{TPP} \cup \mathsf{NTPP}$, $\mathsf{PP}^{-1} = \mathsf{TPP}^{-1} \cup \mathsf{NTPP}^{-1}$. RCC5 uses the same domain as RCC8 but uses base relations on a different level of granularity. RCC5 can be regarded as a customized version of RCC8 and many of the computational properties of RCC5 can be derived from computational properties of RCC8 as all 2^5 RCC5 relations are contained in RCC8.

But it is not always that simple. Consider the RCC7 calculus [2] which is a customized version of the RCC8 calculus where regions cannot overlap. The base relations of RCC7 are the same as those of RCC8 with the only difference that the RCC8 base relation PO is not allowed and not used. It has been shown that the consistency problem for RCC7 is NP-hard even if only the RCC7 base relations are used.

Another example are the 9-intersection relations [10]. RCC8 and the 9-intersection relations have a similar meaning and the composition tables are identical. However, the domain of RCC8 are regular regions of an arbitrary topological space while the domain of the 9-intersection relations a 2-D regions which are homeomorphic to disks, i.e., they consist of one piece and have no holes. The 9-intersection relations can be regarded as a customized version of RCC8 for a particular kind of spatial entities.

The previous examples can all be considered as customized versions of other calculi. We have seen three different kinds of customizations: (1) Combining base relations to form macro relations, (2) Excluding some of the base relations, (3) Restricting the domain. We can now summarize and formalize the different kinds of customization, and what basic requirements must be met by a customized calculus.

Definition 1 (Coarser Calculus). *Given a qualitative spatial or temporal calculus $\mathcal{F}$ with base relations $\mathcal{B}_\mathcal{F}$ over a domain $\mathcal{D}_\mathcal{F}$. A calculus C with base relations $\mathcal{B}_\mathcal{C}$ and universal relation $U_\mathcal{C}$ over a domain $\mathcal{D}_\mathcal{C}$ is called* coarser than $\mathcal{F}$, *written as $\mathcal{C} < \mathcal{F}$, iff (1) for each base relation $B_F \in \mathcal{B}_\mathcal{F}$ there is a base relation $B_C \in \mathcal{B}_\mathcal{C}$ such that $(B_F \cap U_\mathcal{C}) \subseteq B_C$, (2) $\mathcal{D}_\mathcal{C} \subseteq \mathcal{D}_\mathcal{F}$, and one of the following conditions holds:*

(a) macro relations: *there exists a base relation B_F in $\mathcal{B}_\mathcal{F}$ such that $(B_F \cap U_\mathcal{C}) \subset B_C$,*
(b) unused relation: *there exists a base relation B_F in $\mathcal{B}_\mathcal{F}$ such that $(B_F \cap U_\mathcal{C}) = \emptyset$,*
(c) reduced domain: *$\mathcal{D}_\mathcal{C} \subset \mathcal{D}_\mathcal{F}$.*

In this paper we define the customization of an existing calculus $\mathcal{F}$ as the process of finding a coarser calculus $\mathcal{C}$ which better matches particular requirements about domain and granularity than $\mathcal{F}$. Note that here we lift the requirement that base relations are jointly exhaustive in order to allow the possibility of having unused relations independently of restricting the domain. This is done by assuming that the universal relation U of a calculus is the union of all the base relations which can be smaller than the cross-product of the domain $\mathcal{D} \times \mathcal{D}$ which typically defines the universal relation. Furthermore the coarser calculus has all the properties of a normal calculus, i.e., it is closed under the operators, and is therefore different from tractable subsets of a calculus.

4 Customization Using Macro Relations

We have seen in the previous section that we cannot use any partition of base relations into macro relations in order to obtain a coarser calculus because the new calculus might not be closed under the operators. We implemented an algorithm which enumerates all possible partitions of a given set of base relations into macro relations and tests whether they are closed under the operators or not.

Proposition 1. *Given a set $\mathcal{B}$ of base relations which consists of two types of relations: (1) the relations $S_1, \ldots, S_k \in \mathcal{B}$ are equivalent to their own converse relation, i.e., $conv(S_i) = S_i$, (2) the relations $C_1, \ldots, C_h \in \mathcal{B}$ have a converse relation among themselves, i.e., $conv(C_i) = C_j$ for some $i \neq j$. All macro relations R of a valid partition of $\mathcal{B}$ must satisfy the following properties:*

- *If R contains a relation S_i, then for every relation B_j which is contained in R, the relation $conv(B_j)$ must also be contained in R.*
- *If R contains only relations C_i and $R \neq conv(R)$ then $conv(R)$ must also be contained in the partition.*

It is a mathematically interesting question to compute the number of partitions that satisfy the proposition, i.e., the number of partitions for any number h, k with $h + k = n$, but this is outside the scope of this paper. In Figure 2 we give an algorithm for computing all partitions of a set of base relations $\mathcal{B}$ which are closed under the operators. All the resulting partitions can be used as customized calculi. For the IA it turns out that there are only 16033 partitions which satisfy the proposition and only 117 of them are closed under the operators.

```
Algorithm: FINDMACROCALCULI($\mathcal{P}, \mathcal{R}$)
Input: a partial partition $\mathcal{P}$ of the base relations $\mathcal{B}$ and the remaining base relations $\mathcal{R}$

1. if $\mathcal{R} = \emptyset$ then
2.     if $closed(\mathcal{P})$ then print $\mathcal{P}$;
3. select one base relation $B \in \mathcal{R}$;
4. for all macro relations $S$ of $\mathcal{R}$ for which $B \in S$ do
5.     if $S \cap conv(S) \neq \emptyset$ and $S \neq conv(S)$ then continue;
6.     FINDMACROCALCULI($\mathcal{P} \cup \{S, conv(S)\}, \mathcal{R} \setminus \{S, conv(S)\}$)
```

Fig. 2. The recursive algorithm FINDMACROCALCULI computes all partitions of $\mathcal{B}$ which are closed under the operators when started with $\mathcal{P} = \emptyset$ and $\mathcal{R} = \mathcal{B}$

What we are interested in now, is when and how we can inherit computational properties from an existing calculus $\mathcal{F}$ to a customized calculus $\mathcal{C}$. We can only refer to computational properties if they are known for the original calculus, so we assume that we know one or more tractable subsets of the original calculus. For the case of having macro relations, inheriting computational properties is simple as all macro relations are also contained in the powerset of the original base relations $\mathcal{B}_{\mathcal{F}}$. Whenever the base relations $\mathcal{B}_{\mathcal{C}}$ of a customized calculus $\mathcal{C}$ are contained in a tractable subset of the original calculus, the intersection of $\mathcal{C}$ with any tractable subset of $\mathcal{F}$ is also tractable.

Lemma 1. *Given a calculus $\mathcal{F}$ with base relations $\mathcal{B}_{\mathcal{F}}$ over a domain $\mathcal{D}_{\mathcal{F}}$ and a known tractable subset $\mathcal{T}_{\mathcal{F}}$ of $\mathcal{F}$. If a calculus $\mathcal{C}$ has the same domain as $\mathcal{F}$, $\mathcal{C} < \mathcal{F}$, and all relations of $\mathcal{B}_{\mathcal{C}}$ are macro relations of $\mathcal{B}_{\mathcal{F}}$, then $\mathcal{C} \cap \mathcal{T}_{\mathcal{F}}$ is a tractable subset of $\mathcal{C}$. If algebraic closure decides* CSPSAT*($\mathcal{T}_{\mathcal{F}}$), then it also decides* CSPSAT*($\mathcal{C} \cap \mathcal{T}_{\mathcal{F}}$).*

The algorithm in Figure 2 can be modified to check if a customized calculus is contained in the known tractable subsets and to compute the intersection with the known tractable subsets. If the base relations of a customized calculus are macro relations of an existing calculus $\mathcal{F}$ and not contained in a known tractable subset of $\mathcal{F}$, then it is only possible to derive its complexity if all tractable subsets of $\mathcal{F}$ are known. Otherwise, its complexity has to be proved independently.

Table 1. All partitions of the Interval Algebra where deciding consistency is tractable

$\mathcal{M}_1 = \{=, < d\, o\, m\, s, > di\, oi\, mi\, si, f, fi\}$	$\mathcal{M}_2 = \{=, < d\, o\, m\, s, > di\, oi\, mi\, si, f\, fi\}$
$\mathcal{M}_3 = \{=, <> d\, di\, o\, oi\, m\, mi\, s\, si, f, fi\}$	$\mathcal{M}_4 = \{=, <> d\, di\, o\, oi\, m\, mi\, s\, si, f\, fi\}$
$\mathcal{M}_5 = \{=, < d\, o\, m\, s\, f, > di\, oi\, mi\, si\, fi\}$	$\mathcal{M}_6 = \{=, s, si, > d\, oi\, mi\, f, < di\, o\, m\, fi\}$
$\mathcal{M}_7 = \{=, s\, si, > d\, oi\, mi\, f, < di\, o\, m\, fi\}$	$\mathcal{M}_8 = \{=, > di\, oi\, mi\, si\, f, < d\, o\, m\, s\, fi\}$
$\mathcal{M}_9 = \{=, > d\, oi\, mi\, si\, f, < di\, o\, m\, s\, fi\}$	$\mathcal{M}_{10} = \{=, > d\, oi\, mi\, s\, f, < di\, o\, m\, si\, fi\}$
$\mathcal{M}_{11} = \{=, s, si, <> d\, di\, o\, oi\, m\, mi\, f\, fi\}$	$\mathcal{M}_{12} = \{=, s\, si, <> d\, di\, o\, oi\, m\, mi\, f\, fi\}$
$\mathcal{M}_{13} = \{=, <> d\, di\, o\, oi\, m\, mi\, s\, si\, f\, fi\}$	$\mathcal{M}_{14} = \{=\ s\, si, > d\, oi\, mi\, f, < di\, o\, m\, fi\}$
$\mathcal{M}_{15} = \{=\ s\, si, <> d\, di\, o\, oi\, m\, mi\, f\, fi\}$	$\mathcal{M}_{16} = \{< d\, o\, m\, s, > di\, oi\, mi\, si, =\ f\, fi\}$
$\mathcal{M}_{17} = \{<> d\, di\, o\, oi\, m\, mi\, s\, si, =\ f\, fi\}$	$\mathcal{M}_{18} = \{= <> d\, di\, o\, oi\, m\, mi\, s\, si\, f\, fi\}$

For the IA all maximal tractable subsets are known [6] and we can determine the computational properties for all 117 customized calculi we identified above. We applied our modified algorithm to the IA and tested for all 117 coarser calculi whether they are contained in any of the maximal tractable subsets. The result was surprising: (1) For 18 of the 117 valid partitions, all base relations are contained in a maximal trabtable subset. (2) For all 18, the powerset of the relations is also contained in a maximal tractable subset, i.e., the 117 customized calculi of the IA are either completely tractable or completely NP-hard. (3) All 18 are contained in ORD-Horn, the only maximal tractable subset which contains all base relations. Table 1 gives all 18 tractable partitions.

5 Customization by Unused Relations

For every spatial or temporal domain $\mathcal{D}$, we can find base relations which are jointly exhaustive and pairwise disjoint, i.e., for any two entities there will be exactly one base relation which holds for every pair. The universal relation, which is the union of all base relations is then equal to $\mathcal{D} \times \mathcal{D}$. If one or more of the base relations R_i cannot occur due to requirements of an application, then there are two possibilities of modifying a set of base relations $\mathcal{B}$. The first possibility is to say that the relations R_i are forbidden, i.e., they can not occur in any solution of the consistency problem over $2^{\mathcal{B}}$. We have to make sure that whenever we are forced to use a relation R_i, then our set of constraints is inconsistent. Alternatively, the second possibility is to say that the base relations R_i are empty and can be removed. This has the consequence that the base relations are not jointly exhaustive anymore and that the universal relation is not equal to $\mathcal{D} \times \mathcal{D}$ anymore, but it has the advantage of making reasoning simpler as we can just remove the unused relations. In the following we will always remove unused relations.

Given a calculus $\mathcal{F}$ with the base relations $\mathcal{B}_{\mathcal{F}}$, there are $2^{\mathcal{B}} - 1$ possibilities of removing unused base relations in order to obtain a coarser calculus $\mathcal{C}$. If we remove one or more base relations, we have to make sure that these relations are removed from all the compositions of relations of $\mathcal{C}$. All the relations which we do not remove are exactly the same relations as those of $\mathcal{F}$. When we remove base relations, the universal relation $U_{\mathcal{C}}$ of $\mathcal{C}$ is the union of all remaining base relations. Composition of relations C_i, C_j of $\mathcal{C}$ is therefore defined as follows: $C_i \circ_{\mathcal{C}} C_i = (C_i \circ_{\mathcal{F}} C_j) \cap U_{\mathcal{C}}$, where $\circ_{\mathcal{C}}$ is

Algorithm: FINDUNUSEDCALCULI($\mathcal{B}$)
Input: a set of base relations $\mathcal{B}$

```
1. for all relations $U \in 2^{\mathcal{B}}$ with $U = conv(U)$
2.     $\mathcal{N} = \emptyset$
3.     for all relations $B \in \mathcal{B}$
4.         if $\{B\} \cap U \neq \emptyset$ then $\mathcal{N} := \mathcal{N} \cup \{B, conv(B)\}$
5.     for all pairs of relations $N_i, N_j \in \mathcal{N}$ with $i \neq j$ do
6.         if $(N_i \circ N_j) \cap U = \emptyset$ then $\mathcal{N} = \emptyset$; break;
7.         $\mathcal{N} := \mathcal{N} \cup \{conv(N_i)\} \cup (\{N_i\} \cap \{N_j\}) \cup (N_i \circ N_j)$
8.     if $\mathcal{N} \neq \emptyset$ then print $\mathcal{N}$
```

Fig. 3. Algorithm FINDUNUSEDCALCULI computes all customized calculi of $\mathcal{B}$ which have unused relations and which satisfy Proposition 2

the composition of relations of $\mathcal{C}$ while $\circ_{\mathcal{F}}$ is the composition of relations of $\mathcal{F}$. Since the relations (and their converses) which we remove with this operation are empty, it is obvious that this way of computing composition is correct.

Another requirement a coarser calculus has to satisfy is that the composition of two relations must not give the empty relation. This can occur if the composition of two relations contains only unused relations. The following proposition lists the properties that a coarser calculus resulting from unused relations has to satisfy.

Proposition 2. *Given a set $\mathcal{B}_{\mathcal{F}}$ of base relations from which we remove the unused relations $S_1, \ldots, S_k \in \mathcal{B}$, resulting in the new set of base relations $\mathcal{B}_{\mathcal{C}}$. The relations $R_i, R_j \in \mathcal{C}$ must satisfy the following properties:*

(1) If $R_i \in \mathcal{C}$ then $conv(R_i) \in \mathcal{C}$;   (2) $R_i \circ_{\mathcal{C}} R_j \neq \emptyset$.

The algorithm in Figure 3 computes all coarser calculi $\mathcal{C}$ of a given calculus $\mathcal{F}$ which can be obtained by removing unused relations and which satisfy Proposition 2. We applied the algorithm to the IA and found that 63 calculi satisfied the requirements.

We cannot just compare the coarser calculi $\mathcal{C}$ with the tractable subsets of the original calculus $\mathcal{F}$. The reason for this is that the closure of subsets of $\mathcal{C}$ will be different from the closure of the same subsets of $\mathcal{F}$, and therefore it is possible that the same set of relations has different computational properties depending on whether it is a subset of $\mathcal{C}$ or of $\mathcal{F}$. But we are able to derive some properties if we look at what the actual closure of a set of relations of $\mathcal{C}$ is.

Lemma 2. *Given a calculus $\mathcal{F}$ and known tractable subsets $\mathcal{T}^i_{\mathcal{F}}$ of $\mathcal{F}$ for which algebraic closure decides consistency. Let $\mathcal{C}$ be a coarser calculus of $\mathcal{F}$ with base relations $\mathcal{B}_{\mathcal{C}}$ which results from $\mathcal{F}$ by removing unused base relations. If the closure $\widehat{\mathcal{B}_{\mathcal{C}}}$ of $\mathcal{B}_{\mathcal{C}}$ is contained in one of the sets $\mathcal{T}^i_{\mathcal{F}}$, then* CSPSAT*($\widehat{\mathcal{B}_{\mathcal{C}}}$) is tractable and can be decided by the algebraic closure algorithm.*

Proof. Given a set Θ of constraints over $\widehat{\mathcal{B}_{\mathcal{C}}}$ and let Θ' be the result of applying algebraic closure to Θ. If the empty relation occurs while computing algebraic closure, then Θ is inconsistent. Since $\widehat{\mathcal{B}_{\mathcal{C}}}$ is closed, all constraints of Θ' will also be from $\widehat{\mathcal{B}_{\mathcal{C}}}$. All

$\mathcal{R}_0^* = \{=\}$, $\mathcal{R}_1 = \{<,>\}$, $\mathcal{R}_2 = \{=,<,>\}$, $\mathcal{R}_3 = \{d, di\}$, $\mathcal{R}_4 = \{=, d, di\}$, $\mathcal{R}_5 = \{o, oi\}$, $\mathcal{R}_6 = \{=, o, oi\}$, $\mathcal{R}_7^* = \{d, di, o, oi\}$, $\mathcal{R}_8^* = \{=, d, di, o, oi\}$, $\mathcal{R}_9^* = \{<,>, d, di, o, oi\}$, $\mathcal{R}_{10}^* = \{=,<,>, d, di, o, oi\}$, $\mathcal{R}_{11}^* = \{=,<,>, d, di, o, oi, m, mi\}$, $\mathcal{R}_{12}^* = \{s, si\}$, $\mathcal{R}_{13}^* = \{=, s, si\}$, $\mathcal{R}_{14} = \{<,>, s, si\}$, $\mathcal{R}_{15} = \{=,<,>, s, si\}$, $\mathcal{R}_{16} = \{d, di, s, si\}$, $\mathcal{R}_{17} = \{=, d, di, s, si\}$, $\mathcal{R}_{18} = \{o, oi, s, si\}$, $\mathcal{R}_{19} = \{=, o, oi, s, si\}$, $\mathcal{R}_{20}^* = \{d, di, o, oi, s, si\}$, $\mathcal{R}_{21}^* = \{=, d, di, o, oi, s, si\}$, $\mathcal{R}_{22}^* = \{<,>, d, di, o, oi, s, si\}$, $\mathcal{R}_{23}^* = \{=,<,>, d, di, o, oi, s, si\}$, $\mathcal{R}_{24}^* = \{=,<,>, d, di, o, oi, m, mi, s, si\}$, $\mathcal{R}_{25}^* = \{f, fi\}$, $\mathcal{R}_{26}^* = \{=, f, fi\}$, $\mathcal{R}_{27} = \{<,>, f, fi\}$, $\mathcal{R}_{28} = \{=,<,>, f, fi\}$, $\mathcal{R}_{29} = \{d, di, f, fi\}$, $\mathcal{R}_{30} = \{=, d, di, f, fi\}$, $\mathcal{R}_{31} = \{o, oi, f, fi\}$, $\mathcal{R}_{32} = \{=, o, oi, f, fi\}$, $\mathcal{R}_{33}^* = \{d, di, o, oi, f, fi\}$, $\mathcal{R}_{34}^* = \{=, d, di, o, oi, f, fi\}$, $\mathcal{R}_{35}^* = \{<,>, d, di, o, oi, f, fi\}$, $\mathcal{R}_{36}^* = \{=,<,>, d, di, o, oi, f, fi\}$, $\mathcal{R}_{37}^* = \{=,<,>, d, di, o, oi, m, mi, f, fi\}$, $\mathcal{R}_{38}^* = \{d, di, o, oi, s, si, f, fi\}$, $\mathcal{R}_{39}^* = \{=, d, di, o, oi, s, si, f, fi\}$, $\mathcal{R}_{40}^* = \{<,>, d, di, o, oi, s, si, f, fi\}$, $\mathcal{R}_{41}^* = \{=,<,>, d, di, o, oi, s, si, f, fi\}$, $\mathcal{R}_{42}^* = \{<,>, d, di, o, oi, m, mi, s, si, f, fi\}$

Fig. 4. The 43 customized calculi of the Interval Algebra with unused relations and tractable base relations. The closure of the sets marked with $*$ are contained in ORD-Horn.

relations of $\widehat{\mathcal{B}_\mathcal{C}}$ are exactly the same relations as the corresponding relations of $\mathcal{F}$, so let $\Theta_\mathcal{F} = \Theta'$ be the corresponding set of constraints over $\mathcal{F}$. Since $R_i \circ_\mathcal{C} R_j \subseteq R_i \circ_\mathcal{F} Rj$, $\Theta_\mathcal{F}$ is also algebraically closed. Since $\widehat{\mathcal{B}_\mathcal{C}}$ is contained in a tractable subset of $\mathcal{F}$ for which algebraic closure decides consistency, $\Theta_\mathcal{F} = \Theta'$ is consistent.

Note that if $\widehat{\mathcal{B}_\mathcal{C}}$ is contained in a tractable subset of $\mathcal{F}$ for which algebraic closure does not decide consistency, then $\mathsf{CSPSAT}(\widehat{\mathcal{B}_\mathcal{C}})$ is still tractable. We added this test to our implementation of the algorithm of Figure 3 and applied it to the IA. It turned out that for 25 of the 63 coarser calculi $\widehat{\mathcal{B}_\mathcal{C}}$ is contained in ORD-Horn, and for 18 further coarser calculi, $\widehat{\mathcal{B}_\mathcal{C}}$ is contained in one of the other maximal tractable subsets of the IA. The 43 tractable calculi are given in Figure 4. We can obtain larger tractable subsets of our coarser calculi simply by intersecting them with the known tractable subsets of $\mathcal{F}$. This can be proved in the same way as Lemma 2.

6 Customization by Restricting the Domain

The computational properties of a qualitative calculus strongly depend on the used domain, as can be seen with RCC8 and 9-intersection. 9-intersection has a much more restricted domain than RCC8, but until recently it was unknown if the consistency problem for the 9-intersection problem is decidable at all [11]. This shows that reducing the domain does not necessarily make a calculus simpler. Other restrictions of the RCC8 domain that have been analyzed are the restriction of the domain to convex regions [12] and to closed disks [13]. A well-known example for a domain restriction in the interval domain is to use only intervals whose endpoints are integers.

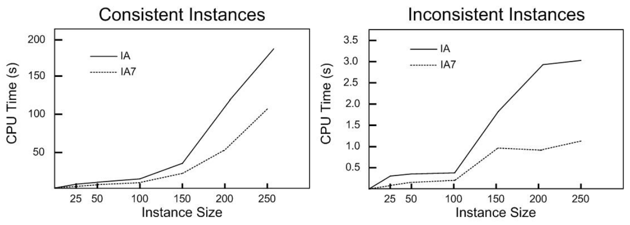
Due to the strong dependence of computational properties to the domain, there are no general methods as in the previous sections for inheriting complexity results to coarser calculi. Instead each customization requires a separate analysis. In the following we will give guidelines about how to inherit complexity results from a calculus $\mathcal{F}$ to a coarser calculus $\mathcal{C}$ with a reduced domain. First of all, we have to analyze how the composition is affected by removing values from the domain. (1) If it turns out that some base relations will not be used anymore, we can first apply the methods discussed in the previous section and then continue our analysis using the calculus and the results we obtain by applying these methods. (2) If the weak composition of the relation stays the same, we can test if the reduced domains allow us to refine some of the base relations to non-overlapping sub-relations. This is a method introduced and discussed in more detail in [14]. If this is not possible (as in the case of intervals over integer endpoints), then the computational properties of the coarser calculus $\mathcal{C}$ are the same as those of $\mathcal{F}$. If it is possible, then algebraic closure does not decide consistency even for the base relations of $\mathcal{C}$ and the computational properties of $\mathcal{F}$ can not be inherited . (3) If only some of the compositions change without completely removing some of the base relations of $\mathcal{F}$, then we have to make a new analysis and cannot inherit results from $\mathcal{F}$.

7 Benefits of Customization

Having a calculus with a smaller number of base relations has several benefits over using a calculus with a larger number of base relations. For larger calculi such as the Directed Intervals Algebra [15] it might not be possible to store the complete composition table in memory. Instead we might always have to compute composition from the basic composition table, which leads to slower reasoning. Independent of composition tables, using the smaller calculus is much more efficient because of the way the reasoning algorithms work, namely, by splitting relations into sub-relations and backtracking over the different sub-relations [16]. For a smaller calculus, relations can be split at most into the coarser base relations, but for larger calculi they are split into the finer base relations leading to much bigger backtracking trees and consequently much slower reasoning.

We empirically compared the reasoning performance of a coarser calculus with that of a finer calculus on exactly the same instances, i.e., we randomly generated only instances that can be expressed by the coarser calculus and then expressed the same instances with the finer calculus. We used a coarser calculus of the IA where intervals cannot have endpoints in common and compare it with the IA. The coarser calculus has seven base relations $<, >, o, oi, d, di, =$ and we call it IA7. It is obtained from the IA by removing all unused relations. Deciding consistency is tractable for the IA7 base relations, but NP-hard for the full calculus. We randomly generated 100 IA7 instances for different sizes. The constraints between the nodes of an instance were chosen randomly from IA7 and had an average density of $d = 5(\pm 2)$ constraints per node, which marks the *phase transition region* [17] for IA7. For our experiments we used the SFB/TR8 Generic Qualitative Reasoner[1] for checking consistency. The experiments where implemented on a Linux-PC with a 1,5Ghz VIA C7 CPU and 1GB RAM.

[1] https://sfbtr8.informatik.uni-freiburg.de/R4LogoSpace/Resources/GQR



As we can see from the graphs, we can solve the same instances much faster if we use IA7 for solving them instead of IA. In most cases we are more than twice as fast. This shows that customizing a calculus to a particular application can lead to considerable performance gains and is therefore highly recommended.

8 Conclusions

Sometimes applications require a coarser representation of spatial or temporal information than what existing calculi offer. In this paper we show different possibilities of how existing calculi can be customized to form coarser calculi which might be more suited for a given application. We present general algorithms for generating coarser calculi. We show when and how we can derive computational properties for coarser calculi from existing results and demonstrate empirically that using customized calculi leads to a much better reasoning performance than using non-customized calculi. We demonstrate our general methods on the IA for which we identify several coarser calculi and easily derive their computational properties.

References

1. Cohn, A.G., Renz, J.: Qualitative spatial Representation and Reasoning. In: Van Hermelen, F., Lifschitz, V., Porter, B.B. (eds.) Handbook of Knowledge Representation, Elsevier, Amsterdam (2007)
2. Gerevini, A., Renz, J.: Combining topological and size information for spatial reasoning. Artificial Intelligence 137(1-2), 1–42 (2002)
3. Randell, D.A., Cui, Z., Cohn, A.G.: A spatial logic based on regions and connection. In: Proc. of KR 1992, pp. 165–176 (1992)
4. Renz, J., Nebel, B.: Efficient Methods for Qualitative Spatial Reasoning. Journal of Artificial Intelligence Research 15, 289–318 (2001)
5. Ligozat, G., Renz, J.: What is a Qualitative Calculus? A General Framework. In: Zhang, C., Guesgen, H.W., Yeap, W.K. (eds.) PRICAI 2004. LNCS (LNAI), vol. 3157, pp. 53–64. Springer, Heidelberg (2004)
6. Krokhin, A., Jeavons, P., Jonsson, P.: Reasoning about temporal relations: The tractable subalgebras of Allen's Interval Algebra. J. ACM 50(5), 591–640 (2003)
7. Allen, J.F.: Maintaining knowledge about temporal intervals. CACM 26(11), 832–843 (1983)
8. Nebel, B., Bürckert, H.J.: Reasoning about temporal relations: A maximal tractable subclass of Allen's interval algebra. Journal of the ACM 42(1), 43–66 (1995)
9. Golumbic, M.C., Shamir, R.: Complexity and algorithms for reasoning about time: A graph-theoretic approach. Journal of the ACM 40(5), 1128–1133 (1993)

10. Egenhofer, M.J.: Reasoning about binary topological relations. In: Günther, O., Schek, H.-J. (eds.) SSD 1991. LNCS, vol. 525, pp. 143–160. Springer, Heidelberg (1991)
11. Schaefer, M., Sedgwick, E., Stefankovic, D.: Recognizing string graphs in NP. Journal of Computer and System Sciences (2003)
12. Davis, E., Gotts, N.M., Cohn, A.G.: Constraint networks of topological relations and convexity. CONSTRAINTS 4(3), 241–280 (1999)
13. Düntsch, I., Schmidt, G., Winter, M.: A necessary relation algebra for mereotopology. Studia Logica 69(3), 381–409 (2001)
14. Renz, J., Ligozat, G.: Weak composition for qualitative spatial and temporal reasoning. In: van Beek, P. (ed.) CP 2005. LNCS, vol. 3709, pp. 534–548. Springer, Heidelberg (2005)
15. Renz, J.: A Spatial Odyssey of the Interval Algebra: 1.Directed Intervals. In: Proc. IJCAI 2001, pp. 51–56 (2001)
16. Nebel, B.: Solving hard qualitative temporal reasoning problems: Evaluating the efficiency of using the ORD-Horn class. CONSTRAINTS 3(1), 175–190 (1997)
17. Cheeseman, P., Kanefsky, B., Taylor, W.M.: Where the *really* hard problems are. In: Proceedings of the 12th International Joint Conference on Artificial Intelligence, pp.331–337 (1991)
18. Renz, J.: Qulitative Spatial and Temporal Reasoning: Efficient Algorithms for Everyone. In: IJCAI -2007. Proceedings of the 20th International Joint Conference on Artificial Intelligence, Hyderabad, India (January 2007)

extraRelief: Improving Relief by Efficient Selection of Instances

Manoranjan Dash and Ong Cher Yee

School of Computer Engineering, Nanyang Technological University, Singapore
{asmdash,ongc}@ntu.edu.sg

Abstract. In this paper we propose a modified and improved Relief method, called extraRelief. Relief is a popular feature selection algorithm proposed by Kira and Rendell in 1992. Although compared to many other feature selection methods Relief or its extensions are found to be superior, in this paper we show that it can be further improved. In Relief, in the main loop, a number of instances are randomly selected using simple random sampling (srs), and for each of these selected instances, the nearest hit and miss are determined, and these are used to assign ranks to the features. srs fails to represent the whole dataset properly when the sampling ratio is small (i.e., when the data is large), and/or when data is noisy. In extraRelief we use an efficient method to select instances. The proposed method is based on the idea that a sample has similar distribution to that of the whole. We approximate the data distribution by the frequencies of attribute-values. Experimental comparison with Relief shows that extraRelief performs significantly better particularly for large and/or noisy domain.

1 Introduction

Feature selection is often used as a preprocessing solution to address the problem of high-dimensionality [4]. It selects a subset of features based on some criteria. These criteria vary but, in general, accuracy is considered to be an agreed upon measure to determine the quality of feature selection. So, a rough definition of feature selection is, choose an optimal number of features so that the accuracy using the selected features is not significantly dropped. However, as often experienced, the accuracy increases on many occasions.

A popular feature selection method is Relief by Kira and Rendell (1992[6]). This method is further analyzed by Kononenko in [7] and recently in [9]. In Figure 1 we give the algorithm for the basic Relief algorithm. Relief is a simple yet efficient algorithm. It chooses n instances randomly using simple random sampling (srs). For each selected instance R_i it determines nearest hit and nearest miss where 'hit' means both R_i and the "nearest hit" belong to the same class, whereas miss means R_i and the "nearest miss" belong to two different classes. The feature weight vector W is updated for such each iteration. If for a feature A, R_i and "nearest hit" have very different values, then weight of A is incremented less, i.e., its relevance is less in discriminating the classes. On the other hand, if for a feature A, R_i and "nearest miss" have very close values, then weight of A is incremented less, i.e., its relevance is less in discriminating the classes. This simple technique is used to determine the discriminating power of each feature. At the end of the main loop, the weight vector W gives the relevance of each feature to discriminate among the classes.

```
Algorithm RELIEF
Input: Training data with class values
Output: Vector $W$ of estimations of relevance of each feature

1.      set all weights $W[A] = 0.0$;
2.      for $i = 1$ to $n$ do begin
3.          randomly select an instance $R_i$;
4.          find nearest hit $H$ and nearest miss $M$;
5.          for $A = 1$ to $a$ do
6.              $W[A] = W[A] - diff(A, R_i, H)/m + diff(A, R_i, M)/m)$;
7.      end;
```

Fig. 1. RELIEF Algorithm

In [9] many aspects of RELIEF and its extensions have been studied. One aspect is the number of instances (n) selected in the main loop. In this paper we focus on the method of choosing n instances so as to improve the performance of RELIEF. To motivate and explain the significance of the proper choice of n instances inside the main loop, we have shown a plot in Figure 2. In this plot there are two features $A1$ and $A2$. There are two classes $Class1$ and $Class2$. In the plot there are two big clusters (each with 500 instances) having class labels $Class1$ and $Class2$ respectively. But there is one outlier (let us name it $o1$) having class label $Class2$. If $o1$ is selected among the n instances in the main loop, then it will have a positive effect on the weight of $A2$. The reason is as follows. Nearest miss of $o1$ is some instance from $Class1$. The "nearest miss" is separated nicely from $o1$ on feature $A2$ compared to feature $A1$. So, although $o1$ is an outlier, because it has been chosen among the n selected instances, it ended up positively (yet **wrongly**) affecting the feature $A2$. This may lead to wrong ranking among the features $A1$ and $A2$. In this case $A2$ gathered weight because of the outlier $o1$.

Note that even with the modification by Kononenko [7] (RELIEFF) to replace 1-nearest neighbor by k-nearest neighbor this problem persists. Although RELIEFF is more robust in the presence of noise, it cannot help when a noisy point is selected among the n instances. The reason is, RELIEFF has no control over the selection of noisy points among the n instances.

So it is very important to choose the n instances for the main loop wisely, i.e., these instances should be representative of the whole dataset. In RELIEF and its extensions, simple random sampling (SRS) is used for selecting the instances. Although SRS works reasonably well in obtaining a representative sample when the sampling ratio is moderate or large, it fails to represent the whole data

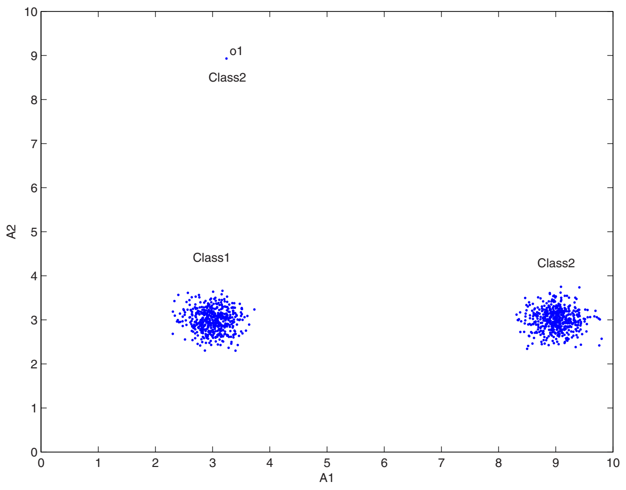


Fig. 2. A dataset with two distinct classes and one outlier

properly when sampling ratio is small [2]. Smaller sampling ratios are typically encountered when the whole dataset is very large and the memory is limited. SRS also fails when data is noisy. True to its definition, SRS treats the noise as bona fide and thus ends up selecting in equal percentage from both noisy data and non-noisy data. Such random selection of instances causes the ranking of the features to be incorrect (as shown earlier) thus leading to poor performance of the classifier.

In this paper we want to arrest such downfall of performance by proper selection of instances. We propose an algorithm that can choose n instances based on their representativeness to the whole data, and carefully yet elegantly avoids the outliers from being selected. The proposed instance selection algorithm works particularly well when sampling ratio (i.e., $\frac{n}{N}$) is small where N is the total number of instances in the whole dataset. It also works very well when data has noise.

Related Work: RELIEF is further analyzed by Kononenko in [7] and recently in [9]. They conduct experiments on different aspects of RELIEF and propose extensions. In [9] they carry out experiments to show the performance behavior when n varies, where n is the number of selected instances inside the main loop. Their study shows that stable performance is obtained typically when n is more than 50. They did an experiment to understand the behavior of SRS when data has noise. The noise they added is of misclassification type, i.e., some of the class labels were randomly changed. They used two metrics to evaluate, *separability* and *usability*. *Separability* is the difference in the lowest estimate of the important features and the highest estimate of the un-important features. *Usability* is the difference between the highest estimate of the important features and and highest estimate of the un-important features. Their experiment with noise showed that, even in presence of some noise RELIEF is able to output positive *separability* and *usability*. In this paper we make RELIEF even better at handling noise. They also conducted experiments to see at what minimal value of n Relief outputs positive values for *separability* and *usability*. Their experiments show that Relief requires small value for n to achieve this. The modified extraRelief proposed in this paper shows that by choosing the n instances properly one will require even smaller number of instances than Relief. In [5], Draper et al. showed that Relief is biased against non-monotonic features including Gaussian features. They propose a method to remove this bias. In [1], the authors proposed a method to remove the redundancy among the features selected using Relief. Instead of estimating the relevance of individual features, they used an "OR" operator to evaluate a feature set. If a feature in a set of features is able to discriminate two examples, then the feature set is also able to discriminate the two examples.

2 Efficient Selection of Instances and extraRelief

In this section we describe a novel and efficient method to select instances so that the performance of Relief increases. The idea is borrowed from the transactional market basket kind of data where each transaction has several items, and each item has a frequency in the whole database which is based on the count of transactions that contains the item. In this paper, frequency means relative frequency. *A sample of the whole database is considered a good representative if the frequencies of the items in the sample are very close to that of the corresponding frequencies in the whole database.* Note that these notations are for transactional data where an item is like a binary feature taking the value 0 or 1. In the context of feature selection, an item is equivalent to a combination of a feature and one of its value. For example, if a feature $A1$ takes the values {a, b, c}, then it can be understood as 3 items for each of 'a', 'b', 'c'. Thus when an instance has $A1 = a$ then the item corresponding to the value 'a' is set to '1' and the other two are set to '0'. If the data consists of continuous data, the data is first of all discretized using a discretization method [8] in order to determine the frequencies. In the rest of this section we discuss a method for efficiently selecting a sample that satisfies this constraint. We first of all give some notations that are used to describe the proposed method.

Notations: Denote by D the database of interest, by S a simple random sample drawn without replacement from D, and by I the set of all items that appear in D. Let $N = |D|$, $n = |S|$ and $m = |I|$. Here $|\cdot|$ means the number of data or items in the corresponding dataset or itemset. Also denote by $\mathcal{I}(D)$ the collection of itemsets that appear in D; a set of items A is an element of $\mathcal{I}(D)$ if and only if the items in A appear jointly in at least one transaction $t \in D$. If A contains exactly $k(\geq 1)$ elements, then A is sometimes called a k-itemset. In particular, the 1-itemsets are simply the original items. The collection $\mathcal{I}(S)$ denotes the itemsets that appear in S; of course, $\mathcal{I}(S) \subseteq \mathcal{I}(D)$. For $k \geq 1$ we denote by $\mathcal{I}_k(D)$ and $\mathcal{I}_k(S)$ the collection of k-itemsets in D and S, respectively. For an itemset $A \subseteq I$ and a transactions set T, let $n(A;T)$ be the number of transactions in T that contain A. The support of A in D and in S is given by $f(A;D) = n(A;D)/|D|$

and $f(A;S) = n(A;S)/|S|$, respectively. Given a threshold $s > 0$, an item is frequent in D (resp., in S) if its support in D (resp., in S) is no less than s. m is the number of items. Specifically, denote by S^i the set of all transactions in S that contain item A_i, and by r_i and b_i the number of red and blue transactions in S^i respectively. Red means the transactions will be kept in the final subsample and blue means the transactions will be deleted. Q is the penalty function of r_i and b_i. f_r denotes the ratio of red transactions, i.e., the sample ratio. Then the ratio of blue transactions is given by $f_b = 1 - f_r$.

Binarization: The proposed method requires that data should be formatted like transactional records typically found in association rule mining applications where a transaction consists of some items and each item takes 0/1 value. For classification and clustering tasks, data should be binarized if it is not already in binary format. We use a simple binarization technique: if an attribute is of nominal type then each such nominal value is converted to a binary attribute, i.e., if attribute A_1 takes five different nominal values, then after binarization there will be five binary attributes (or in terms of association rule mining, items) in place of A_1; if an attribute is continuous, it is converted to two values 0 and 1 by determining its arithmetic mean and values less than mean is converted to 0, otherwise 1. Thus each continuous attribute is converted to an item.

2.1 Selecting Sample to Minimize Distance

In order to obtain a good representation of a huge database, ε-approximation method is used to find a small subset so that the supports of 1-itemset are close to those in the entire database. The sample S_0 of S is an ε-approximation if its discrepancy satisfies $Dist(S_0, S) \leq \varepsilon$. The discrepancy is computed as the distance of 1-itemset frequencies between any subset S_0 and the superset S. It can based on L^p-norm distances, for example: [3]

$$Dist_1(S_0;S) = \sum_{A \in I_1(S)} |f(A;S_0) - f(A;S))| \tag{1}$$

$$Dist_2(S_0;S) = \sum_{A \in I_1(S)} (f(A;S_0) - f(A;S))^2 \tag{2}$$

In this paper $Dist_\infty$ metric is used as the distance metric and the discrepancy is calculated as follows:

$$Dist_\infty(S_0, S) = \max_{A \in I_1(S)} |f(A;S_0) - f(A;S)|. \tag{3}$$

The concept of ε-approximation has been widely used in machine learning [10]. A seminal result of Vapnik and Chervonenkis shows that a random sample of size $O(\frac{d}{\varepsilon^2} \log \frac{1}{\varepsilon})$, where d is the "VC dimension" (assumed finite) of the dataset, is an ε approximation. This result establishes the link between random samples and frequency estimations over several items simultaneously.

Each transaction is either selected (colored as red) or rejected (colored as blue) based on a penalty function Q_i for item A_i. A hyperbolic cosine based penalty

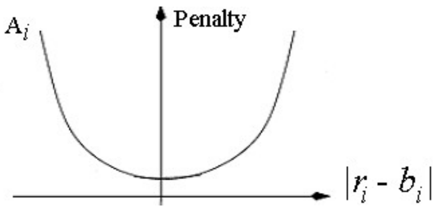


Fig. 3. Penalty as a function of $|r_i - b_i|$

function is applied. When the sampling ratio $f_r = \frac{|S_0|}{|S|}$ is 0.5, the penalty function has the shape depicted in Figure 3. Q_i is low when $r_i = b_i$ approximately, otherwise Q_i increases exponentially in $|r_i - b_i|$. The penalty function Q_i for each item A_i is converted into exponential equivalent shown as follows:

$$Q_i = 2\cosh(\delta_i(r_i - b_i)) \tag{4}$$
$$= \exp(\delta_i(r_i - b_i)) + \exp(-\delta_i(r_i - b_i)) \tag{5}$$

where δ_i controls the penalty function.

In order to directly obtain a sample set of any sample ratio f_r (not just 0.5), set the ratio of red transactions as $r_s = f_r$ and the ratio of blue transactions is $f_b = 1 - r_s$. Then we have

$$r_i = f_r \cdot |S^i| \tag{6}$$
$$b_i = f_b \cdot |S^i| \tag{7}$$

So

$$\frac{r_i}{f_r} = \frac{b_i}{f_b} = |S^i| \tag{8}$$
$$\frac{r_i}{2f_r} + \frac{b_i}{2f_b} = |S^i| \tag{9}$$

The penalty function Q_i tries to minimize:

$$\left|\frac{r_i}{2f_r} - \frac{b_i}{2f_b}\right| \tag{10}$$

instead of $|r_i - b_i|$.

The modified penalty function Q_i for each item A_i is

$$Q_i = 2\cosh(\delta_i(\frac{r_i}{2f_r} - \frac{b_i}{2f_b})) \tag{11}$$
$$= \exp(\delta_i(\frac{r_i}{2f_r} - \frac{b_i}{2f_b})) + \exp(-\delta_i(\frac{r_i}{2f_r} - \frac{b_i}{2f_b})) \tag{12}$$

Because δ_i is small, $\exp(\delta_i)$ and $\exp(-\delta_i)$ can be replaced by $(1 + \delta_i)$ and $(1 - \delta_i)$. Thus, Equation 12 is modified as follows.

The modified penalty Q_i of j-th transaction will be:

$$Q_i = Q_i^{(j)} = Q_{i,1}^{(j)} + Q_{i,2}^{(j)} \tag{13}$$

$$Q_{i,1}^{(j)} = (1+\delta_i)^{\frac{r_i}{2f_r}}(1-\delta_i)^{\frac{b_i}{2f_b}} \tag{14}$$

$$Q_{i,2}^{(j)} = (1-\delta_i)^{\frac{r_i}{2f_r}}(1+\delta_i)^{\frac{b_i}{2f_b}} \tag{15}$$

where $Q_i^{(j)}$ means the penalty of ith item in jth transaction and δ_i controls the steepness of the penalty plot. The initial values of $Q_{i,1}$ and $Q_{i,2}$ are both 1.

Suppose the $(j+1)$-th transaction is colored as r (or b), the corresponding penalty function $Q_i^{(j||r)}$ (or $Q_i^{(j||b)}$) is:

$$\begin{aligned} Q_{i,1}^{(j||r)} &= (1+\delta_i)^{\frac{r_i+1}{2f_r}}(1-\delta_i)^{\frac{b_i}{2f_b}} \\ &= (1+\delta_i)^{\frac{1}{2f_r}}(1+\delta_i)^{\frac{r_i}{2f_r}}(1-\delta_i)^{\frac{b_i}{2f_b}} \\ &= (1+\delta_i)^{\frac{1}{2f_r}} Q_{i,1}^{(j)} \end{aligned} \tag{16}$$

$$\begin{aligned} Q_{i,2}^{(j||r)} &= (1-\delta_i)^{\frac{1}{2f_r}} Q_{i,2}^{(j)} \\ Q_{i,1}^{(j||b)} &= (1-\delta_i)^{\frac{1}{2f_b}} Q_{i,1}^{(j)} \\ Q_{i,2}^{(j||b)} &= (1+\delta_i)^{\frac{1}{2f_b}} Q_{i,2}^{(j)} \end{aligned} \tag{17}$$

The computation process of $Q_{i,1}^{(j||r)}$ is given in Equation 16. Other penalty functions are computed with a similar procedure and the results are shown in Equation 17.

The penalty function of the current transaction is the summation of penalties for all items. If $Q^{(j||b)} = \sum_i Q_i^{(j||b)}$ is less than $Q^{(j||r)} = \sum_i Q_i^{(j||r)}$, the $(j+1)$-th transaction will be colored blue and rejected. Otherwise, it will be colored red and added to the sample.

An important observation is that $Q_i^{(j\|r)} + Q_i^{(j\|b)} = 2Q_i^{(j)}$, and so when summing over all items, by linearity $Q^{(j\|r)} + Q^{(j\|b)} = 2Q^{(j)}$. It follows that

$$\min\Big(Q^{(j\|r)}, Q^{(j\|b)}\Big) \le \frac{1}{2}\Big(Q^{(j\|r)} + Q^{(j\|b)}\Big) = Q^{(j)}.$$

This means that the coloring chosen by the algorithm *never* increases the penalty!

By the symmetry of the penalty function, and since $\frac{r_i}{2f_r} + \frac{b_i}{2f_b} = |S^i|$, we have

$$Q_i^{(final)} \ge (1+\delta_i)^{\left|\frac{r_i}{2f_r} - \frac{b_i}{2f_b}\right|}(1-\delta_i^2)^{|S^i|}$$

where $Q_i^{(final)}$ is the penalty at the end of all transactions. From this and the fact that $Q^{(init)} = 2m$, we get that the final overall penalty is at most $2m$, hence $Q_i^{(final)} \le 2m$ for every i, which in turns implies that

$$\left|\frac{r_i}{2f_r} - \frac{b_i}{2f_b}\right| \le \frac{\ln(2m)}{\ln(1+\delta_i)} + \frac{|S^i|\ln(1/(1-\delta_i)^2)}{\ln(1+\delta_i)}.$$

We can choose the value of δ_i to make the right-hand side as small as possible. The first (resp., second) term in the sum is decreasing (resp. increasing) in δ_i, and so a reasonable choice is to balance the two terms, leading to

$$\delta_i = \sqrt{1 - \exp\Big(-\frac{\ln(2m)}{|S^i|}\Big)}. \tag{18}$$

For that value, it can be shown that the final error on item i is at most $O(\sqrt{|S_i|\log(2m)})$ which, if S^i is not too small, is much smaller than $|S^i|$.

In order to guarantee that there are $f_r * |S|$ number of red transactions, we can add a (fictitious) item A_0 that is contained in all transactions. In that case, $\frac{S_r}{2f_r} - \frac{S_b}{2f_b}$ is also $O(\sqrt{n\log(2m)})$, and this implies that $|S_r| = n * f_r + O(\sqrt{n\log(2m)})$. Since $f(A_i; S_r) = r_i/|S_r| = r_i/(n * f_r + O(\sqrt{n\log(2m)}))$, and $f(A_i; S) = (r_i + b_i)/n$, from the above bounds on $\left|\frac{r_i}{2f_r} - \frac{b_i}{2f_b}\right|$ we get that for each i,

$$|f(A_i; S_r) - f(A_i; S)| \leq \varepsilon(n, m) = O(\sqrt{\log(2m)/n}). \tag{19}$$

In practice, the proposed method will work with any choice of δ_i, but the bounds on $|f(A_i; S_r) - f(A_i; S)|$ will not necessarily be guaranteed. In our implementation, we have found that setting

$$\delta_i = \sqrt{1 - \exp\Big(-\frac{\ln(2m)}{n}\Big)}$$

is very effective. The advantage is that if the defining parameters of the database, i.e., the number n of transactions and number m of items, are already known then the proposed method requires a single scan of the database. It also guarantees that $\left|\frac{r_i}{2f_r} - \frac{b_i}{2f_b}\right|$ is at most $O(\sqrt{n\log(2m)})$ for any item, which is not as strong as $O(\sqrt{|S_i|\log(2m)})$, but still strong enough to guarantee the upper bound of (19).

2.2 Handling Noise

Noise is a random error or variance in a measured variable. In terms of data mining, the error may appear both in attributes and the class variable of the data. As SRS randomly selects the sample set, the noisy data in the sample set has the same percentage as the original data when the experiment repeats for several times. For the proposed sampling method, from Equation 16 and Equation 17, it can be seen that as the sample ratio f_r is usually very small, i.e.,$f_r \ll f_b$, and $Q_i^{(j||b)}$ is smaller than $Q_i^{(j||r)}$ in the beginning. That means only after painting sufficient number of transactions as blue, the proposed sampling method paints some transactions as red. For the data with noisy items, their corresponding frequency are very low. In the red sample these noisy transactions are usually not selected because first of all the blue sample has to be considerably filled with such noisy transactions for even one such noisy transaction to be painted red. But for usual non-noisy items the corresponding frequency is maintained, thus the proposed algorithm retains *bona fide* transactions and removes the noisy transactions. Usually in outlier detection one needs to input some threshold which is not easy determine off-hand. But, note that the proposed method removes the outliers without any user given threshold.

3 Experimental Results

Experiments are conducted to compare the performance of traditional RELIEF and the proposed EXTRARELIEF. Seven datasets from UCI machine learning repository (www.ics.uci.edu/mlearn/MLRepository.html) were taken. A summary of the datasets is as follows: {Cars, #Instances = 1728, #Features = 6, #Classes = 4}, {Corral-46 [1], #I = 128, #F = 46, #C = 2}, {Kr-vs-kp, #I = 3196, #F = 36, #C = 2}, {LED, #I = 3196, #F = 24, #C = 10}, {Mushroom, #I = 8124, #F = 22, #C = 2}, {Nursery, #I = 12960, #F = 8, #C = 2}, {Tic-Tac-Toe, #I = 958, #F = 9, #C = 2}.

Table 1. Bold font stands for accuracy of EXTRARELIEF, Italic font stands for accuracy of RELIEF

Sample Ratio	10 Features	20 Features	Sample Ratio	10 Features	20 Features
	Cars			Mushroom	
1%	*68.0* **70.3**	*69.0* **79.0**	1%	*46.9* **61.9**	*79.3* **85.4**
3%	*71.5* **72.0**	*76.6* **83.1**	3%	*73.9* **76.6**	*81.4* **79.3**
	Corral-46			Nursery	
1%	*55.3* **88.7**	*68.7* **89.2**	1%	*43.9* **53.3**	*40.9* **61.1**
3%	*58.5* **89.6**	*73.2* **90.4**	3%	*25.5* **50.5**	*23.5* **46.8**
	Kr-vs-Kp			Tic-Tac-Toe	
1%	*73.0* **73.2**	*68.9* **76.4**	1%	*56.8* **71.6**	*68.4* **77.9**
3%	*82.1* **83.6**	*83.9* **85.8**	3%	*63.2* **73.7**	*71.6* **85.3**

We added 10-20% of noise (probability p_{noise}) in each dataset. We perform 10-fold cross validation. The dataset is divided into 10 folds, in each run one fold is used as test set and the other 9 folds are used for training. To perform EXTRARELIEF we perform the following. We run the proposed sampling algorithm on the training set to get a sample. The sample ratios vary from 1–5 % of the training set. The chosen sample is used in RELIEF as selected instances to rank the features. Then we select a series of top ranked features. To perform the traditional RELIEF we run SRS to choose a sample randomly and use it to run RELIEF. Then we select a series of top ranked features. We choose a decision tree (C4.5) as the classifier to compare the accuracy of the chosen feature by the two methods. C4.5 is trained using the selected features over the training

[1] A modified Corral dataset used in [11].

set. Then finally the corresponding accuracies for the traditional RELIEF and EXTRARELIEF are compared. In Table 1 we show the accuracies of the two methods for two sample ratios (1% and 3%) and two sets of top ranked features (top 10 and top 20).

Results show that EXTRARELIEF outperforms RELIEF almost all the time. The gain is very significant in many cases. For example, for Corral-46 dataset, 1% sample ratio and top 10 features, the gain is as much as 33%. EXTRARELIEF's characteristic nature is to not select noisy points, whereas RELIEF is blind to noise and non-noise.

We experimented the time taken by the two algorithms, and the result is very similar. Due to space constraint the results are not shown here.

In conclusion, we proposed a modification of the popular RELIEF algorithm. The idea is that if the sampled instances are selected such that noisy data is not selected then the performance of RELIEF increases, sometime very significantly.

References

1. Arauzo-Azofra, A., Benitez-Sanchez, J.M., Castro-Pena, J.L.: A feature set measure based on relief. In: RASC. Proceedings of the 5th International Conference on Recent Advances in Soft Computing, pp. 104–109 (2004)
2. Brönnimann, H., Chen, B., Dash, M., Haas, P., Scheuermann, P.: Efficient data reduction with EASE. In: SIGKDD. Proceedings of 9th International Conference on Knowledge Discovery and Data Mining, pp. 59–68 (2003)
3. Chen, B., Haas, P., Scheuermann, P.: A new twophase sampling based algorithm for discovering association rules. In: SIGKDD. Proceedings of International Conference on Knowledge Discovery and Data Mining (2002)
4. Dash, M., Liu, H.: Feature selection for classification. International Journal of Intelligent Data Analysis 1(3) (1997)
5. Draper, B., Kaito, C., Bins, J.: Iterative relief. In: Proceedings of Workshop on Learning in Computer Vision and Pattern Recognition (2003)
6. Kira, K., Rendell, L.A.: The feature selection problem: Traditional methods and a new algorithm. In: AAAI. Proceedings of Ninth National Conference on AI (1992)
7. Kononenko, I.: Estimating attributes: Analysis and extension of RELIEF. In: Bergadano, F., De Raedt, L. (eds.) ECML 1994. LNCS, vol. 784, pp. 171–182. Springer, Heidelberg (1994)
8. Liu, H., Hussain, F., Tan, C.L., Dash, M.: Discretization: An enabling technique, pp. 393–423 (2002)
9. Robnik-Sikonja, M., Kononenko, I.: Theoretical and empirical analysis of ReliefF and RReliefF. Machine Learning Journal 53, 23–69 (2003)
10. Valiant, L.G.: A theory of the learnable. Communications of the Association for Computing Machinery 27(11), 1134–1142 (1984)
11. Yu, L., Liu, H.: Efficient feature selection via analysis of relevance and redundancy 5, 1205–1224 (2004)

Constraint-Based Mining of Web Page Associations

Mohammad El-Hajj, Jiyang Chen, Osmar R. Zaïane, and Randy Goebel

University of Alberta, Canada
{mohammad, jiyang, zaiane, goebel}@cs.ualberta.ca

Abstract. The use of association rule mining carries the attendant challenge of focusing on appropriate data subsets so as to reduce the volume of association rules produced. The intent is to heuristically identify "interesting" rules more efficiently, from less data. This challenge is similar to that of identifying "high-value" attributes within the more general framework of machine learning, where early identification of key attributes can profoundly influence the learning outcome. In developing heuristics for improving the focus of association rule mining, there is also the question of where in the overall process such heuristics are applied. For example, many such focusing methods have been applied *after* the generation of a large number of rules, providing a kind of ranking or filtering. An alternative is to constrain the input data earlier in the data mining process, in an attempt to deploy heuristics in advance, and hope that early resource savings provide similar or even better mining results. In this paper we consider possible improvements to the problem of achieving focus in web mining, by investigating both the articulation and deployment of rule constraints to help attain analysis convergence and reduce computational resource requirements.

1 Introduction

Anyone with experience in searching for trends in large data volumes will understand that it is easier to find "what you're looking for" when you know what factors contribute to those data artifacts you seek. It is also unlikely that, having deployed *any* association rule mining system, one has not further considered what might be provided before re-running the mining system in order to produce fewer and more "interesting" if not more relevant outputs. At least one component of our interest is that of identifying *any* methods which might consistently reduce the expenditure of computational resources, and increase the value of data mining results.

An association rule is an expression of some relationship between a conjunction of observed features in the antecedent of the rule and a conjunction of observed features in the consequent of the rule, typically based on frequency of co-occurrence called support. It is quickly apparent that the sheer number of discovered association rules makes it difficult to find interesting ones and hence the focus on relevant navigational patterns in the case of web mining. Increasing the support threshold leads to overlooking rare but pertinent patterns, while reducing the support threshold yields to an "explosion" in the number of patterns. The discovery of these patterns becomes problematic, not only because of the huge number of discovered rules, but also because of performance: high memory dependencies, huge search space, and massive I/O operations. To reduce the effects of these problems we devised a new method with a fast traversal technique which drastically reduces the search space. Using constraints that reduce the output size while directly discovering patterns that are of interest to the user, is germane to the problem at hand: focusing on a subset of patterns to analyze. In addition to the problem of knowing what kind of knowledge might heuristically improve focus in reducing resource consumption and improve results during data mining, there is the problem of managing the evaluation of such methods. Typical association rule mining systems produce sets of rules well beyond the ability of any direct human interpretation, so it is important to provide sensible and intuitive methods of rule set evaluation. In addition to our interest in determining the content and form of methods to constrain data mining input to produce improved results, we also deploy our own method of visualizing association rules, in order to provide high level evaluation of our methods. inputs.

1.1 Contributions

We introduce our efficient algorithm for mining frequent itemsets while considering two types of constraints during the mining process, namely *anti-monotone* and *monotone* constraints.

We apply our algorithm on web navigational data with constraints to generate focused associations between web page visits. Our contribution are in two areas. First, we provide a constraint-based association rule mining algorithm for the web access log mining domain which improves the efficiency of rule set generation by moving item set selection further back in the computations. Second, we explain a method for coupling the association rule set generation with visualization to help user understand the navigational traffic within their web connectivity context. In what follows, we first clarify how constraint-based mining works with our web connectivity data, study two types of constraints and elaborate on how they can be considered during the mining of frequent itemsets, then introduce our constraint-based algorithm.

After that, we show some experimental results on constraint-based mining, and explain our approach for association rule in-context visualization before concluding.

2 Constraint-Based Mining

It is known that algorithms for discovering association rules typically generate an overwhelming number of those rules. While many more efficient algorithms have been recently proposed for the mining of extremely large datasets, the problem of large number of rules remains. The set of discovered rules is often so large that they have no value in identifying trends in the data from which they were mined. Various measures of interestingness and filters have been proposed to reduce the volume of discovered rules, but one of the most realistic ways to find only those interesting patterns is to express constraints on the rules we want to discover. However, filtering the rules post-mining adds a significant overhead and misses the opportunity to reduce the search space using the constraints. Application of constraints should be done as early as possible during the mining process, a procedure known as “pushing constraints”.

2.1 Association Rules on Web Page Connectivity

The problem of mining association rules consists of finding associations between items or itemsets in transactional data. For web access data, click-stream visitation is modeled by sets of transactions. An *association rule* is an implication of the form "$X \Rightarrow Y$", where X and Y are disjoint sets of items (i.e. web pages). The rule $X \Rightarrow Y$ has *support s* if $s\%$ of the transactions contains $X \cup Y$, and holds with a *confidence c* if $c\%$ of the transactions that support X also support Y. The main and most expensive component in mining association rules is the mining of frequent itemsets. Formally, the problem is stated as follows: Let $I = \{i_1, i_2, ...i_m\}$ be a set of literals, called items. In our case each item is a web page or web resource. Each such item is an object with some predefined attributes such as size, duration of visitation, time, etc. and m is considered the dimensionality of the problem. Let $\mathcal{D}$ be a set of transactions, where each transaction T is a set of items such that $T \subseteq I$. A transaction T is said to contain X, a set of items in I, if $X \subseteq T$. A constraint ζ is a predicate on itemset X that yields either *true* or *false*. An itemset X satisfies a constraint ζ if and only if $\zeta(X)$ is *true*. Constraints are imposed on the predefined attributes describing the items. An itemset X has a *support s* in the transaction set $\mathcal{D}$ if $s\%$ of the transactions in $\mathcal{D}$ contain X. Two particular constraints pertain to the support of an itemset, namely the *minimum support* and the *maximum support* constraints. An itemset X is said to be *infrequent* if its *support* s is smaller than a given minimum support threshold σ; X is said to be *too frequent* if its *support* s is greater than a given maximum support Σ; and X is said to be *frequent* if its *support* s is greater or equal than σ and less or equal than Σ.

2.2 Categories of Constraints

A variety of constraints have been identified in the literature [9]. We discuss two important categories – *monotone* and *anti-monotone*.

Definition 1 (*Anti-monotone constraints*): A constraint ζ is *anti-monotone* if and only if an itemset X violates ζ, so does any superset of X. That is, if ζ holds for an itemset S then it holds for any subset of S.

The minimum support threshold is a typical *anti-monotone* constraint. Also, as an example, $sum(S) \leq v$ $(\forall a \in S, a \geq 0)$ is an *anti-monotone* constraint, where v is a value. Assume that web-pages A, B, and C have an average visit time of 10, 15, and 5 minutes respectively. Given the constraint $\zeta = (sum(S) \leq 20\ minutes)$, then since the page set AB, with a total average visit time of 25 minutes violates ζ, there is no need to test any of its supersets (e.g. ABC) as they *de facto* also violate the constraint ζ.

Definition 2 (*Monotone constraints*): A constraint ζ is *monotone* if and only if an itemset X holds for ζ, so does any superset of X. That is, if ζ is violated for an itemset S then it is violated for any subset of S.

The maximum support threshold is a typical monotone constraint. Another example of a *monotone* constraint is $sum(S) \geq v$ $(\forall a \in S, a \geq 0)$. Using the same web-pages

A, B, and C as before, and with constraint ζ = ($sum(S) \geq 35\ minutes$), then knowing that ABC violates the constraint ζ is sufficient to know that all subsets of ABC will violate ζ as well.

2.3 Constraining Connectivity Rule Solutions

Most proposed algorithms that handle constraints during the frequent itemset mining process either deal with only *anti-monotone* constraints, in the case of bottom-up approaches, or only *monotone* constraints, in the case of top-down approaches. FIC^M [10], an FP-Growth based algorithm [8], generates most frequent patterns before pruning them. Its main contribution is that it checks for monotone constraints early, and once a frequent itemset is found to satisfy the monotone constraints, all subsequent itemsets having this item as a prefix are sure to satisfy the constraint so there is no need to apply further checks. The only known algorithm that can consider *monotone* and *anti-monotone* constraints simultaneously is DualMiner[4]. Nonetheless, it suffers from many practical limitations and performance issues. First, it is built on top of the MAFIA [5] algorithm, which produces the set of maximal patterns without considering retaining sufficient information about item supports, and consequently all frequent patterns generated using this model do not have their support attached. Second, it assumes that the whole dataset can fit in main memory which is not always the case. Third, their top-down computation exploiting the monotone constraint often performs many unnecessary tests for relatively large datasets. In a recent study of parallelizing DualMiner [11], the authors showed that by mining relatively small sparse datasets consisting of 10K transactions and 100K items, the sequential version of DualMiner took an excessive amount of time. A recent strategy dealing with monotone and anti-monotone constraints suggests reducing the transactional database input via pre-processing by successively eliminating transactions that violate the constraints and then applying any frequent itemset mining algorithm on the reduced transaction set [1,3]. The main drawback of this approach is that it is highly I/O bound due to the iterative process needed in re-writing the reduced dataset to disk. This algorithm is also sensitive to the results of the initial monotone constraint checking which is applied to full transactions. In other words, if a whole transaction satisfies the monotone constraint, then no pruning is applied and consequently no gains are achieved, even if parts of this transaction do not satisfy the same monotone constraint. To overcome some of the issues in [1], the same approach has been tested against the FP-Growth approach in [2] with new effective pruning heuristics. The major contender remains DualMiner. However, as mentioned, DualMiner has significant practical limitations due to its modest performance and lack of frequency calculation for all items. Efficiently finding the support for any itemset (even for ones that do not satisfy the constraints) is critical for generating association rules. For example, if itemset ABC satisfies the *monotone* constraints then there is a possibility that itemset AB would not satisfy those same constraints. In such cases most algorithms would not even generate and test them. However, when generating association rules, the support of AB is critical for finding the association rules in which ABC is involved.

Our algorithm is capable of bidirectional pushing of constraints and can consider both types of constraints during the mining process. It also finds the support for any itemset efficiently using a special data structure that encodes the supports of all frequent patterns.

3 Our Approach

Most existing algorithms traverse the itemset lattice top-down or bottom-up, and search using a depth first or breadth first strategy. In contrast, we propose a *leap traversal* strategy that finds pertinent maximal itemsets (i.e. frequent itemsets that are not subsumed by any other frequent itemset) by "leaping" between promising nodes in the itemset lattice. In addition to finding these relevant candidate itemsets, sufficient information is gathered to produce the frequent itemset patterns, along with their support values. Here, we use leap traversal in conjunction with the frequent co-occurrence idea (COFI-trees) introduced by [7], where the locally frequent itemsets of each frequent item are explored separately. This creates additional opportunities for pruning. We will first present the COFI tree structure, then we will introduce our algorithm which exploits the COFI trees and jumps in the pattern lattice. In the next section, this same algorithm will be enhanced with constraint checking to produce our final algorithm.

3.1 COFI-Trees

El-Hajj et al. introduced the COFI-tree idea [7] as a means to reduce memory requirements and speed up the mining strategy of FP-growth [8]. Rather than recursively building conditional trees (i.e. FP-trees conditioned on the existence of a given frequent item) from the global FP-tree, the COFI strategy was to create COFI-trees for each frequent item and mine them separately. The FP-tree [8] is a compact prefix-tree representation of the sub-transactions in the input data. Here, sub-transactions are the original transactions after infrequent items removal. The FP-tree contains a header and inter-node links which facilitate downward traversal (forward in the itemset pattern) as well as lateral traversal (next node representing a specific item).

Building COFI-trees based on the FP-tree: For each frequent item in the FP-tree, in order of increasing support, one COFI-tree is built [7]. This tree is based on sub-transactions which contain the root item and are composed only of items locally frequent with the root item that have not already been used as root items in earlier COFI-trees. The COFI-tree is similar to the FP-tree, but includes extra links for upward traversal (earlier in the itemset pattern), a data structure to allow traversal of all leaves in the tree, and an additional counter in each node. This counter is used during the mining process to count the participation of each node in the generation of a frequent pattern.

3.2 Taking Advantage of COFI-Trees

Our algorithm is different from the algorithm presented in [7] in the sense that it generates maximal patterns, where a pattern is said to be maximal if there is no other frequent pattern that subsumes it. Rather than traversing the pattern lattice top-down our approach leaps from one node to the other in search of the support border where maximals sit. Once maximals are found, with the extra information collected, all other patterns can be generated with their respective support. A brief summary of the algorithm follows. First, a frequent pattern FP-tree [8] is created, using two scans of the database. Second, for each frequent item, a COFI-tree is created including all co-occurant frequent items to the right (i.e. in order of decreasing support). Each COFI-tree is generated from the FP-tree without returning to the transactional database for scans. Unique sub-transactions in the COFI-tree along with their count (called *branch support*) are obtained from the COFI-tree. These unique sub-transactions are called *frequent path bases* (FPB), and can be obtained by traversing upward from each leaf node in the COFI-tree, updating the participation counter to avoid over-counting nodes. Clearly, there is at most one FPB for each sub-transaction in the COFI-tree. Frequent FPBs are declared candidate maximals. Infrequent FPBs are intersected iteratively, producing subsets which may be frequent. When an intersection of infrequent FPBs results in a frequent itemset, that itemset is declared as a candidate maximal and is not subject to further intersections. When an intersection is infrequent, it participates in further intersections looking for maximals. This is indeed how the leaps in the lattice are done. The result of the intersection of FPBs indicates the next node to explore. Given the set of frequent path bases along with their branch supports, it is possible to count the support of any itemset. This is done by finding all FPBs which are supersets of the target itemset, and summing their branch supports.

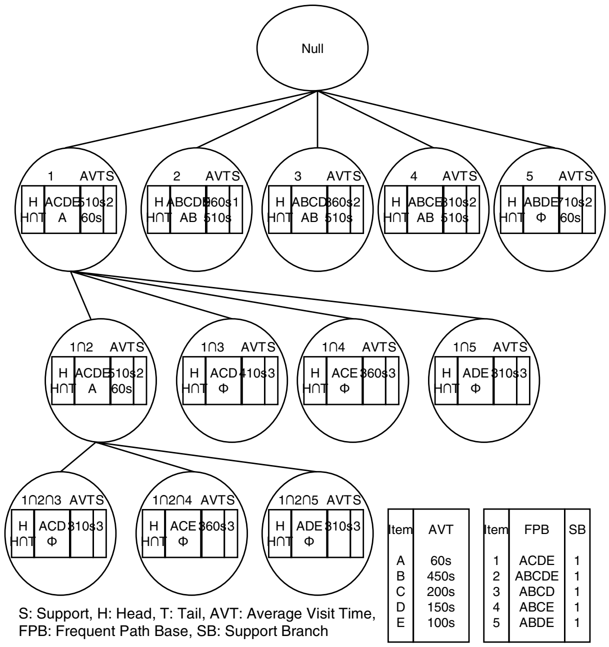
4 Pushing Constraints in Our Algorithm

We extended our approach by pushing constraints during the jumping process in such a way that if a tested node does not satisfy the *anti-monotone* constraints then we do not jump upward any more as all its supersets do not satisfy the same *anti-monotone* constraints. The same idea is applied for *monotone* constraints, but instead we do not jump downward because all the subsets of an itemset that do not satisfy a *monotone* constraint also do not satisfy it.

The conjunction of all *anti-monotone* constraints comprises a predicate that we call $P()$. A second predicate $Q()$ contains the conjunction of the *monotone* constraints. Pushing $P()$ and $Q()$ starts by defining two terms which are head (H) and tail (T) where H is a frequent path base or any subset generated from the intersection of frequent path bases, and T is the itemset generated from intersecting all remaining frequent path bases not used in the intersection of H. The intersection of H and T, $H \cap T$, is the smallest subset of H that may yet be considered. In this way, we focus on finding frequent H that can be declared as local maximals and candidate global maximals.

The algorithm is also capable of applying fewer predicate checks because if it detects that a candidate itemset satisfies the *anti-monotone* constraints, it does not test all its subsets as they all satisfy the constraints; the same idea is applied in reverse order for the *monotone* constraints. Four pruning strategies are used to reduce the intersections between nodes which are:

1. If an intersection of frequent path bases (H) fails $Q()$, then H can be discarded;
2. If an intersection of frequent path bases (H) passes $P()$, it is a candidate P-maximal (i.e. a local maximal that satisfies $P()$), and there is no need to further evaluate intersections with H;



Item	AVT
A	60s
B	450s
C	200s
D	150s
E	100s

Item	FPB	SB
1	ACDE	1
2	ABCDE	1
3	ABCD	1
4	ABCE	1
5	ABDE	1

Fig. 1. Pushing P() and Q()

3. If a node's $H \cap T$ fails $P()$, the H node can be discarded. There is no need to evaluate further intersections with H;
4. If a node's $H \cap T$ passes $Q()$, $Q()$ is guaranteed to pass for any itemset resulting from the intersection of a subset of the frequent path bases used to generate H plus the remaining frequent path bases yet to be intersected with H. $Q()$ does not need to be checked in these cases.

Ultimately, the generated set of maximals satisfy both types of constraints and no relevant maximal is missed. Generating all subsets from this set and then the relevant association rules is the last step in the mining process. Only $Q()$ needs to be checked whenever a frequent subset is generated from a maximal. If there is violation by a given itemset, no subsequent subsets of it are further generated. Algorithm 1 presents the steps needed to generate the frequent patterns while pushing constraints.

The following example, shown in Figure 1, illustrates how the algorithm works. An A-COFI-tree is made from five web pages, A, B, C, D, and E, with average visit time in seconds of 60, 450, 200, 150, and 100 respectively. In our example, this COFI-tree generates 5 Frequent-Path-Bases, $ACDE$, $ABCDE$, $ABCD$, $ABCE$, and $ABDE$, each with branch support one. The *anti-monotone* predicate, $P()$, is Sum (average visit time) $\leq$ 500 seconds, and the *monotone* predicate, $Q()$, is Sum (average visit time) $\geq$ 100. Intersecting the first FPB with the second produces $ACDE$ which has a average visit time of 510 seconds, and therefore violates $P()$ and passes $Q()$. Next, we examine the $H \cap T$, the intersection of this node with the remaining three FPBs, which yields A

with total average visit time equals to 60 seconds, passing $P()$ and failing $Q()$. None of these constraint checks provide an opportunity for pruning, so we continue intersecting this itemset with the remaining frequent path bases. The first intersection is with the third FPB, producing ACD with average visit time equals to 410 seconds, which satisfies both the *anti-monotone* and *monotone* constraints. The second intersection produces ACE, which also satisfies both constraints. The same thing occurs with the last intersection, which produces ADE. Going back to the second Frequent-Path-Base, ABCDE, we find that the $H \cap T$, AB, violates the *anti-monotone* constraint with average visit time equals to 510 seconds. Therefore, we do not need to consider $ABCDE$ or any further intersections with it. The remaining nodes are eliminated in the same manner. In total, three candidate P-maximals were discovered. We can generate all of their subsets while testing only against $Q()$. Finally, the support for these generated subsets can be computed from the existing frequent path bases.

5 Experiments in Constraining Associations

Our experiments use monthly server-produced access logs as web usage data. The data preparation includes crawling the website structure, cleaning irrelevant records, and breaking the access log into user sessions. In order to cut usage records into sessions, we identify users by their authentication or cookie value. The session timeout is arbitrarily set to be 30 minutes. Our interest in the log data are attributes that include page visit frequency, page average view time and link usage frequency. and session user IP address. Our experiments were conducted on a 3GHz Intel P4 with 2 GB of memory running Linux. The times reported also include the time to output all itemsets.

To illustrate the effectiveness of our approach, we report experiments applied on two data sets: the web requests for the month of March and the month of April. Our experiments are run with a minimum support of 1% and a minimum confidence of 90%. Using only these two thresholds, March and April generated respectively 10,985 and 13,543 association rules. It is obvious that visualizing all these rules for analysis is impractical. Constraints were later added for better focus. Figures 2 and 3 present the number of rules generated with and without some specific constraints. We first report

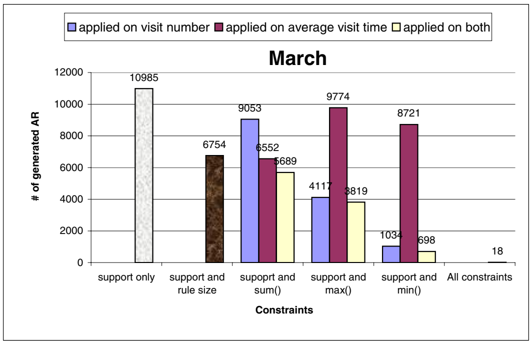


Fig. 2. Rules in March, Support = 1%, Confidence = 90%

Algorithm 1. Pushing P() and Q()

```
Input: $D$ (transactional database); $\sigma$; $\Sigma$; $P()$; $Q()$.
Output: Frequent patterns satisfying P(), Q()
$PF1 \leftarrow$ Scan $D$ to find the set of frequent P1-itemsets
$FPT \leftarrow$ Scan $D$ to build FP-tree using $PF1$ and $Q()$
$PGM(PGlobalMaximals) \leftarrow \emptyset$
for each item $I$ in Header($FPT$) do
    $LF \leftarrow FindlocalFrequentWithRespect(I)$
    Add $(I \cup LF)$ to $PGM$ and Break IF $(P(I \cup LF))$
    break IF (Not $Q(I \cup LF)$)
    if Not $(I \cup LF) \subseteq PGM$ then
        $ICT \leftarrow$ Build COFI-Tree for $I$
        $FPB \leftarrow$ FindFrequentPathBases($ICT$)
        $PLM(PLMaximals) \leftarrow \{P(FPB)$ and frequent$\}$
        $InFrequentFPB \leftarrow notFrequent(FPB)$
        for each pair $(A, B) \in InFrequentFPB$ do
            $header \leftarrow A \cap B$
            Add $header$ in $PLM$ and Break IF ($P(header)$ AND is frequent and not $\emptyset$)
            Delete $header$ and break IF (Not $Q(header)$)
            $tail \leftarrow$ Intersection(FPBs not in $header$)
            delete $header$ and break IF (Not $P(header \cap tail)$)
            Do not check for $Q()$ in any subset of $header$ IF ($Q(header \cap tail)$)
        end for
        for each pattern $P$ in $PLM$ do
            Add $P$ in $PGM$ IF (($P$ not subset of any $M \in PGM$)
        end for
    end if
end for
PQ-Patterns $\leftarrow$ GPatternsQ($FPB, PGM$)
Output PQ-Patterns
```

the reduction by rule size constraint. We forced the rule size to be between 3 and 6 web pages. This decreases the number of rules to 6,754 and 4,681 respectively for March and April. We further applied constraints on two web page attributes: number of visits per page and average visit time per page. To examine the selectivity of some constraints we separately applied the constraints $sum()$, $max()$ and $min()$ monotonically and anti-monotonically. For Figures 2 and 3 the constraints are as follows: the sum of average visit time for all nodes is set to be between 120 to 1000 seconds, and the summation of average number of visits for all pages in one rules is set between 1000 and 5000; the maximum visit time for one nodes is limited between 30 to 240 seconds and for average number of visits between 1000 and 2000; the minimum is set to be between 5 to 60 seconds for average visit time for one page and for the average number of visits between 10 and 200.

We compared our algorithm with DualMiner [4]. Based on its authors' recommendations, we built the DualMiner framework on top of the MAFIA [5] implementation provided by its original authors. We used the same set of experiments described above and found that in most cases our implementations while pushing both types of

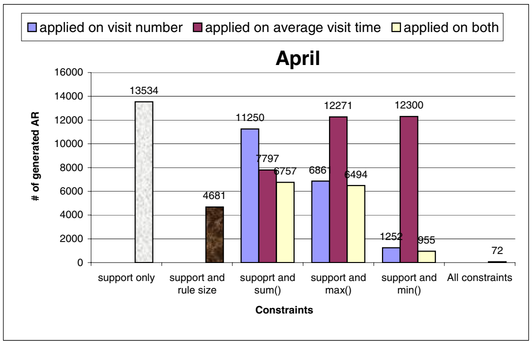


Fig. 3. Rules in April, Support = 1%, Confidence = 90%

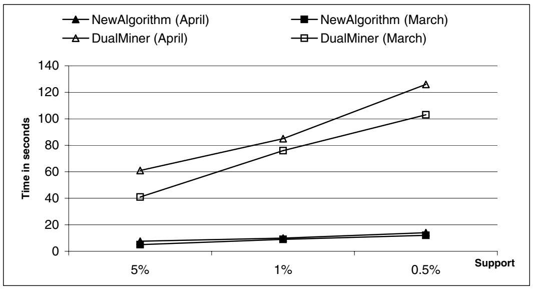


Fig. 4. Time for mining March and April datasets

constraints outperforms DualMiner. Figure 4 depicts the results of these experiments using three different support levels which are 0.5%, 1%, and 5%. The constraints cover rules that affect the average number and time of visits.

6 Association Rule Visualization

After discovering association rules using our constraint-based mining method, displaying those rules within the web page connectivity context becomes essential to take advantage of human visual capabilities in identifying valuable knowledge patterns. In this section, we present our visualization approach to analyze association rules in their structure context, by overlaying discovered patterns on the web connectivity graph.

We adopt the Disk-tree [6] representation to generate the web structure image, in which a node represents a page and an edge indicates the hyperlink that connects two pages; the root, a node located in the center of the graph, is a given starting page. A disk-tree is a concentric circular hierarchical layout. The nodes in the outer circles are children of the inner circle's nodes. This means that the corresponding pages of the outer nodes can be connected from that of the inner nodes. We use a Breadth First

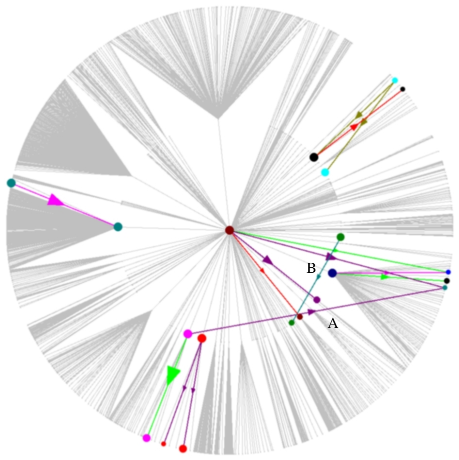


Fig. 5. Association Rule Visualization in Context

Search(BFS) method to convert the connected graph into a tree structure, by placing a node as closely to the root node as possible. For each page, only one incoming link can be represented and BFS picks the first link following the scan of the web topology.

Figure 5 is a screen shot of the association rule visualization in our web mining system. It shows rules whose $support > 0.2\%$ within two levels of the disk-tree in one month usage record of our department website. Association rules are overlaid on the web site topology structure context using visual cues such as edge colours and sized arrow heads. For each rule, one colour is used to draw both the antecedent and consequent nodes. The size of the node represents the visit frequency of the corresponding page. The colour of the edge indicates the confidence of the rule and the size of the arrow head represents the support of the rule. For rules that have multiple antecedent or consequent items, edges between any two pages are connected and display the same value. The visualization result can be used by web administrators to analyze the navigational behaviour and improve the site structure design whereever needed. For example, we can clearly see the fact that rule A and B jump across from one subtree of the web graph to another. This phenomena means that the majority of navigation sessions involving those pages did not follow the designed linking structure as expected, which indicate potential design problems and thus merit additional investigation. Note that the detailed information regarding edges, pages and properties of those association rules represented in the visualization can be displayed by the system upon request simply by moving the mouse over them. The static Figure 5 does not illustrate all those interactive features.

Compared with existing approaches to visualizing association rules, our method is superior in that we deal with visualization of association in their structural context, which is not only the domain context for the web site structure, but also the inner relationship and connection between a set of association rules.

7 Conclusions

Discovering trends and patterns in web navigation is undeniably advantageous to web designers and web-based application architects. Associations between web pages, their visits and connectivity articulated in terms of association rules are very useful web patterns. Visualizing the rules that express web page associations and their visits in the context of the web site structure is of major importance as it puts web page requests and their connectivity in perspective. Also, the massive number of association rules typically discovered prevents any practical analysis or interpretation. What is needed is a means to effectively and efficiently focus on relevant association rules and then visualize pertinent relations between pages visited. In this work, we essentially propose a pattern focusing method to reach a practical and reasonably sized set of interpretable association rules. The method concentrates on mining only the relevant frequent patterns by means of constraint-based mining. Other constrains specific to the form of the rule are also possible.

References

1. Bonchi, F., Giannotti, F., Mazzanti, A., Pedreschi, D.: Examiner: Optimized level-wise frequent pattern mining with monotone constraints. In: IEEE ICDM, Melbourne, Florida (November 2004)
2. Bonchi, F., Goethals, B.: Fp-bonsai: the art of growing and pruning small fp-trees. In: Dai, H., Srikant, R., Zhang, C. (eds.) PAKDD 2004. LNCS (LNAI), vol. 3056, pp. 155–160. Springer, Heidelberg (2004)
3. Bonchi, F., Lucchese, C.: On closed constrained frequent pattern mining. In: Perner, P. (ed.) ICDM 2004. LNCS (LNAI), vol. 3275, Springer, Heidelberg (2004)
4. Bucila, C., Gehrke, J., Kifer, D., White, W.: Dualminer: A dual-pruning algorithm for itemsets with constraints. In: Eight ACM SIGKDD Internationa Conf. on Knowledge Discovery and Data Mining, Edmonton, Alberta, pp. 42–51 (August 2002)
5. Burdick, D., Calimlim, M., Gehrke, J.: Mafia: A maximal frequent itemset algorithm for transactional databases. In: ICDE, pp. 443–452 (2001)
6. Chi, E.H., Pitkow, J., Mackinlay, J., Pirolli, P., Gossweiler, R., Card, S.K.: Visualizing the evolution of web ecologies. In: CHI 1998. Proceedings of the Conference on Human Factors in Computing Systems (1998)
7. El-Hajj, M., Zaïane, O.R.: Non recursive generation of frequent k-itemsets from frequent pattern tree representations. In: Kambayashi, Y., Mohania, M.K., Wöß, W. (eds.) DaWak 2003. LNCS, vol. 2737, Springer, Heidelberg (2003)
8. Han, J., Pei, J., Yin, Y.: Mining frequent patterns without candidate generation. In: Chen, W., Naughton, J., Bernstein, P.A. (eds.) ACM SIGMOD Intl. Conference on Management of Data, vol. 05, pp. 1–12. ACM Press, New York (2000)
9. Lakshmanan, L.V., Ng, R., Han, J., Pang, A.: Optimization of constrained frequent set queries with 2-variable constraints. In: ACM SIGMOD Conference on Management of Data, pp. 157–168 (1999)
10. Pei, J., Han, J., Lakshmanan, L.V.: Mining frequent itemsets with convertible constraints. In: IEEE ICDE Conference, pp. 433–442 (2001)
11. Ting, R.M.H., Bailey, J., Ramamohanarao, K.: Paradualminer: An efficient parallel implementation of the dualminer algorithm. In: Dai, H., Srikant, R., Zhang, C. (eds.) PAKDD 2004. LNCS (LNAI), vol. 3056, pp. 96–105. Springer, Heidelberg (2004)

Does Multi-user Document Classification Really Help Knowledge Management?

Byeong Ho Kang, Yang Sok Kim, and Young Ju Choi

School of Computing, University of Tasmania, Sandy Bay,
7005 Tasmania, Australia
{bhkang, yangsokk}@utas.edu.au

Abstract. In general, document classification research focuses on the automated placement of unseen documents into pre-defined categories. This is regarded as one core technical component of knowledge management systems, because it can support to handle explicit knowledge more systematically and improve knowledge sharing among the users. Document classification in knowledge management systems should support incremental knowledge acquisition and maintenance because of the dynamic knowledge changes involved. We propose the MCRDR document classifier as an incremental and maintainable document classification solution. Even though our system successfully supported personal level document classification, we did not examine its capability as a document classification tool in multi-user based knowledge management contexts. This paper focuses on the analysis of document classification results performed by multiple users. Our analysis reveals that even though the same documents and the classification structure are given to the users, they have very different document classification patterns and different acceptance results for each other's classification results. Furthermore, our results show that the integration of multiple users' classification may improve document classification performance in the knowledge management context.

Keywords: Knowledge Management, Document Classification, MCRDR.

1 Introduction

Knowledge management systems have been developed as a critical component on many organizations because knowledge is regarded as an organizational competency. Knowledge can be classified into tacit knowledge and explicit knowledge [1]. In this paper, we focus on the latter, which is usually represented by documents such as reports, papers, and e-mails. Document classification systems are regarded one of main sub-systems of explicit knowledge management systems, because they automatically place unseen documents into predefined categories using classification knowledge [2].

Typical document classification research focuses on the performance of the automated classification results such as accuracy, precision, and recall[3]. However, it is required for document classification systems to support the following requirements within the knowledge management context. Firstly, document classification systems should support incremental knowledge acquisition and maintenance, because knowledge management is not a finite but an infinite process [4]. As knowledge management is dynamic over time, so is document classification. Changes come from document content and human factors. Documents' contents continually change with time. For example, news articles on information technology in 2007 significantly differ from those of 1990. Therefore, it is hard to expect the classifier developed in 1990 to exhibit a similar performance as that developed in 2007. Changes also come from human factors. On one hand, people usually learn domain knowledge while they perform document classification. Therefore, they gradually change classification rules and modify classification structures. On the other hand, the person classifying documents may change with the passage of the time. As the person changes, so does the classification knowledge as classification knowledge is dependent on the person who classifies the documents.

Secondly, considerable portions of one user's document classification results might be accepted by the other users. When we refer to a 'user', this does not only mean a human user, but may also embrace a computerized system. In case of a human, the classification creator may be an ordinary user or an expert person such as cataloguer in a library. On the other hands, computerized systems can be implemented by various supervised learning or unsupervised learning methods. In any case considerable level of acceptance of other's document classification results is the critical factor of document classification system in knowledge management context. Though this requirement is important, little research has been conducted. For this reason, we undertook an extensive user study on the classification acceptance during our research.

Our current research is based on our prior document classification research [5-7], where we proposed a document classification system based on the MCRDR (Multiple Classification Ripple-Down Rules) knowledge acquisition method. The focus of our system was mainly on the first issue, incremental acquisition and maintenance of knowledge. We already reported evaluation results of our system as a personal level document classification. However, we did not thoroughly research the second issue, namely the acceptance of classification results. In this paper, we conducted an interpersonal acceptance experiment of document classification results to prompt further study direction in relation to document classification in knowledge management systems.

This paper consists of the following contents. Section 2 briefly explains systems used in this research, including Web monitoring system, MCRDR document classification system, and classification evaluation system. Section 3 describes our experiment methods, which include participant description, document classification experiment process, and classification acceptance evaluation process. Section 4 summarizes participants' classification results – document usage and knowledge acquisition results. Section 5 demonstrates the classification acceptance results, including document filtering results, classification acceptance results, actual filtering results and evaluation acceptance comparison results. Section 6 describes relevant prior research, while Section 7 contains our conclusions.

2 Systems

For this research, the following systems were deployed:

- **Web Information Monitoring System (WebMon):** This program collects newly uploaded information from the registered Web pages. It compares two URL lists (URL_{old} and URL_{new}) that are extracted from the old web pages and the current web pages. If there are any new URLs in URL_{new}, they are reported as new information. More detailed explanation on the WebMon is in [6].
- **MCRDR Document Classification System (MCRDR Classifier):** This system employs an incremental knowledge acquisition method, called MCRDR (Multiple Classification Ripple-Down Rules) and helps users to classify documents with classification rules. Cornerstone, rule path, and localized validation help general or expert users to gain easy knowledge acquisition and maintenance. More detailed explanation on the MCRDR Classifier is in [6].
- **Document Classification Evaluation System (DCES):** This system supports interpersonal evaluation of the document classification results. It displays randomly sampled classification results and each user can evaluate whether they are acceptable or not. The following section explains more details of this system.

3 Experiment Methods

Participants. There were two different participant groups in this experiment – students and experts. The students comprised of ten masters or honors course students at School of Computing, University of Tasmania who had no experience in classifying government information. The experts were three cataloguers from the Tasmanian State Library, and were regarded as experts in the classification of government information, because they maintained Australian government information in their web sites, entitled ‘Service of Tasmania’ and ‘Tasmania Government Online’. We included these experts to determine if there was any difference between general users and experts in the document classification acceptance experiment.

Document Classification. For this experiment, the WebMon system collected newly uploaded web documents from 162 Australian Government homepages and media release pages and forwarded them to the each participant’s MCRDR classifier. These documents were automatically evaluated by each participant’s classification rules. Each participant created new rules to classify unclassified documents or to reclassify misclassified documents. Hierarchical classification structure, which has three levels and 769 categories, was given to the participants. Only student participants were involved in the document classification process. They had no experiences in government information classification and acquired classification knowledge and classification structure while they conducted classification knowledge base. For this reason, they incrementally constructed their classification knowledge base, even though the classification structure was given to them at the beginning of experiment.

In the trial phase, the MCRDR classifier was provided to the participants for two weeks before main classification phase to give them some experience of the classifier. Participants could use the classifier as the same conditions with the main classification phase. After resetting the trial period knowledge base, classification of the monitored documents was undertaken for eight weeks from 21 August, 2006, to 9 October, 2006. A total of 2,567 documents were collected during the experiment period. On average 46 documents were presented to the participants each day. At this stage, we did not give any information about the following acceptance experiment, because the participants could be contaminated by it when they classified the monitored documents.

Classification Acceptance Evaluation. For this experiment, 100 documents were randomly sampled from the collected documents. Each participant was given a sample set. If a participant started the evaluation process, the DCES displayed a list of sampled documents. When the participant selected a document from this list, the DCES posed the following questions.

- The first question asked the participant to decide whether he/she wanted to classify the document or not. This question evaluated document filtering differences among participants. If the participant chose to classify, the following question was asked. Otherwise, he/she did not need to answer this question.
- In the second question, the participant was requested to evaluate other users' classification results. The evaluation interface only displayed the category path from root node to the classified node without displaying classification rule conditions (e.g., Root > Our Government > Law and Public Safety > Police and Protective Services). If the participant did not accept the classification results, he/she was required to select one reason from the following three reasons:
  (1) the current document should be classified into one of the child categories;
  (2) the current document should be classified into the parent category; or
  (3) the current document should be classified into elsewhere, other than the child categories or the parent category.

4 Document Classification Results

4.1 Document Usage Results

The document usage results of the participants are illustrated in Fig. 1. The lowest document usage frequency was 1,388 (54% of all monitored documents), while the highest frequency was 2,155 (84% of all monitored documents). The mean average document usage was 1,841. Significant variations among the participants in document classifications were also noted. The lowest document classification frequency was 1,621 (1.2 classifications per document) with the highest frequency being 7,016 (3.4 classification per document). The mean average classifications were 4,307 and mean average classification per document, 2.2. Detailed document statistics are summarized in Fig. 2.

Typical document classification research does not include this phase or assumes that documents are already pre-processed before the classifiers conduct classification.

However, document classification in the knowledge management context usually requires selecting useful documents by the users, and our results show that document usage varies among them. The mechanism that supports document usage collaboration among users may improve document classification in the knowledge management context.

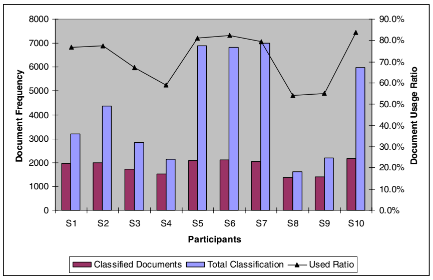


Fig. 1. Document Usage

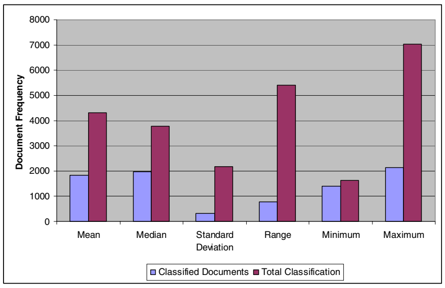


Fig. 2. Document Usage by Participants

4.2 Knowledge Acquisition

Rule Results. In the MCRDR classifier, KA is performed by creating new rules with a case and its process consists of condition selection and conclusion selection. There are two types of rules in the MCRDR classifier – refine rule and stop rule. Fig. 3 illustrates each participant's rule count and document classification count, where the rule count is a sum of the refine and stop rule count. The results show the following. Firstly, the total number of rule creation varied greatly among participants. While the mean average number was 434.8, the maximum number was 668 and minimum was 362. Therefore, the number range was 306 (see Fig 4). This implies that knowledge acquisition was very different among participants. Secondly, the combination of refine rules and stop rules was significantly different among participants. Whereas some users created a large number of stop rules, other users created few stop rules (see Fig 3). Lastly, the number of refine rules and the number of stop rules showed significant correlation with the number of document classifications. The correlation coefficient is -0.63 between the number of document classification and the refine rules and 0.66 between the number of document classification and stop rules.

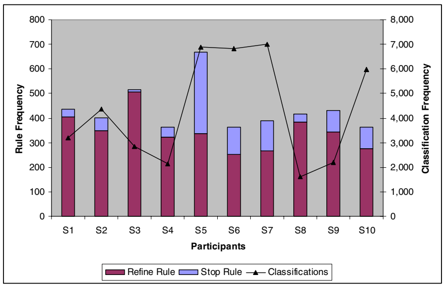


Fig. 3. Rule and Classification

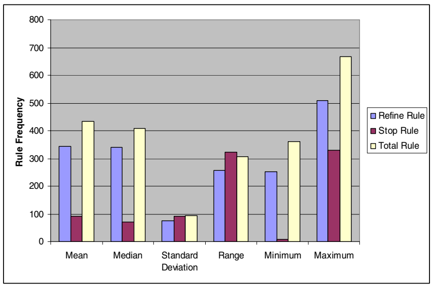


Fig. 4. Rule Statistics

Category Results. When the participants created a rule with the MCRDR classifier, they needed to select one or more categories into which the current handing document was classified. The participants did not know the whole classification structure from the beginning, rather they gradually came to understand it while performing classification task. Therefore, it may be expected that the resulting category usage may differ among the participants. Fig. 5 illustrates that this expectation is correct, where each participant's category usage results are displayed with the classification count. On average, 168.9 categories (22% of 769 categories) were used by participants, with the maximum usage count is 262 and the minimum 127. More detailed statistics on category usage are illustrated in Fig. 6. Each participant's category usage increases as the number of refine rules increase and decreases as the number of stop rules increase. The correlation coefficient is 0.77 between category usage and the refine rule frequency and - 0.57 between category usage and the stop rule frequency.

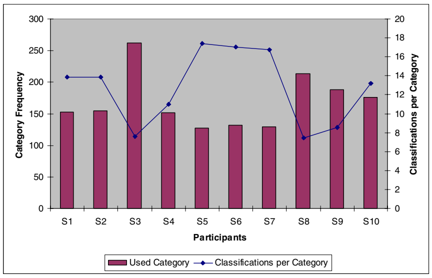


Fig. 5. Category Usage

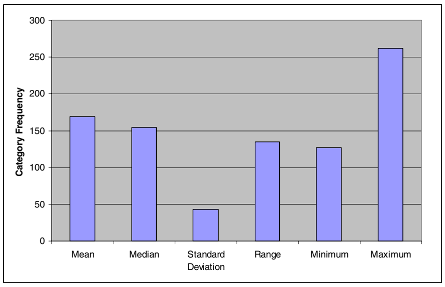


Fig. 6. Category Usage Statistics

Condition Words Usage Results. In the MCRDR classifier, the participants needed to choose condition words from the document feature, which totally depends on the user's decision. Therefore, condition usage reflects each user's classification knowledge. Fig. 7 summaries each participant's average condition words count per rule. On average 2.36 words were used for one rule, with the maximum condition words count was 3.33 words and the minimum was 1.65 words. Detailed statistics on the condition word usage are illustrated in Fig. 8.

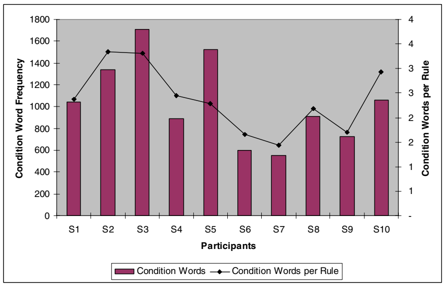


Fig. 7. Condition Words Usage

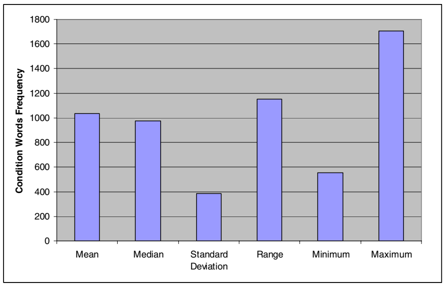


Fig. 8. Condition Words Usage Statistics

5 Classification Acceptance

Filtering Results. At first, we analyzed filtering differences among the participants. Fig. 9 illustrates each participant's filtering rates with actual document filtering rates (1-document usage rate). Firstly, the results show that each participant filtered out irrelevant documents with very different level. For example, whereas some participants wanted to classify all documents (100%), others participants filtered out 77% documents from given data set. Secondly, the results show that filtering results were significantly different between students and experts. The extent of this difference is clearly illustrated in Fig. 10. It is shown that the experts filtered out 73.3% of the evaluation documents and the students did only 28.6% (see Fig. 10). Furthermore, the results show that the variations between the experts are smaller than those of the students. Standard deviation of each group is 0.04 (experts) and 0.25 (students).

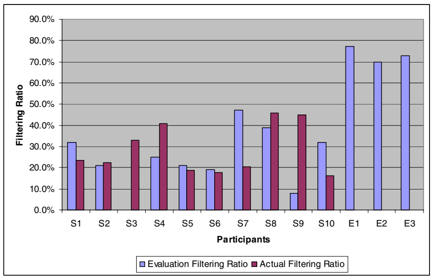


Fig. 9. Document Filtering

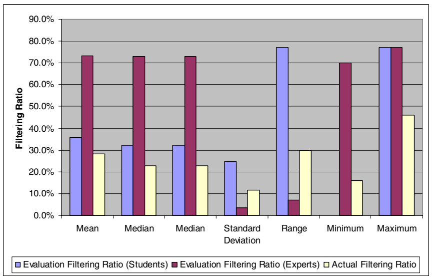
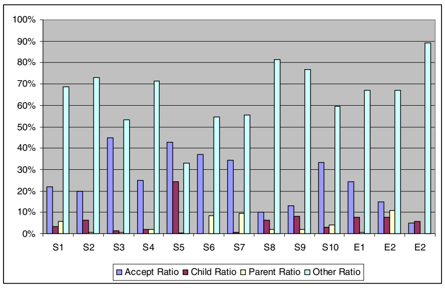


Fig. 10. Document Filtering Comparison

Classification Acceptance. As illustrated in Fig. 11, classification acceptance rates varied considerably among participants. Whereas students more positively accepted other participants' classification results than their expert counterparts, their acceptance rate range was greater than that of the experts (See Fig 12). On mean average, the students' acceptance rate was 28% and experts' rate only 15%. The student's acceptance rate ranged from 10% to 45%, while that of experts was 5% to 24% (See Fig. 12). Fig. 11 also illustrates reasons why the participants do not accept the classification results. Participants were able to select the following three reject reasons – reclassification into child category, reclassification into parent category and reclassification any other categories except child and parent category. Overall, the results show that the portion of reclassification into the child or parent category was smaller than 'other' category in both student and expert results, but each participant had a different reason for their rejection. The portion for child and parent category varied slightly between the two groups. Whereas the exports reported 8% of documents should be classified into child category and 5% into parent category, the students opted for 7% (child category) and 8% (parent category).



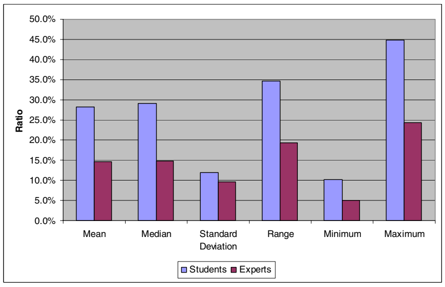
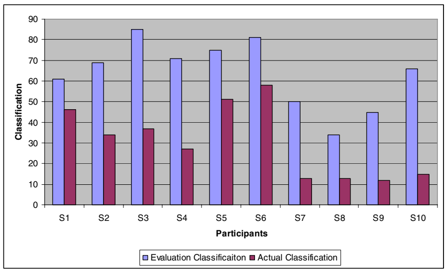


Fig. 11. Classification Accept/Reject

Fig. 12. Accept/Reject Statistics

Actual Classification vs. Evaluation Acceptance. Section 4 results show that each participant has significantly different classification patterns and Section 5 shows that participants accept other participant's classification results very differently. Now we address how much the classification performance improves when one participant considers other participant's classification results. To this end, we compared evaluation classification results with actual classification results. That is, if one or more classification results of a document are accepted by a participant, we examined whether the document is classified or not by using the participant's current classification knowledge and how many categories it is classified into. Whereas Fig. 13 compares the document number, Fig. 14 compares the category numbers. The results show that the participants consistently classify more documents and use more categories than those of their actual classification results. On average, they classified 2.1 times documents of their own classified documents and used 3.4 times categories of their own classified categories. These results imply that if classification results are shared among users, it can improve classification performance significantly.



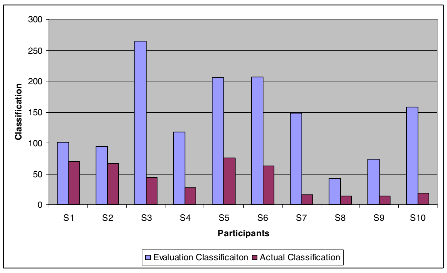


Fig. 13. Actual Classification vs. Evaluation Classification (Article)

Fig. 14. Actual Classification vs. Evaluation Classification (Category)

Relationship Analysis. We analyzed which measures are related to the document filtering and classification acceptance results. To this end, the correlation coefficient values were calculated between these two values and other performance values such as document classification results, document usage, classification rules, categories, condition words. The results we obtained are illustrated in Fig. 15 and

Fig. 16. These figures demonstrate that the filtering ratio closely related to the refine rule count, used category count, condition word count, and acceptance ratio. The acceptance ratio closely resembles that of the filtering ratio, document classification counts, and usage ratio.

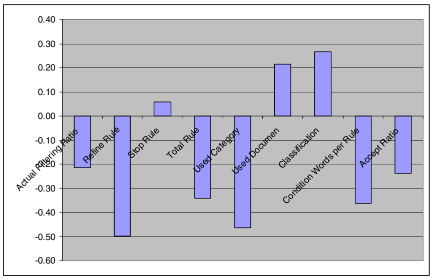


Fig. 15. Evaluation Filtering Relationship with Other Measures

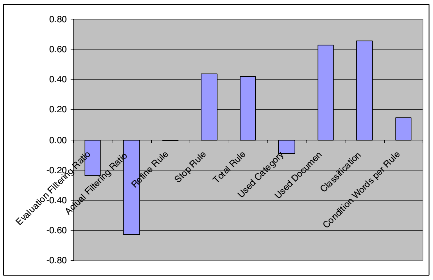


Fig. 16. Acceptance Relationship with Other Measures

6 Related Work

The relationship between knowledge management and document classification is discussed in [2, 4]. Our research indicates document classification is a core technology that enables explicit knowledge management. Our document classification employed MCRDR methodology. The philosophy and method of MCRDR knowledge acquisition are discussed in [8-10]. Incremental knowledge acquisition and maintenance by the domain user and easy localized validation and verification are core benefits of this method. Implementation details and performance evaluation of our web monitoring system and document classification system are found in [5, 6]. Some researchers are undertaking multiple users-based knowledge acquisition methods. They try to extend the MCRDR to encompass collaborative classification, classification merging, and classifications re-use [11]. However, there is no empirical study on classification acceptance.

7 Conclusions

Our experiment demonstrated the following results. Firstly, the participants chose different document usage and document classification results, including rule creation, category usage, and condition selection. These results are consistent with our prior research results. Secondly, the classification acceptance experiment results show that document filtering results and classification acceptance results also significantly differ among participants. Lastly, the experimental results show that the classification performance and reusability can be improved when document classification performed by multiple users, and their classification results, are shared by all participants. These results have encouraged us to undertake further study into classification results sharing method, which can be performed by one knowledge base strategy or multiple distributed knowledge based strategies. Whereas the first approach tries to create a knowledge base that integrates multiple users' knowledge base, the latter approach does not try to construct any new knowledge base, but each user can extend his/her own knowledge base by accepting other user's classification knowledge.

Acknowledgement. This work is supported by the Asian Office of Aerospace Research and Development (AOARD).

References

1. Polanyi, M.: The Tacit Dimension. Routledge & Kegan Paul, London (1996)
2. Marwick, A.D.: Knowledge Management Technology. IBM Systems Journal 40(4), 814–830 (2001)
3. Sebastiani, F.: Text Categorization in Text Mining and its Applications. In: Zanasi, A. (ed.) Text Mining and its Applications, A, pp. 109–129. WIT Press, Zanasi (2004)
4. Brucher, H., Knolmayer, G., Mittermayer, M.A.: Document Classification Methods for Organizing Explicit Knowledge. In: Proceedings of the Third European Conference on Organizational Knowledge, Learning, and Capabilities, Athens, Greece (2002)
5. Park, S.S., Kim, Y.S., Kang, B.H.: Web Document Classification: Managing Context Change. In: WWW/Internet 2004. Proceedings of the IADIS International Conference, Madrid, Spain, pp. 143–151 (2004)
6. Park, S.S., Kim, S.K., Kang, B.H.: Web Information Management System: Personalization and Generalization. In: WWW/Internet 2003. Proceedings of the IADIS International Conference, Algarve, Portugal, pp. 523–530 (2003)
7. Kim, Y.S., Park, S.S., Kang, B.H., Choi, Y.J.: Incremental Knowledge Management of Web Community Groups on Web Portals. In: Proceedings of the 5th International Conference on Practical Aspects of Knowledge Management. Vienna, Austria, pp. 198–207 (2004)
8. Compton, P., Edwards, G., Kang, B., Lazarus, L., Malor, R., Menzies, T., Preston, P., Srinivasan, A., Sammut, C.: Ripple down rules: possibilities and limitations. In: Proceedings of the 6th Banff AAAI Knowledge Acquisition for Knowledge Based Systems Workshop. Banff, Canada, pp. 6.1–6.20 (1991)
9. Compton, P., Jansen, R.: A Philosophical Basis for Knowledge Acquisition. Knowledge Acquisition 2(3), 241–258 (1990)
10. Kang, B., Compton, P., Preston, P.: Multiple Classification Ripple Down Rules: Evaluation and Possibilities. In: Proceedings of the 9th AAAI-Sponsored Banff Knowledge Acquisition for Knowledge-Based Systems Workshop. Banff, Canada, pp. 17.1–17.20 (1995)
11. Vazey, M., Richards, D.: Troubleshooting at the Call Centre: A Knowledge-based Approach. In: Proceedings of the Artificial Intelligence and Applications 2005, Innsbruck, Austria (2005)

Not All Words Are Created Equal: Extracting Semantic Orientation as a Function of Adjective Relevance

Kimberly Voll[1] and Maite Taboada[2]

[1] University of British Columbia, Vancouver BC, Canada
kvoll@cs.ubc.ca
[2] Simon Fraser University, Burnaby BC, Canada
mtaboada@sfu.ca

Abstract. Semantic orientation (SO) for texts is often determined on the basis of the positive or negative polarity, or sentiment, found in the text. Polarity is typically extracted using the positive and negative words in the text, with a particular focus on adjectives, since they convey a high degree of opinion. Not all adjectives are created equal, however. Adjectives found in certain parts of the text, and adjectives that refer to particular aspects of what is being evaluated have more significance for the overall sentiment of the text. To capitalize upon this, we weigh adjectives according to their relevance and create three measures of SO: a baseline SO using all adjectives (no restriction); SO using adjectives found in on-topic sentences as determined by a decision-tree classifier; and SO using adjectives in the nuclei of sentences extracted from a high-level discourse parse of the text. In both cases of restricting adjectives based on relevance, performance is comparable to current results in automated SO extraction. Improvements in the decision classifier and discourse parser will likely cause this result to surpass current benchmarks.

1 Introduction

As a measure of understanding text, semantic orientation (SO) is the attitude of a particular text toward a given subject. This can be broken down into the evaluative factor (either positive or negative) and the potency (the degree to which the document is positive or negative) [1]. When used in the analysis of public opinion, such as the automated interpretation of online product reviews, semantic orientation can be extremely helpful in marketing, measures of popularity and success, and compiling reviews. Furthermore, SO can provide a means for quantifying an attitudinal dimension, useful as a feature in machine learning.

In general, the problem of determining the overall SO of a document is broken down into first determining the essential units of the document, and then analyzing those units for sentiment. We need to determine, then, what the essential units are, and how we can measure their impact on the overall SO of a document. In this paper, we rely on adjectives as the essential units, and hypothesize that the efficacy of the adjectives used in determining SO is affected by the relevance of those adjectives in the text. We motivate and present two different approaches to their extraction on the basis of relevance: One approach extracts on-topic sentences, and relies only on the adjectives found within those. The other approach extracts the main parts of the text (nuclei, as defined within a theory of discourse structure), and also uses only the adjectives found in the nuclei. We compare the success of these methods, including a baseline analysis using all adjectives, and discuss future work.

2 Background

Much of the previous research in extracting semantic orientation has focused on adjectives as the primary source of subjective content in a document [2,3,4,5,6]. In general, the SO of an entire document is the combined effect of the adjectives found within, based upon a dictionary of adjective rankings (scores). The dictionary can be created in different ways: manually, using existing dictionaries such as the General Inquirer [7], or semi-automatically, making use of resources like WordNet [8]. More frequently, the dictionary is produced automatically via association, where the score for each new adjective is calculated using the frequent proximity of that adjective with respect to one or more seed words. Seed words are a small set of words with strong negative or positive associations, such as *excellent* or *abysmal*. In principle, a positive adjective should occur more frequently alongside the positive seed words, and thus will obtain a positive score, while negative adjectives will occur most often alongside negative seed words, thus obtaining a negative score. The association is usually calculated following Turney's method for computing mutual information [3,4].

It is obvious that words other than adjectives play a role in conveying sentiment, such as verbs (*hate, love*); adverbs (*poorly, correctly*); nouns (*disaster, hit*). In addition, certain words change the polarity of the word they accompany, including negative words (*not, no*) and intensifiers and diminishers (*extremely, barely*). We are assuming, for the time being, that we can extract an adequate measure of sentiment by relying exclusively on adjectives. Further improvements would involve taking into account other parts of speech, and phrasal units, not just individual words.

The challenge, however, is in not simply determining which class of words is essential to a document's SO, but also whether all words in a particular class (e.g., adjectives) are equally relevant. One need only consider a few documents before it becomes apparent that not all adjectives are used equally. For example, an adjective may have a greater or lesser impact on the overall sentiment of a document depending upon its location within the document. Taboada and Grieve [9] found that assigning weights to adjectives based upon their position increased the performance of automatic SO extraction. The assumption was that adjectives at the beginning of the text are not as relevant, and that an opinion on the main topic tends to be found towards the end of the text.

In addition, it is not the case that positively ranked adjectives necessarily occur with higher frequency in positive documents, indicating that other factors are affecting SO beyond the face value of the adjective content. Such adjectives may be more characteristic of negative documents, despite their positive value. Therefore a deeper analysis of the role of adjectives is motivated.

In this paper, we focus on the relevance of adjectives within their surrounding text. Adjectives may be tangential, or even irrelevant, such as the words *soggy* and *stale* in the following (invented) movie review segment: *This movie was fantastic, although the popcorn was soggy and stale.* Here the sentiment is clearly positive toward the movie, though the popcorn is mentioned negatively. Such sentences and collections of sentences (where the reviewer may discuss a tangential topic for a span of more than one sentence before returning to the main topic) indicate that not all adjectives are relevant to the overall sentiment. This degree of relevance can be measured in two forms.

The first measure of relevance relates to the internal structure of a document. Texts have internal structure and are built hierarchically, through building blocks of different importance to the overall communicative intention behind the text. For example, a paragraph may consist of a claim and evidence supporting that claim. The claim is the most important part, and the evidence is secondary. We have used Rhetorical Structure Theory [10] to build discourse structures that capture those basic insights. RST postulates relations between nuclei (most important parts) and satellites (supporting parts). Relations receive names such as Concession, Condition, Evidence and Elaboration. In the example below, we present a fragment from our corpus, with two main units in brackets. In the first one, there is a high number of positive words. The second unit contains one negative phrase *hidden rip-offs*[1]. The two units are joined by a Concession relation, which could be paraphrased as *Although the movie has good aspects, it is too similar to other movies.* If we were able to capture this structure, we could restrict the aggregation of adjective scores to those occurring within the nuclei, and thus remove potentially distracting adjectives.

(1) 1. [It could have been a great movie. It could have been excellent, and to all the people who have forgotten about the older, greater movies before it, will think that as well. It does have beautiful scenery, some of the best since Lord of the Rings. The acting is well done, and I really liked the son of the leader of the Samurai. He was a likeable chap, and I hated to see him die.] 2. [But, other than all that, this movie is nothing more than hidden rip-offs.]

The second means by which we can measure relevance is topicality. We try to determine which sentences are mostly related to the topic being discussed (the movie), and which ones are tangential (the popcorn). If an adjective is found in an off-topic sentence, its score should not be counted in the overall SO. A classifier trained on the concept of topicality is applied to novel documents at the sentence level to determine which sentences are on-topic.

[1] This is a noun phrase, which would not be part of our current analysis, but the adjective *hidden* may convey enough negative SO to help capture the negativity of the second unit.

2.1 SO-CAL

To determine the overall SO score of a document, we use our SO-CAL (Semantic Orientation CALculator) software, inspired by Turney et al.'s work [4], which used a statistical measure of a word's association with positive and negative paradigm or seed words to determine individual word SO values. SO-CAL relies on an adjective dictionary to predict the overall SO of a document, using a simple aggregate-and-average method: The individual scores for each adjective in a document are summed, and then divided by the total number of adjectives in that document. If a word is not found in the dictionary, it is not considered in the analysis[2].

To generate a word's SO, Turney developed a list of seed words that were of either positive or negative polarity (e.g. *excellent* is a positive word). Additional words were then assessed for sentiment according to their co-occurrence with these seed words on the web. Each word was searched for in the presence of the seed words, and using pointwise mutual information we calculated the word's overall sentiment. The basic principle is that if a word is surrounded by negative words, then it, too, is likely to be negative.

Our current system uses the Google search engine (www.google.ca), and the available Google API for our calculations. (See Taboada et al. [6] for other experiments with search engines.) One unfortunate side effect of relying on the web to generate our dictionary was instability. When rerun, the results for each word were subject to change, sometimes by extreme amounts. As a result, an additional dictionary was produced by hand-tagging all adjectives on a scale ranging from -5 for extremely negative, to +5 for extremely positive, where 0 indicates a neutral word. Although clearly not as scaleable, and subject to risk of bias, this gave us a solid dictionary for testing our adjective analyses and a point of comparison for evaluating the utility of the Google-generated dictionaries.

The dictionary currently contains 3,306 adjectives, mostly extracted from the texts that we are processing. The automatic extraction of scores from the web using pointwise mutual information provides values such as those shown in Table 1. The table also provides the hand-tagged values for those adjectives. Note that assigning automatic scores allows us to generate a score for an adjective such as *unlisteneable*, unlikely to be present in a human-generated list. When a new document is processed, any adjectives that are not in the current dictionary are scored using Turney's method, and added to the dictionary.

3 The Experiment

This experiment is broken down into three parts: The baseline extraction of document sentiment from all adjectives in the document, as in Taboada, Anthony, and Voll [6]; the extraction of sentiment on the basis of discourse analysis; and extraction using topic assessment. In all cases, the final calculation of sentiment is done via SO-CAL. Only the choice of which adjectives to include in that calculation varies.

[2] A more detailed description of this algorithm is available in Taboada et al. [6].

Table 1. Automatically-generated and manual scores for some sample adjectives

Word	Automatic	Hand-ranked
air-conditioned	9.11	5
configurable	3.61	2
flawless	2.03	5
ghastly	-6.84	-5
listeneable	-0.87	2
stand-offish	-4.85	-2
rude	-4.62	-3
tedious	-0.88	-1
top-quality	5.33	4
unlisteneable	-7.94	-5

3.1 Corpus

We use the corpus described in Taboada et al. [6], which consists of a collection of Epinions reviews (www.epinions.com) extracted on eight different categories: books, cars, computers, cookware, hotels, movies, music and phones. Within each collection, the reviews were split into positive and negative reviews, providing 25 each, for a total of 50 in each category, and a grand total of 400 reviews in the corpus, which contains 279,761 words. We determined whether a review was positive or negative through the “recommended” or “not recommended” feature provided by the review’s author.

3.2 Discourse Analysis: Using SPADE

In Rhetorical Structure Theory (RST), the coherence of any piece of text is attributable to the underlying discourse structure, which can be captured through the presence of rhetorical relations within the text [11,10]. Rhetorical relations represent functional correspondences between consecutively-placed text spans and assist in conveying the document’s objective, including emphasis. In analyzing the discourse structure of a text, it is possible to not only identify spans, but also to determine which span is more central to the text’s objective. Central spans are marked as nuclei, while less central, or supporting spans, are marked as satellites. As the dominant elements in a text, we hypothesize that the adjectives within the nuclei of a document are also more central to the overall sentiment, while avoiding potential interference by the satellite adjectives, whose sentiments are arguably more tangential to the text’s overall sentiment.

In order to extract sentiment from the nuclei, we need a discourse parser that can segment text into spans, and identify which ones are nuclei and which satellites. To date, few successful discourse parsers exist, leaving much RST annotation to be done by hand. Soricut and Marcu’s SPADE parser [12] parses the relationships within a sentence, but does not address cross-sentential relationships. As such, we use SPADE in our analysis.

Each document is prepared and run through SPADE, creating a series of discourse-annotated files. Subsequently, the outer-most nuclei of each sentence are extracted. Nuclei nested further inside the tree are not considered at this point. We then run the nuclei for each text through our SO-CAL to determine the text's semantic orientation.

3.3 Machine Learning: Topic Classification Using WEKA

In this experiment, we first extract topic sentences from each text, and then run only those through our SO-CAL. To determine topicality for sentences, we use decision trees. Within the WEKA software suite [13] we train a decision tree[3] on the basis of on-topic/off-topic documents, and then use the resulting model to classify the individual sentences of the on-topic documents as on- or off-topic.

Data is split into eight cases, where each Epinions topic is in turn used as the positive, or on-topic instance for the classifier, while all others topics indicate negative, or off-topic instances. The remaining instance attributes rely on the words present within each document. Unfortunately, using all possible words (even if confined to only those present within the training set) creates an impossibly large attribute set[4]. Having too many features during training causes significant amounts of noise, leading to data overfit and consequently useless results. In order to prune the set of attributes, each word found in our corpus is listed in order of its occurrence rate. For this initial iteration of the experiment, the top 500 most common words were extracted and formed the attribute set[5]. Once created, the attribute list is used to generate a feature vector for each document, where a 1 indicates the presence of one or more occurrences of a word in a document, while a 0 indicates its absence (i.e., an ARFF file is generated in WEKA parlance). In addition, an attribute indicating on- or off-topic, for each document, is also included in the vector. These vectors are then run through the classifier, training a total of eight models (one for each topic designation), each of which demonstrates a 96% accuracy when tested using 10-fold cross validation.

Since the ultimate goal is to determine the topicality of the individual sentences, not the entire document, the test set for each classifier model is formed from the sentences in each on-topic document (depending on the relevant topic and model). Each sentence results in a feature vector, generated in the same fashion as for the entire document, with the topic set to unknown. After training, the on-topic sentences are compiled into collections representing the documents now limited to on-topic sentences only. Each collection is then run through SO-CAL to determine its SO value.

[3] We used the Id3 algorithm, after running a comparative analysis of several decision-tree algorithms on our data.

[4] In our case, this amounted to over 15,000 features on which to train.

[5] A stop list of 300 words was used to remove the effect of the most frequently occurring, low-information content words, as well as the addition of any words determined irrelevant. As a useful side effect, this also eliminated any noise in the form of spelling errors, which arise as rare or single-use words.

4 Results and Discussion

The reviews in the corpus are classified into negative and positive, according to the "recommended" feature selected by the author. Our evaluation is based on how often our SO score coincided with the author's recommendation. Below, we present results for all three methods. But first, we would like to mention two changes necessary because of over-positive reviews and lack of adjectives.

We detected a trend towards positive results, suggesting a bias present perhaps in the original reviews: Negative reviews do not contain as many negative adjectives as positive reviews do positive ones. To account for this, all SO values were shifted by adding a normalization factor to each. This value was determined by graphing the change in the results for all topics over various normalization factors, and choosing the factor with the highest overall improvement. The factor was 0.46 for SO-ALL, 0.03 for SO-SPADE, and 0.8 for SO-WEKA.

Another problem encountered was the presence of reviews that are too short, that is, reviews containing too few sentences (and as a result, too few adjectives) for the SO-CAL analysis to work correctly, especially when we restrict the analysis to relevant sentences. In the original data set, there were no cases of a document containing zero adjectives, and thus there was at least an SO value generated in all cases. In both the SPADE and the WEKA analyses, however, since sentences were removed from the original texts, this was not always the case. We therefore introduced a threshold, and files not containing sufficient sentences for analysis were not considered in the overall results. One counter-argument to this approach is that, in some instances, even the one or two sentences remaining in a document after analysis may be the most characteristic.

Table 2 shows a comparison between a baseline analysis using the Google-generated dictionary and using the hand-ranked dictionary. The hand-ranked dictionary shows a significant performance increase. These results were not shifted as normalization had no effect on the Google results. The remaining results are calculated using the hand-ranked dictionary, and show the results both as-is and shifted. These results are summarized in Table 3.

Table 2. Google vs. hand-ranked dictionary, no normalization

Dictionary	Percent correct
Google	56%
Hand-ranked	63%

When normalized, the results generated using our topic and discourse-based analyses are comparable to that of the baseline aggregate-and-average over all adjectives defined by SO-ALL. The use of SPADE limits our system to approximately an 80% accuracy rate in assigning discourse structure. This error is compounded in the subsequent analyses in SO-CAL, and is a likely explanation for the failure of SO-SPADE to improve beyond SO-ALL. SO-WEKA showed a considerable improvement over both SO-SPADE and SO-ALL before normalization.

Table 3. SO-CAL Results using hand-ranked dictionary

Category	Percent Correct
SO-ALL	63%
SO-ALL-SHIFT	72%
SO-SPADE	61%
SO-SPADE-SHIFT	69%
SO-WEKA	69%
SO-WEKA-SHIFT	73%

Improvements in the classification models, and choices of attributes, will offer further improvements in SO-WEKA's ability to discern relevant adjectives.

It is also interesting to note that all three algorithms show an improvement over the results found in our previous work [6]. A much larger dictionary, and improvements in the underlying SO-CAL algorithm have resulted in better performance. In particular, we have modified the part-of-speech tagging, to take into account phenomena found in on-line writing, such as lack of capitalization and extensive use of acronyms and abbreviations.

In terms of performance, the base algorithm, SO-CAL (and thus SO-ALL) performed all analyses in under a minute[6]. In running SO-WEKA there is an initial cost of training the WEKA classifier; however, this does not need to be repeated once completed. Testing the individual sentences against the topic models for each document incurred a time cost of approximately one document per ten seconds. The most expensive of the algorithms was SO-SPADE, which incurred a very high time cost due to its need to first partially parse the document. The approximate cost for this analysis was one document per six minutes.

5 Future Work

The initial results stated above clearly motivate future work in this area. In particular, an emphasis is needed on the production of a high-quality word dictionary for SO analysis. As mentioned earlier, the Google generated dictionary is unstable and requires further study to determine the nature of this instability and its effect on analysis; such instability has the greatest impact on potency, and can be normalized by assigning a binary assessment of sentiment that does not consider magnitude (i.e., assigning either 1 or -1 to each adjective).

Increasing the coverage of the dictionary is also one possibility, though it requires further study into the robustness of doing so. It may be the case that increasing the dictionary beyond a certain point will add noise to the system, and may degrade results. Analyses of occurrence counts, such as were done for the choice of attributes in the decision-tree classifier, may prove useful.

In addition, we plan to expand the dictionary beyond adjectives, and to consider full phrases, rather than individual words.

[6] All tests were run on a 1.5 gHz G4 (OS 10.3.9).

In determining topicality, a comparative look at the effect of a variety of feature sets may be interesting (for example, considering the top 300 most commonly used words, and studying the impact of the chosen stop list). Furthermore, in generating the feature vectors for each sentence with a document, it is possible to analyze which sentences in a document contain the highest degree of features. For example, in this study it was noted that rarely did the first sentence of a document contain any features. Re-running the texts using weights, as we already did in Taboada and Grieve [9] will likely result in further improvements.

Expanding the analysis of rhetorical relations to those existing across sentences may enhance the ability to detect central spans, and reduce the effect of inadequate data once the satellites are removed (as presumably we would extract larger spans with more data). We did not consider the effect of nested relations, but this choice should be more thoroughly examined. In general, the rhetorical parser stands to improve greatly. There is some ongoing research, by our group and elsewhere, on automatically classifying rhetorical relations [14,15]. We expect to be able to take advantage of such improvements.

Finally, we do not make use of comparative and superlative forms of adjectives (*cooler, coolest*). Stemming would be useful, but so would be preserving the information that the adjective was used in a comparative or superlative form.

6 Conclusion

The automated assessment of a text's semantic orientation can be used in a wide variety of tasks, ranging from assessing public opinion to enhancing machine-learning algorithms. In this paper, we consider the role of adjectives in a text as the primary conveyors of sentiment. In particular, we demonstrate that not all adjectives are created equal and their significance varies depending on the text spans or sentences in which they occur. Relevant adjectives include those found in on-topic sentences, and in text spans playing a central role in the overall communicative objective of the text. By restricting our analysis of SO to such sentences, we can remove the effect of text spans tangential to the main sentiment and improve the accuracy of the resulting SO scores. Our initial results show promise by performing comparably to baseline analyses that do not take relevance into account. As there remains room for improvement in the determination of adjective relevance, the conclusion is that the performance will improve beyond such baseline analyses in correctly predicting the orientation of documents.

Acknowledgments

This work was supported by the Natural Sciences and Engineering Research Council of Canada. Thanks to Caroline Anthony for her work on the SO-CAL software, and to Chris Mathieson for his insightful comments. We also thank Google for making their API available.

References

1. Osgood, C.E., George, S., Tannenbaum, P.: The Measurement of Meaning. University of Illinois, Urbana (1957)
2. Hatzivassiloglou, V., McKeown, K.: Predicting the semantic orientation of adjectives. In: Proceedings of 35th Meeting of the Association for Computational Linguistics, Madrid, Spain, pp. 174–181 (1997)
3. Turney, P.: Thumbs up or thumbs down? Semantic orientation applied to unsupervised classification of reviews. In: Proceedings of 40th Meeting of the Association for Computational Linguistics, pp. 417–424 (2002)
4. Turney, P., Littman, M.: Unsupervised learning of semantic orientation from a hundred-billion-word corpus. Technical Report ERB-1094, NRC 44929, National Research Council of Canada (2002)
5. Turney, P., Littman, M.: Measuring praise and criticism: Inference of semantic orientation from association. ACM Transactions on Information Systems 21(4), 315–346 (2003)
6. Taboada, M., Anthony, C., Voll, K.: Creating semantic orientation dictionaries. In: LREC. Proceedings of 5th International Conference on Language Resources and Evaluation, Genoa, Italy, pp. 427–432 (2006)
7. Stone, P.J., Dunphy, D.C., Smith, M.S., Ogilvie, D.M.: The General Inquirer: A Computer Approach to Content Analysis. MIT Press, Cambridge (1966)
8. Fellbaum, C. (ed.): WordNet: An Electronic Lexical Database. MIT Press, Cambridge (1998)
9. Taboada, M., Grieve, J.: Analyzing appraisal automatically. In: Proceedings of AAAI Spring Symposium on Exploring Attitude and Affect in Text (AAAI Technical Report SS-04-07), pp. 158–161, Stanford University, CA, AAAI Press (2004)
10. Mann, W.C., Thompson, S.A.: Rhetorical Structure Theory: Toward a functional theory of text organization. Text 8(3), 243–281 (1988)
11. Taboada, M., Mann, W.C.: Applications of Rhetorical Structure Theory. Discourse Studies 8(4), 567–588 (2006)
12. Soricut, R., Marcu, D.: Sentence level discourse parsing using syntactic and lexical information. In: HLT-NAACL 2003. Proceedings of Human Language Technology and North American Association for Computational Linguistics Conference, Edmonton, Canada (2003)
13. Witten, I.H., Frank, E.: Data Mining: Practical Machine Learning Tools and Techniques, 2nd edn. Morgan Kaufmann, San Francisco (2005)
14. Egg, M., Redeker, G.: Underspecified discourse representation. In: Benz, A., Kühnlein, P. (eds.) Constraints in Discourse, John Benjamins, Amsterdam and Philadelphia (to appear)
15. Sporleder, C., Lascarides, A.: Using automatically labelled examples to classify rhetorical relations. Natural Language Engineering (to appear)

A Bio-inspired Method for Incipient Slip Detection

Rosana Matuk Herrera[*]

Department of Computer Science, Facultad de Ciencias Exactas y Naturales, Universidad de Buenos Aires, Argentina
rmatuk@dc.uba.ar

Abstract. Few years old children lift and manipulate unfamiliar objects more dexterously than today's robots. Therefore, it has arisen an interest at the artificial intelligence community to look for inspiration on neurophysiological studies to design better models for the robots. In a human dexterous manipulation a crucial event is the detection of incipient slips. Humans detect the incipient slips based on the responses of their tactile mechanoreceptors. In this paper, we propose a method to detect the incipient slips using artificial neural networks that receive as input simulated human afferent responses. This method is strongly inspired on neurophysiological studies of the afferent responses during the human dexterous manipulation. Finite element analysis was used to model two fingers and an object, and simulated experiments using the proposed method were done. To the best of our knowledge, this is the first time that simulated human afferent signals are combined with finite element analysis and artificial neural networks, to detect the incipient slips.

Keywords: Neural networks, Dexterous manipulation, Robotics.

1 Introduction

In the field of robotics, one of the most important problems is the development of robot hands that can grip and lift objects avoiding slip and crushing, even when the weight and friction coefficient are unknown. Grasp stability, i.e., the prevention of accidental slips as well as of excessive fingertip forces, requires tuned grip force. Dexterous manipulation of objects requires to apply the following fingertip forces to the object of interest:

1. *Load force (LF):* Vertical force required to overcome the force of gravity.
2. *Grip force (GF):* Normal force required to prevent slips.

To avoid a slip between the fingers and the object, the grip force must be greater than the slip limit, i.e., the load force divided by the friction coefficient (μ):

[*] Postal address: Departamento de Computación, Pabellón 1, Ciudad Universitaria, 1428, Buenos Aires, Argentina.

$GF \geq LF/\mu$. Thus, the friction coefficient determines the ratio between the grip and the lift force required to prevent slips. Unfortunately, the friction is a difficult parameter to estimate since it depends on several factors, such as the amount of sweat from the fingertips and the relative motion between the fingertips and the contact area, and therefore changes over time. Hence, the ability to adequately tune the grip force relies on the tactile information from local micro-slips at the periphery of the finger-object contact area [1]. These micro-slips are a precursor to the unwanted gross slip, and makes it possible to sense the slip limit before it is reached. This sensory mechanism is an important key to the efficient grasp stability in humans [2].

2 Incipient Slip Detection by Humans

There are four types of afferents in the glabrous skin of the human hand: FA-I, SA-I, SA-II and FA-II. All together, there are approximately 17,000 mechanoreceptors in the glabrous skin of each hand. The SA I and SA II units are referred to as *slowly adapting afferents*, which means that they show a sustained discharge in the presence of ongoing stimuli. In contrast, the *fast adapting afferents*, FA I and FA II, fire rapidly when a stimulus is presented (or removed) and then fall silent when the stimulation cease to change [3]. The skin area where responses can be evoked from a receptor is called its receptive field (RF). The RFs of FA I and SA I units typically have sharply delineated borders and show several small zones of maximal sensitivity. In contrast, the SA II and FA II have large RFs but only one zone of maximal sensitivity and a gentle increase of threshold further and further away from this zone [3,4,5,6].

Responses to slips are evoked in FA-I, FA-II and SA-I afferents. In contrast, responses originated from small slips localized to only a part of the skin area in contact with the object and in the absence of detectable vibrations events, are only observed in the FA-I and SA-I units and are often followed by small force ratio changes. Such slips, called *localized frictional slips*, also trigger appropriate grip force adjustments for grasp stability when an object is held in the air. Strong burst responses in the FA II units caused by transients other than those evoked by slips did not trigger upgradings of the force ratio. Thus, by exclusion, signals in the type I units would be indispensable to the adjustment of the grip to the friction [7,8].

3 Method

Contact between an elastic finger with a curved surface and an object with a plane surface was analyzed using the finite element method (FEM). Finite element code *MSC.Marc release 2* was used to create a 3-D model of two fingers and an object. To simulate the FA-I afferent responses, we adapted software code provided by Anna Theorin (f. Israelsson), Lars Rådman, Benoni Edin and Roland Johansson of Umeå University. Israelsson [4] designed a computational simulator of the responses of the afferents from the glabrous skin during human manipulation. Thus, the afferent responses were obtained adapting software code of the Israelsson's human afferent simulator program. *Matlab 7* modules were programmed to process the output of the *MSC.Marc*, and artificial neural networks were programmed to process the output of the Israelsson's afferents simulator. Thus, our software contribution consists of designing the FEM model, programming the simulated experiments using the FEM model, formatting the output of the FEM model to be feeded to the Israelsson's afferent simulator program, and processing the output of the Israelsson's afferent simulator program with artificial neural networks (Fig. 1).

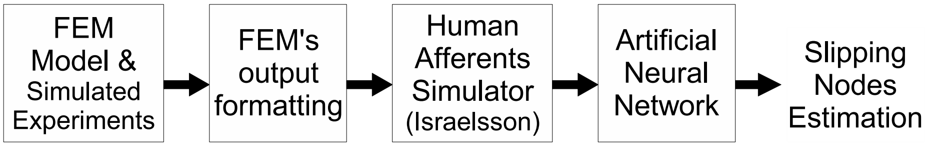

FEM Model & Simulated Experiments
FEM's output formatting
Human Afferents Simulator (Israelsson)
Artificial Neural Network
Slipping Nodes Estimation

Fig. 1. Block diagram of the software modules

Three deformable contact bodies were modeled: left finger, right finger and object (Fig. 2). The table was modeled as a rigid body at the bottom. The skin was modeled as a linear isotropic elastic material, for which two constants, elastic modulus and Poisson's ratio, need to be specified. Since soft tissues are generally considered incompressible owing to their predominant fluid content, the Poisson's ratio was assumed to be 0.48, close to the theoretical limit of 0.5 [9,10]. The elastic modulus was assumed to be 0.18 MPa [11]. The object was modeled as a unique block element. The Poisson's ratio of the object was set at 0.48 and the Young modulus at 0.72 MPa. Therefore the object was almost incompressible and harder than the finger.

Nodes at the surface of the elastic finger should satisfy the following equations (adapted from [12]):

1. non contact nodes: $w_n \neq w_{nobject}, f_n = 0$
2. contact nodes: $w_n = w_{nobject}, f_n > 0$
3. sticking nodes: $w_t = w'_t + (w_{tobject} - w'_{tobject}), |f_t| < \mu|f_n|$
4. slipping nodes: $|f_t| = \mu|f_n|$

where, w_n and w_t are the nodal locations in the normal and tangential directions, f_n and f_t are the nodal forces in the normal and tangential directions, $w_{nobject}$ is the normal location of the object, $w_{tobject}$ is the tangential location of the object, and μ is the friction coefficient. Although the static friction coefficient is usually larger than the kinetic friction coefficient, these values are assumed the same in the fundamental calculations to simplify the argument. The prime represents values in the previous time frame.

When an incipient slip occurs at the preload phase, there are signals of the SA I and FA I mechanoreceptors (see Sect. §2). The experiment to detect the

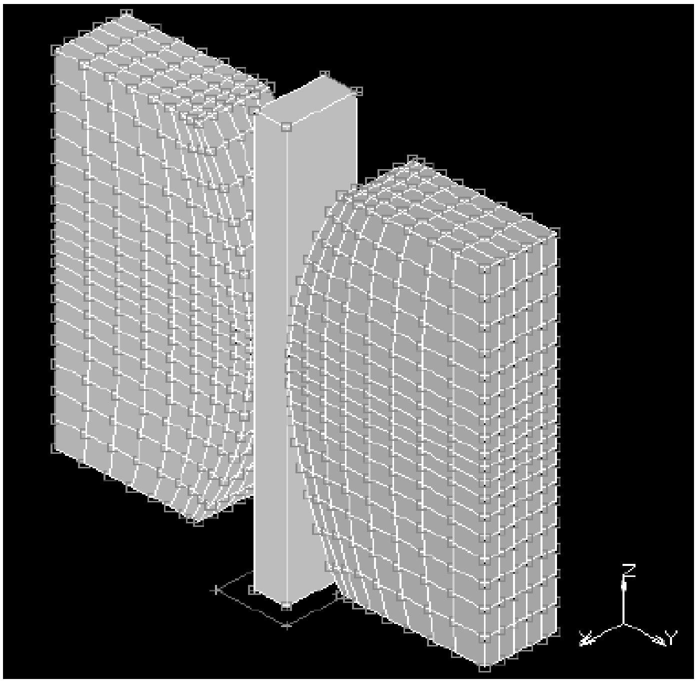


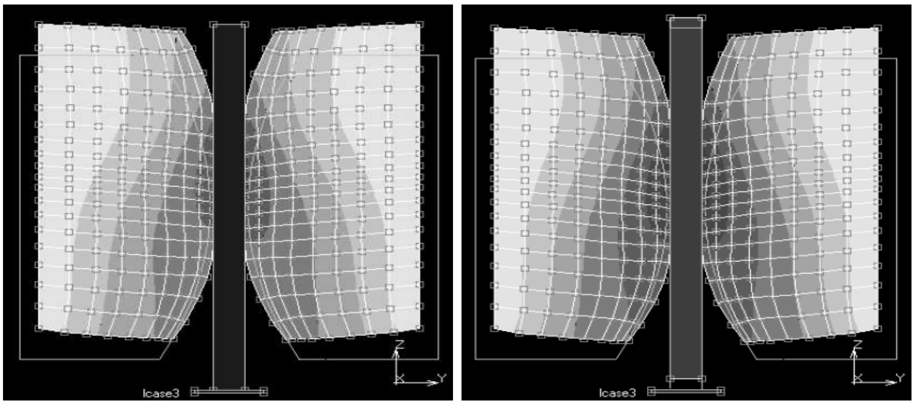
Fig. 2. Fingers and object at the initial state

incipient slip consisted of applying a normal force to grip the object, and then applying a load force to lift the object. The grip and load force were raised incrementally. The preceding equations were used to know if the nodes were sticking or slipping, and to know if the nodes were in contact with the object or not. The goal was to create an association between the state of the nodes at each force increment (i.e., number of sticking/slipping nodes, number of contact/non contact nodes) and the FA I and SA I responses.

4 Implementation

The object was restricted to move on the z-axis. The table was fixed in all axis. The initial position of the fingers and the object is shown in Fig. 2. First, the displacement of the fingers was fixed on the x-axis and a grip force was applied. Afterwards, a vertical load force was applied. Depending on the friction coefficient of the object's material, when the same amount of grip and load force was applied on an object of the same weight, the object may lift off or not (Fig. 3).

Due to the symmetry of the afferent responses, only the afferent responses of the 19 nodes situated along the central line at the front of the finger were recorded. The input patterns were built joining the FA I and SA I signals (i.e.,



(a) (b)

Fig. 3. Fingers with object at the final state after applying the grip and load forces. The contour lines correspond to the initial state. (a) The friction coefficient is $\mu = 0.5$. The fingers have slipped and the object has not been lifted. (b) The friction coefficient is $\mu = 0.7$. The object has lifted off.

a pattern of dimension 38 was obtained for each force increment). To build the target the number of contact nodes cn and slip nodes sn were counted, and the value $\frac{sn}{cn}$ was used as target (i.e., the proportion of contact nodes that were slipping). The patterns were preprocessed to normalize their mean and standard deviation in order to improve the performance of the training of the neural network. A multi-layer perceptron was designed to associate the SA I and FA I patterns at each force increment, with the percentage of contact nodes that were slipping. The activation function in the hidden layer was chosen to be a sigmoidal function in the form of a hyperbolic tangent for its convenient antisymmetric property. The output layer activation function was chosen to be logsig, because the proportion of contact nodes that were slipping could only be a value into the interval [0..1]. For its fast convergence, the Levenberg-Marquardt algorithm was chosen as the minimization method to train the network.

5 Results

Experiments with an object of weight 200 were made. Several simulated experiments were done changing the friction coefficient of the object from 0.50 to 0.90, and a total of 2100 input patterns were generated. A random partition of the input patterns set was done, in which the 90 percent of the patterns was chosen as training set and the remaining 10 percent as testing set. Multilayer perceptrons with different numbers of units in the hidden layer were trained and tested, and a multilayer perceptron with 1 hidden layer of 3 neurons obtained the best performance. Thus, the final architecture of the neural network consisted of an input layer of 38 units, a hidden layer of 3 units, and an output layer of 1 unit. The neural network was trained, and at the final state (46th. epoch), the mean squared error of the training set was 5.5617e-004 and its mean absolute error was 0.0135. The mean squared error of the testing set was 0.0016 and its mean absolute error was 0.0178. The performance for the training and testing sets is shown in Fig. 4. The regression analysis between the network response and the corresponding targets is shown in Fig. 5.

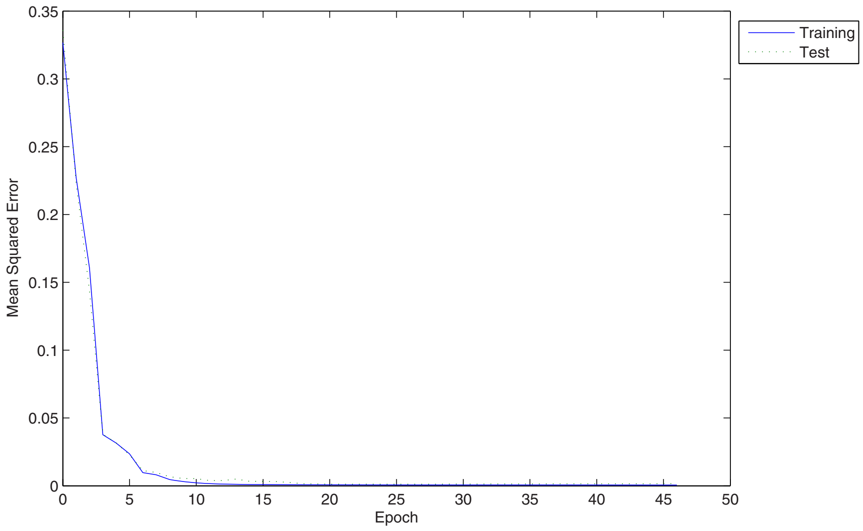


Fig. 4. Performance of the neural network for the incipient slipping estimation

In order to analyze the relation between the proportion of slipping contact nodes and the grasp stability, the same grip:load force ratio was applied on an object of weight 200 changing the friction coefficient. When $\mu = 0.5$ the object did not lift off, and when $\mu = 0.7$ or $\mu = 0.9$ the object lifted off (Figs. 6 and 7).

6 Discussion

The neural network was trained with 1890 patterns. The final mean squared error of the test set of 210 patterns was 0.0016 (Fig. 4). The mean absolute error of the test set was 0.0178. As the maximum value of the target test set was 1.0, this result gives a relative error for the testing set of only 1.78%. The regression analysis between the network response and the corresponding targets is shown in Fig. 5. In Fig. 5, it is seen that $m = 0.995$ and $b = 0.00234$ are very close to 1 and 0 respectively, and then the regression is very near to a perfect fit. The R-value is 0.998 that is very close to 1 and indicates a good fit.

To analyze the relation between the proportion of slipping contact nodes and the grasp stability, in Fig. 6 and 7 we compare the effect of applying the same

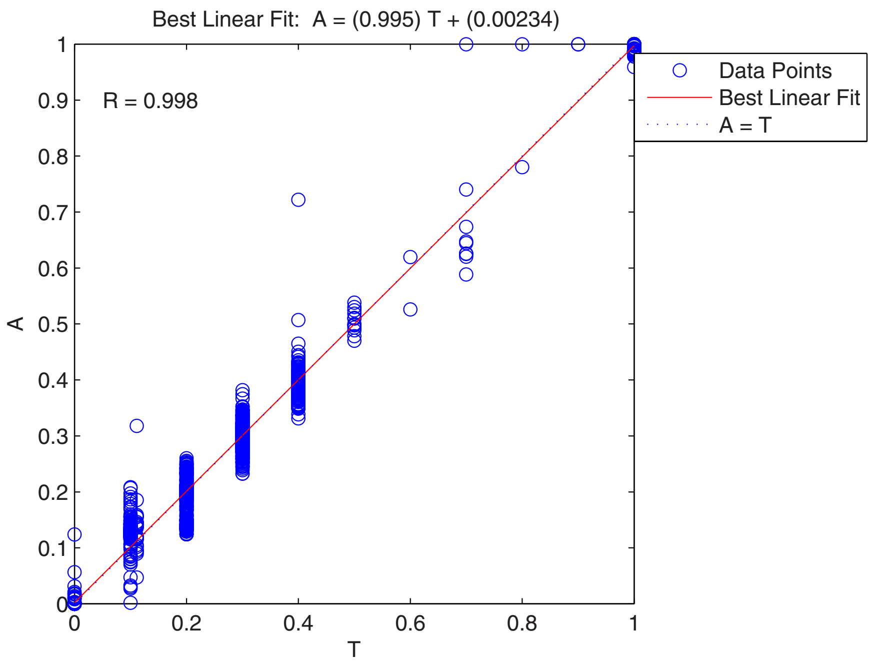


Fig. 5. Regression analysis between the network response (A) and the corresponding targets (T). The best linear fit is indicated by a dashed line. The perfect fit (output equal to targets) is indicated by the solid line. In this figure is difficult to distinguish the best linear fit line from the perfect fit line because the fit is so good.

grip:load force ratio on an object of weight 200 for different friction coefficients. When the material had a friction coefficient of value $\mu = 0.5$, the object slipped and it did not lift off. When the friction coefficient was enlarged (i.e., the object's material was less slippery) the object lifted off (Fig. 6). Analyzing the percentage of contact nodes that were slipping it is observed that when the friction coefficient was enlarged the percentage of slipping nodes decreased (Fig. 7). Comparing the grip:load force ratio with the slip ratio (slip ratio $= \frac{1}{\mu}$), it was seen that when the object did not lift off ($\mu = 0.5$), the force ratio was lesser than the slip ratio, and when the object lifted off ($\mu = 0.7$ and $\mu = 0.9$) the force ratio was greater than the slip ratio. These results are in accordance with the mechanical law that establishes that must be $|LF| < \mu|GF|$ for the object to lift off (see Sect. §1). Comparing Fig. 6 and Fig. 7 it is seen that when the percentage of slipping contact nodes is less or equal to 40% the object lifts off. On the other side, in Fig. 5 and 8 it is observed that the percentage of slipping nodes goes abruptly from 40% to 100%. These results show that 40% is the limit before the gross slip. Then, the estimation of the percentage of slipping contact nodes does not need to have a perfect precision to use it as the support of the grasp stability (i.e., if a 30% of slipping can be detected with a 10% of precision, a gross slip can

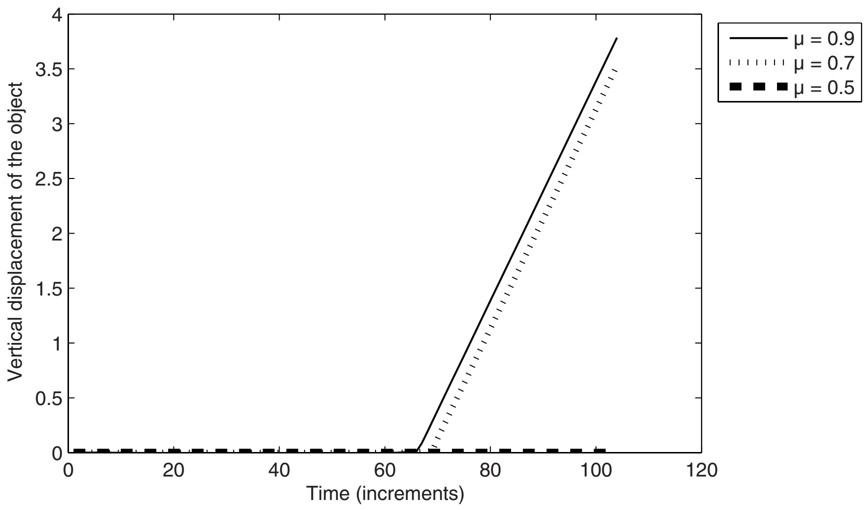


Fig. 6. Vertical displacement of an object of weight 200 for different μ when the same grip:load force ratio was applied

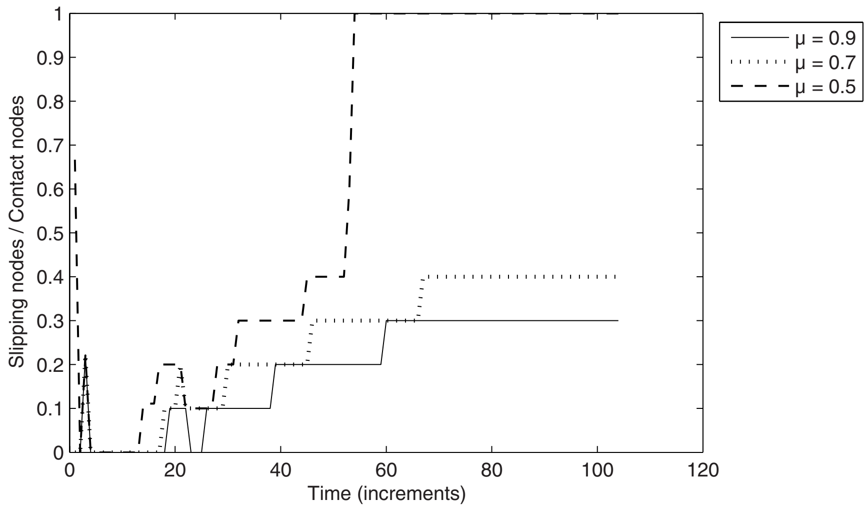


Fig. 7. Slipping nodes:contact nodes for different μ when the same grip:load force ratio was applied on an object of weight 200

be avoided). Therefore, as the proposed method estimates the incipient slipping with a relative error of only 1.78%, this result shows an excellent performance for the incipient slipping detection.

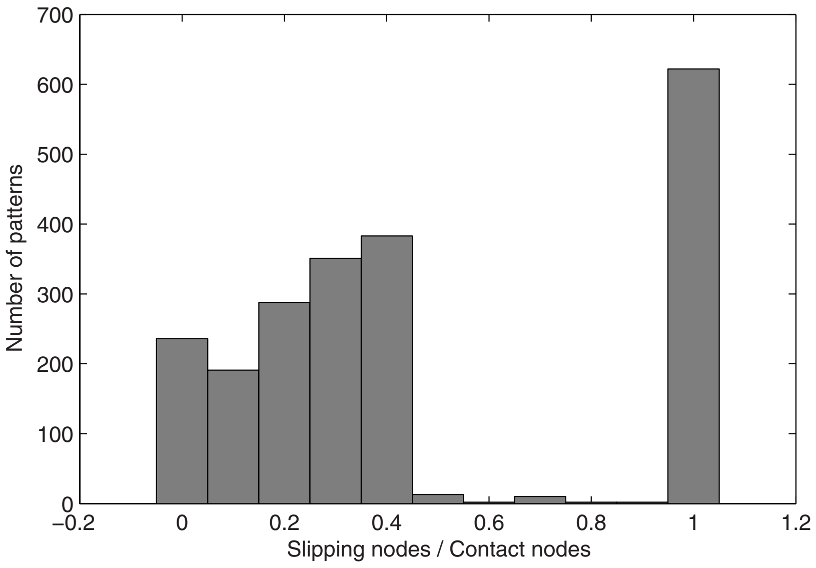


Fig. 8. Histogram of the percentage of slipping contact nodes for all the 2100 input patterns

7 Conclusions and Future Work

In this paper, we have proposed a bio-inspired method for the detection of the incipient slips. This is a crucial information for a dexterous manipulation process in unknown environments. The proposed method is strongly inspired on neurophysiological studies of the human dexterous manipulation. A finite element method model was built, and simulated experiments with planar objects of different friction coefficients were done. A multilayer perceptron was used to estimate the percentage of contact nodes that were slipping using as input simulated human afferent signals. The incipient slip was detected with high precision. To the best of our knowledge, this is the first time that simulated human afferent signals are combined with finite element analysis and artificial neural networks, to detect the incipient slip event.

The conjunction between robotics and human neurophysiology can give fruitful and interesting results. The obtained results show that a bio-inspired approach could be a promising way to explore in order to afford the dexterous manipulation of unknown objects. A future work consists in implementing the method in a real robotic hand with 3-D force sensors.

Acknowledgments. The author acknowledges Anna Theorin (f. Israelsson), Lars Rådman, Benoni Edin and Roland Johansson for kindly providing the software code of their afferent simulator program, and Enrique Carlos Segura for helpful discussions about this paper.

References

1. Smith, A.: Some shear facts and pure friction related to roughness discrimination and the cutaneous control of grasping. Can. J. Physiol. Pharmacol. 72, 583–590 (1993)
2. Fagergren, A.: A multidisciplinary system identification of the human precision grip. PhD thesis, Karolinska Institutet, Stockholm (2003)
3. Johansson, R.: Tactile sensibility in the human hand: receptive field characteristics of mechanoreceptive units in the glabrous skin area. Journal of Physiology 281, 101–123 (1978)
4. Israelsson, A.: Simulation of responses in afferents from the glabrous skin during human manipulation. Master's thesis, Master thesis in Cognitive Science, Umeå University, Umeå, Sweden (2002)
5. Johansson, R.: Sensory and memory information in the control of dexterous manipulation. In: Neural Bases of Motor Behaviour, pp. 205–260. Kluwer Academic Publishers, Dordrecht (1996)
6. Johnson, K.: The roles and functions of cutaneous mechanoreceptors. Curr. Opin. Neurobiol. 11, 455–461 (2001)
7. Johansson, R., Westling, G.: Signals in tactile afferents from the fingers eliciting adaptive motor responses during precision grip. Exp. Brain Res. 66, 141–154 (1987)
8. Westling, G., Johansson, R.: Responses in glabrous skin mechanoreceptors during precision grip in humans. Exp. Brain Res. 66, 128–140 (1987)
9. North, J., Gibson, F.: Volume compressibility of human abdominal skin. J. Biomech. 203–207 (1978)
10. Srinivasan, M., Gulati, R., Dandekar, K.: In vivo compressibility of the human fingertip. Adv. Bioeng. (22), 573–576 (1992)
11. Dandekar, K., Raju, B., Srinivasan, M.: 3-d finite-element models of human and monkey fingertips to investigate the mechanics of tactile sense. Transactions of the ASME 125, 682–691 (2003)
12. Maeno, T., Kawamura, T., Cheng, S.: Friction estimation by pressing an elastic finger-shaped sensor against a surface. IEEE Transactions on Robotics and Automation 20(2), 222–228 (2004)

TalkBack: Feedback from a Miniature Robot

Yasser F.O. Mohammad and Toyoaki Nishida

Nishida-Sumi Laboratory, Department of Intelligence Science and Technology, Graduate School of Informatics, Kyoto University, Japan
yasser@ii.ist.i.kyoto-u.ac.jp, nishida@i.kyoto-u.ac.jp

Abstract. A prerequisite of any successful social robot is the ability to express its internal state and intention to humans in a natural way. Many researchers studied verbal and nonverbal feedback from humanoid robots or humanoid robotic heads but there is little research about the possible feedback mechanisms of non-humanoid and especially miniature robots. The TalkBack experiment is a trial to fill this gap by investigating the effectiveness of using motion cues as a feedback mechanism and comparing it to verbal feedback. The results of the experiment showed that there is no significant difference in the task completion accuracy and time or in the feeling of naturalness between these two modalities and there is a statistically significant improvement when using any of them compared with the no-feedback case. Moreover the subjects selected the motion cues feedback mechanism more frequently as the preferred feedback modality for them.

1 Introduction

For Human-Robot Interaction to proceed in a natural way, the robot must be able to understand human's intention and communicate its own internal state or intention to the human through a combined operation named mutual intention formation and maintenance [1] [2].

In this work we are interested in investigating the effectiveness of using only motion cues in communicating the robot's intention to human users that did not have any previous knowledge of the motion patterns that will be used.

Many researchers have investigated the feedback modalities available to humanoid robots or humanoid heads, but the research about feedback from non-humanoid robots is still limited [3] [4]. In [5], acting in the environment was suggested as a feedback mechanism. For example the robot re-executes a failed operation in the presence of a human to inform him about the reasons it failed to complete this operation in the first place. Although this is an interesting way to transfer information it is limited in use to only communicating failure. Our method of feedback by motion cues can be considered in part as a generalization of the feedback by acting paradigm presented in [5], as it is not limited to failure communication.

2 Design and Setup

The experiment is designed by the authors as a game in which a human is instructing a miniature robot using free hand gestures to help it follow a specific path to a goal in an environment that is projected on the ground. The path is only visible to the human and the robot cannot know it except through the communication with the human operator. Along the path there are some objects of different kinds as will be explained later in this section. Those objects are not seen by the human but are seen by the robot, and the only way the human can give the robot the correct signals is if the robot could transfer its knowledge to him. The experiment was designed so that the partially observable environment can be converted into a completely observable environment if the human and the robot succeeded in exchanging their knowledge. Fig. 1 illustrates this point.

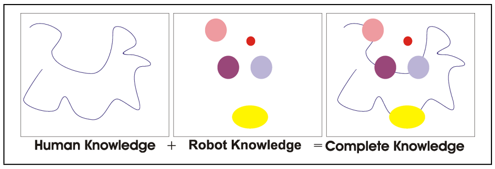


Fig. 1. The TalkBack design principle: Combining Human's and Robot's Knowledge leads to a visible environment

A map describes both the required path that is projected on the virtual environment and the location of different kinds of objects within this virtual environment.

There are five types of objects that can exist in the TalkBack experiment:

– Obstacles that must be avoided
– Sand Islands that slow down the robot
– Key-Gates that open only after stopping over a key
– Keys to open the key-gates
– Time Gates that open periodically

The Obstacles test the ability to send a MUST signal, the sand islands test the ability to send a SHOULD signal, the time gates test the ability to send a time-critical signal, while keys test the ability to send a signal about a location rather than a direction.

Three different maps were used in this experiment. An example of them is shown in Fig. 1. Every time the game is started one object of each kind is placed randomly in the virtual environment subject to the constraint that all the objects except the key lie on the path.

To be able to accomplish its mission (avoiding obstacles, and passing gates while following the human directions as close as possible as fast as possible without being trapped in sand islands), the robot should be able to give the following feedback signals to the human:

1. What about this direction? This signal is a suggestion to go in a specific direction. This can happen in many circumstances like when the robot is facing a key-gate, and needs to go in the direction of the key.
2. I cannot pass in this direction. This signal indicates that there is an obstacle or a closed gate in the direction specified that cannot be passed through now.
3. It will be too slow if I pass here. This signal should be given once the robot is about to enter a sand island.
4. I found the key. This signal should be given once the key is found to help the main operator give the correct gesture to move the robot again toward the path.
5. What should I do now? This signal is needed whenever the robot fails to recognize the gesture given by the human or does not have a specific order for a long time.

Assuming that the robot has no autonomous navigation capabilities, the accuracy and time needed to follow the path will depend on two factors:

1. Recognizing human input
2. Effectiveness and efficiency of robot feedback.

As the main objective of this experiment was to study the effectiveness of different feedback modalities in transferring the internal state or intention of the robot to the human, special care had to be taken to reduce the effects of the first factor above. To achieve that a WOZ-like (Wizard Of Oz like) arrangement was used in which a human watches the main operator and transfers his own understanding of the gesture to the robot. It should be clear that the hidden operator here is not *controlling* the robot, but is used by the robot as a replacement of one of its sensors, namely the gesture recognition sensor. From now on the subject who actually produces the gestures and responds to the robot's feedback will be called the *main operator* while the hidden human who transfers the gestures of the main operator to the robot will be called the *the WOZ operator*.

Using a virtual projected environment was a must in this experiment as it is the only way to make the main operator see the path while not seeing the objects on it.

The environment consists of a 60×40cm rectangular working space at which the required path is projected. A camera is used for localizing the robot using a marker attached to it. There are two environmental cameras to capture the working environment and the main operator in real time and a magic mirror behind which the WOZ operator uses the WOZ software designed by the authors to transfer the main operator's gestures to the robot without being visible. Due to lack of space the details of this software will not be given here.

The miniature robot used in this experiment is called e-puck (http://www.e-puck.org) designed by EPFL with a - two hollow circles - marker attached on top of it. The diameter of the robot is 7cm, and is driven by 2 stepper motors in a differential drive arrangement.

3 Robot Software

The control software of the robot runs on a PC and is built using the EICA (Embodied Interactive Control Architecture) architecture [1]. The details of this implementation are given in the following subsections.

3.1 EICA Architecture

The EICA architecture is a hybrid reactive-deliberative architecture for real world agents that was developed by the authors to combine autonomy with interaction abilities at the lowest level of the agent's control system. Due to lack of space the details of the EICA architecture will not be given here. For details refer to [1], [6], [2].

The EICA system is constituted of system components that are managed by the architecture and application specific components. The system components are the *Intention Function* that represents the user's current intention [6], and the action integrator that fuses the actions suggested by the intention function in light of the *Innate Drives* and *Low Level Emotions* to produce the final commands to the robot's actuators.

There are five types of application-specific components:

– *Virtual Sensors* that represent different views of the raw sensor data that can be easily utilized by other components of the system to reduce duplicated computations. In this experiment virtual sensors were used for localization, detection of different objects, and reading commands from the WOZ operator.
– *Processes* that continuously register intended actions into the *Intention Function* as *Reactive Plans* or *Simple Actions* (usually through *Virtual Actuators*) based on either reactive coupling between sensor data and required actuation, or deliberative planning. In this experiment processes were used in navigation and controlling feedback.
– *Virtual Actuators* that provide a cleaner way to specify the required actuator values in a different vocabulary that is easier to be utilized by other active components. In this experiment virtual actuators were used to solve the reverse kinematics problem.
– *Reactive Plans* that represent intended behaviors of the robot and are implemented as reactive components that create a tight coupling between sensor data and actuator commands in order to achieve a specific short term goal specified by the plan. In this experiment reactive plans were used to give feedback to the main operator.
– *Low Level Emotions* robot-specific information summarizers. In the current implementation, emotions are expressed as Gaussian components with negligible correlation between different components. In this experiment a low level emotion was assigned to each mode of the five modes that are used to adjust the intentionality of the feedback reactive plans. Details of those modes are given in the next subsection

This architecture was chosen in this experiment for the following reasons:

1. Its modular nature allows very fast implementation.
2. Intention communication can be implemented in the reactive part.
3. The Action Integration/Intention Function mechanism of the architecture facilitates implementing signal conflict resolution, and abortion of signals when their conditions are no longer intact.

3.2 Feedback Signals

Five modes where designed that can generate all the needed feedback signals specified in section 2. Those modes are implemented as Low Level emotions in the EICA architecture (see section 3.1). The input from the hidden operator and the internal state adjusts the various values of those modes and when they achieve specific patterns a feedback signal is given.

The modes used in this experiment are:

– *Confusion* which increases when the robot cannot understand the input from the human or when it encounters an unexpected situation.
– *Suggestion* which increases when the robot wants to suggest a specific direction of movement.
– *Resistance* which increases when the command given by the operator will cause a collision from the robot's point of view.
– *Hesitation* which increases when the robot thinks that the human's command is not optimal but has no other suggestion.
– *Satisfaction* which increases when the robot approaches the key.

Table 1 gives the nominal values for the emotion vector that triggers various feedback signals.

Table 1. Nominal Mode Values for Various Feedback Signals

Feedback Signal	Conf.	Sugg.	Resis.	Hesit.	Sat.
What should I do now?	1.0	0	0	0.5	0
I cannot pass in this direction.	0.1	0.8	0	0.8	0
What about this direction?	0	1.0	0	0	0
It will be too slow.	0	0	1.0	1.0	0
I found the key	0	0.9	0	0	1.0

Each one of those modes triggers a specific reactive plan based on its degree. The reactive plans are combined by the action integrator to form the final behavior of the robot. The priority of each reactive plan will be proportional to the mean of the respective Low Level Emotion/Mode and its confidence will depend on the variance of this emotion. This allows the integration manager to choose the most appropriate action at all times without scarifying the consistency in the behavior of the robot.

4 Procedure

The experiment was done by 12 main subjects. The same person served as the WOZ operator in all the sessions to fix gesture recognition accuracy. Each subject interacted with three feedback modalities:

– *Stop.* In this case, the robot just stops whenever it needs to give a feedback. In this mode two simplifications were added to limit the possible time of execution of the experiment. The robot will move in the direction of the key once ordered to rotate to any angle near it, and will stop automatically over the key.
– *Speech.* In this modality, the robot gives a predefined statement for every kind of feedback signal. Table 2 gives the details of those statements. For speech synthesis, SAPI 5.1 was used with the Microsoft default TTS engine.
– *Motion.* In this modality, the robot gives a motion feedback according to Table 1. Table 3 gives the actual outputs of various reactive plans used to implement the motion feedback modality. Table 3 gives the actual outputs of various reactive plans. Two different motion feedback mechanisms were designed for every mode. The robot starts with the first alternative and then switches to the second alternative if the main operator did not correctly understand the signal after a number of times.

Table 2. The statements used by the robot in the Speech feedback mode

Modality	Statement
Confusion	I am confused, What should I do
Suggestion	I suggest going θ degrees clockwise/counterclockwise
Resistance	I do not think I should go in that direction
Hesitation	I am not sure this is wise
Satisfaction	I found the key

The order of the modalities was shuffled so that the six possible orders are used twice each. Each map was used with one third of the subjects. Each subject interacted with the same map in the three episodes of the game to reduce the effects of any inherent difference in difficulty between the various maps.

After finishing interacting with each modality, the main operator guesses how many objects of each kind were existing, and ranks the robot in terms of its ability to recognize gesture, the understandability of its feedback, and how efficient it was in giving this feedback in a 0-10 scale. The main operator is also asked to rank how enjoyable was the session in the same scale. The WOZ operator also ranked his own gesture recognition accuracy after each episode (0-10). After finishing the whole set, the main operator is asked to select his/her preferred feedback modalities and the fastest episode, and to write an optional free comment.

Table 3. The Motion Based Feedback mechanisms associated with each Low Level Emotion/Mode

Modality	First Alternative	Second Alternative
Confusion	rotate around once completed a revolution return back	rotate 30 degrees, go forward and backward, then continue rotating
Suggestion	rotate to the suggested direction go forward and backward	rotate to suggested direction slowly, then rotate back faster
Resistance	rotate to the reverse direction go forward and backward	rotate to reverse direction slowly, then rotate back faster
Hesitation	rotate few degrees to each direction then back	suggest a different direction
Satisfaction	rotate continuously clockwise	stop

5 Results and Discussion

Eight dimensions of evaluation were used to determine the effectiveness of the three feedback modalities used in the experiment. The first four of them are objective measures based on the task, while the last four are subjective evaluations of the main operator.

– Time to completion
– Path error. This error is a measure of how different the actual path of the robot was from the desired path. To calculate this value, the location of the robot is logged every 10 milliseconds except in the motion feedback periods in the *Motion* mode. For each logged point, the distance to the required path is calculated, then the point error is averaged over all the points.
– Number of Collisions
– Correct object identifications.
– Understandability, or how well the main operator understood the robot's feedback
– Recognition, or how well the main operator thinks the robot can understand his/her gesture.
– Efficiency, or how much time is needed to understand robot's messages.
– Pleasure, or how enjoyable was the experiment.

Table 4. Comparative Statistics of different evaluation dimensions for different feedback modalities

modality		time	error	collisions	correct_ objects	understa ndability	recognition	efficiency	pleasure
Stop	Mean	13.4111	19.4348	8.00	1.33	3.25	6.50	2.67	1.75
	Std. Deviation	1.77806	4.47034	3.568	.778	1.815	1.784	1.826	1.603
Speech	Mean	7.0336	14.2188	.83	1.17	5.83	6.42	5.17	6.17
	Std. Deviation	1.31192	3.59448	.577	1.193	1.697	1.621	1.850	1.586
Motion	Mean	6.7701	12.6225	1.00	1.83	6.33	6.83	5.00	6.83
	Std. Deviation	1.52190	4.81379	.953	1.267	1.670	1.749	2.045	1.850
Total	Mean	9.0716	15.4254	3.28	1.44	5.14	6.58	4.28	4.92
	Std. Deviation	3.45818	5.13113	3.983	1.107	2.167	1.680	2.186	2.812

The mean, and standard deviation of each of the eight dimension of evaluation were computed and compared. Table 4 gives the values for each modality and the average value across all modalities.

Fig. 2 gives a graphical view of the difference in the mean between the three modalities in the eight dimensions.

Independent samples t-test was applied to find the significance of the detected mean differences between the feedback modality and each of the eight evaluation modalities. Table 5 summarizes the most important findings. Both the equal variance assumed and equal variance not assumed versions of the test gave the same results. The table shows the equal variance not assumed version of the p-values.

The results of this statistical analysis reveals that the motion and speech modalities were much better than the stop modality in all dimensions except

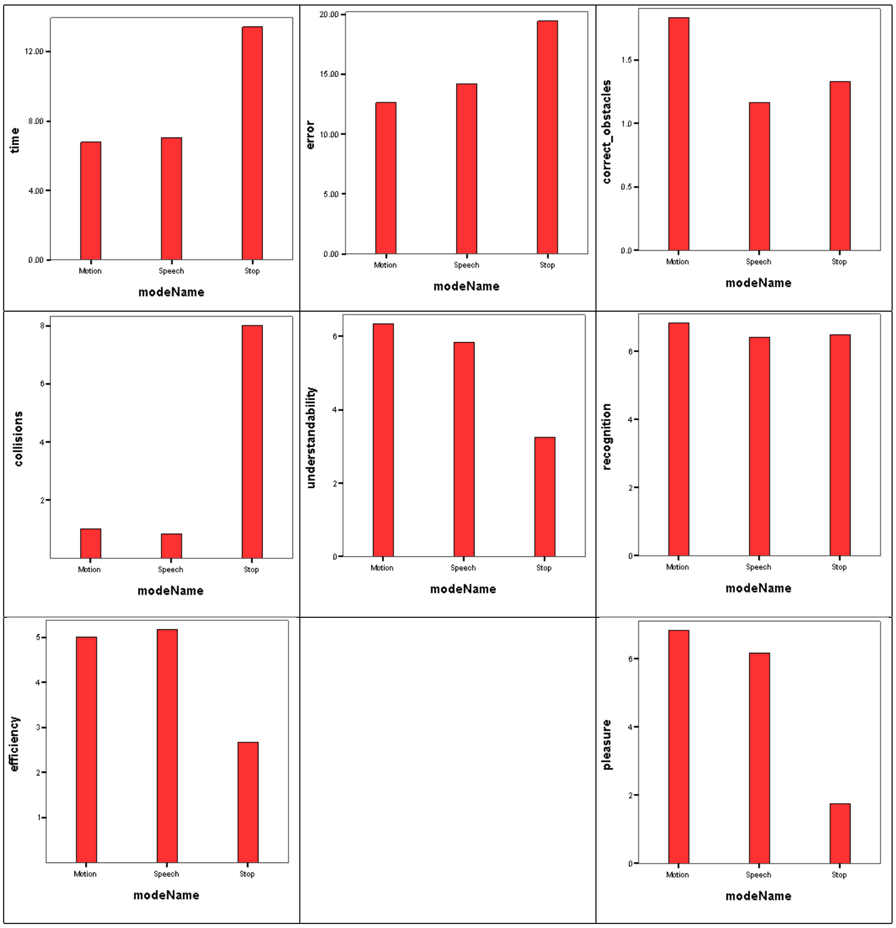


Fig. 2. Mean of evaluation dimensions for each feedback modality

Table 5. p-values of Factorial t-test analysis. The statistically significant mean differences are shown in bold.

Dimension	Speech-Stop	Motion-Stop	Motion-Speech
Time	**<0.001**	**<0.001**	0.645
Error	**0.005**	**<0.001**	0.367
Correct Object Id.	0.383	0.374	0.098
Collisions	**<0.001**	**<0.001**	0.610
Understandability	**0.002**	**<0.001**	0.475
Recognition	0.906	0.649	0.551
Efficiency	**0.003**	**0.007**	0.836
Pleasure	**<0.001**	**<0.001**	0.354

in the correct object identification dimension. The reason that there were no significant difference in this modality is due to the fact that the feedback given by the robot just describes its *intention* and did not try to describe the environment.

Most subjects selected the motion modality as their preferred modality (8 users compared to only 3 users who preferred the speech modality) while only one subject selected the stop modality although her completion time and error were much higher in the stop modality because it gave her more sense of *control.* The superiority of the motion modality in this dimension can be attributed to the fact that the cognitive load it gave to the main operator was less than the speech mode because it gives the direction directly.

An interesting finding was detected in the subjects' choice of the fastest episode: In 10 cases the subjects selected the last episode as the fastest one although in 40% of them it was the *slowest* episode. In the remaining two cases the users selected the correct fastest episode (it was the motion modality in both cases). Detailed analysis showed that there is a significant correlation between being the last session and being chosen as the fastest one while *surprisingly* there were *no* correlation between this choice and being the actual fastest episode. This phenomena can be attributed in part to the increased self-confidence of the main operators that is gained after playing the game for a while, and in part to his/her increased involvement in the game.

Similar analysis of the effect of the order at which the episodes were conducted and the day of conduction revealed no significant difference in any dimension except the recognition dimension which shows mildly significant ($p<0.01$) improvement from the first to the last day of the experiment which is caused by the increased familiarity of the WOZ operator and does not affect the results. This finding shows that the experimental procedure followed could effectively remove any dependence of the results except on the modality.

6 Conclusions and Future Work

In this paper we presented the TalkBack experiment designed to assess the usefulness of different feedback modalities that can be used in a miniature, non-humanoid robot. The analysis of the results showed that both speech and motion based modalities were more effective than just stopping in all dimensions of evaluation while there is no significant difference between them. This encouraging result, if confirmed by larger scale experiments, can be the basis to design feedback mechanisms for non-humanoid robots that are more natural and effective.

In the future the TalkBack experiment will be conducted in a larger scale using more main operators, and testing more kinds of feedback modalities like using LEDs or nonverbal sounds in conjunction to the motion modality. Combining verbal and motion cues will also be considered in the future research.

References

1. Mohammad, Y.F.O., Nishida, T.: Intention thorugh interaction: Towards mutual intention in human-robot interactions. In: IEA/AIE 2007 conference, pp. 115–125 (2007)
2. Mohammad, Y.F.O., Nishida, T.: Eica: Combining interactivity with autonomy for social robots. In: SID2007. International Workshop of Social Intelligence Design 2007, pp. 227–236 (2007)
3. Breazeal, C., Kidd, C.T.A.H.G.B.M.: Effects of nonverbal communication on efficiency and robustness in human-robot teamwork. In: IROS 2005. 2005 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 708–713 (2005)
4. Fukuda, T., Jung, M.J., Nakashima, M.A.F.H.Y.: Facial expressive robotic head system for human-robot communication and its application in home environment. In: Proceedings of the IEEE, vol. 92, pp. 1851–1865 (2004)
5. Nicolescu, M.N., Mataric, M.J.: Learning and interacting in human robot domains. IEEE Transactions on Systems, man, and Cybernetics-part A: Systems and Humans 31(5), 419–429 (2001)
6. Mohammad, Y.F.O., Nishida, T.: A new, HRI inspired, view of intention. In: AAAI 2007. Workshop on Human Implications of Human-Robot Interactions, pp. 21–27 (2007)

Using Viewing Time for Theme Prediction in Cultural Heritage Spaces

Fabian Bohnert and Ingrid Zukerman

Faculty of Information Technology
Monash University
Clayton, Victoria 3800, Australia
{Fabian.Bohnert,Ingrid.Zukerman}@infotech.monash.edu.au

Abstract. Visitors to cultural heritage sites are often overwhelmed by the information available in the space they are exploring. The challenge is to find items of relevance in the limited time available. Mobile computer systems can provide guidance and point to relevant information by identifying and recommending content that matches a user's interests. In this paper we infer implicit ratings from observed viewing times, and outline a collaborative user modelling approach to predict a user's interests and expected viewing times. We make predictions about viewing themes (item sets) taking into account the visitor's time limit. Our model based on relative interests with imputed ratings yielded the best performance.

1 Introduction

The information available in cultural heritage spaces is often overwhelming to visitors. As receptivity and time are limited, the enormous amounts of information presented can leave the visitor trapped in a situation of information overload. Additionally, different interests have niches in different areas of the space. Hence, uninformed decisions about what to view can lead to a suboptimal experience. The challenge is to find relevant content in the limited time available. Mobile context-aware computer systems can provide guidance and support by mediating the presented information to the user. By finding and recommending content that matches the user's interests and preferences, they can help ease the access to information.

In this paper we describe how implicit ratings can be inferred from observed viewing times, and outline a collaborative user modelling approach to predict a user's interests and expected viewing times. Eventually, our predictions will be used to compose and recommend an ordered sequence of items (tour) that is tailored to the user's preferences. This paper addresses the simplified problem of predicting a set of items (theme) given the user's time limit.

The paper is organised as follows. In Section 2 we outline related work. Our approach is described in Sections 3 and 4. In Section 5 we evaluate our work, followed by our conclusions in Section 6.

2 Related Research

Our work is at the intersection of statistical user modelling and personalised guide systems for cultural heritage spaces.

Personalised guide systems have often employed adaptable user models requiring the visitor to explicitly state his/her interests in some form. Less attention has been paid to predicting preferences from non-intrusive observations and to utilising adaptive user models that do not require explicit user input. In the cultural heritage domain, adaptive user models have usually been updated from the user's interactions with the system, with a focus on adapting content presentation rather than recommending item themes. For example, *HyperAudio* [1] dynamically adapts the presentation content and hyperlinks to stereotypical assumptions about the user, and to what the user has already accessed and seems interested in. The *Hippie/HIPS* projects [2] were concerned with the development of an electronic handheld guide for navigating both the physical space as well as the informational space when visiting a museum. The system provided the user with personalised and contextual information gathered from diverse sources including interaction history. Rules were used to reason about the user's interests and knowledge, and a domain hierarchy was exploited for content matching. The augmented audio reality system for museums *ec(h)o* [3] treats user interests as dynamic, and also adapts its user model on the basis of interactions with the system to deliver personalised information associated with exhibits via audio display. Most systems, including those described above, rely on *knowledge-based user models*, which require an explicit, engineered representation of the domain knowledge, e. g., an ontological representation. Little emphasis has been put on statistical user models [4,5] as a basis for reasoning about a user's needs.

The two main techniques used for statistical user models are content-based and collaborative. *Content-based techniques* base their predictions upon the assumption that a user's previous preferences or interests are reliable indicators for his/her future behaviour. Their main shortcoming is that the features selected when building a content-based model have a substantial effect on the usefulness of this model [4]. *Collaborative techniques* exploit the assumption that users who have agreed in their behaviour in the past will agree in the future, making predictions based on user-to-user similarities. The greatest strength of collaborative approaches is that they are independent of any representation of the items being recommended, and work well for complex objects such as music and films, where variations in taste are responsible for much of the variations in preferences. However, being dependent on an overlap in ratings across users, collaborative techniques have difficulty when the space of ratings is sparse (*sparsity problem*).[1]

In this paper we propose a collaborative approach to predict a user's interests and expected viewing times based on non-intrusive observations of the user's previous positions and viewing times. In particular, we do not assume that interactions with the system can be recorded.

[1] Refer to [4,6] for a detailed analysis of advantages and disadvantages of the different modelling techniques.

Table 1. Notation

$I = \{1, \ldots, M\}$	set of users, where M is the number of users
$J = \{1, \ldots, N\}$	set of items, where N is the number of items
$n_{ij} = \mathbb{1}(\text{user } i \text{ has accessed item } j)$	access indicator
$n_{i\cdot} = \sum_{j \in J} n_{ij}$	number of items accessed by user i
$n_{\cdot j} = \sum_{i \in I} n_{ij}$	number of users who accessed item j
$n_{\cdot\cdot} = \sum_{i \in I,\, j \in J} n_{ij}$	total number of accesses over all users and items
$v_{ij} = \mathbb{1}(\text{user } i \text{ has rated item } j)$	indicator of implicit rating
$v_{i\cdot}$, $v_{\cdot j}$ and $v_{\cdot\cdot}$	analogously to $n_{i\cdot}$, $n_{\cdot j}$ and $n_{\cdot\cdot}$
a	active user

3 Inferring Interests from Viewing Times

Non-intrusive adaptive user modelling requires the processing of observations or sensing information. As stated above, we assume that the only information we can sense is the user's physical location and the time spent there, which corresponds well to a real-world setting. Thus, while user a proceeds with his/her visit, we obtain an ordered sequence of viewing times t_{aj_1}, t_{aj_2}, ... for items j_1, j_2, ... respectively.

In an information-seeking context, users in general view relevant information for a longer time than irrelevant information. Thus, viewing time correlates positively with preference and interest. However, viewing time is also positively correlated with item complexity [7,8].[2] Additionally, viewing times vary over different users depending on the visitor type (Section 5.1) and the time available for the visit. Hence, for the purpose of comparing the preferences of different users for different items, the observed viewing times cannot be used directly, but must be transformed into a measure that takes these factors into account. At present we consider item complexity and available time, leading to the following definition of *relative interest* (notation in Table 1):

Definition 1. Let t_{ij} denote the time that user i spent at item j. Moreover, let $\bar{t}_{i\cdot} = \frac{1}{n_{i\cdot}} \sum_{j \in J} n_{ij} t_{ij}$ denote the *average viewing time of visitor* i, and let $\bar{t}_{\cdot j} = \frac{1}{n_{\cdot j}} \sum_{i \in I} n_{ij} t_{ij}$ be the *average time that was spent at item* j. Further, let $\bar{t}_{\cdot\cdot} = \frac{1}{N} \sum_{j \in J} \bar{t}_{\cdot j}$ be the *average viewing time of the average visitor.*

Then

$$r_{ij} = \frac{t_{ij}}{\bar{t}_{i\cdot}} - \frac{\bar{t}_{\cdot j}}{\bar{t}_{\cdot\cdot}} \tag{1}$$

is the *relative interest of user i in item j.*

[2] Viewing time was also found to be negatively correlated with familiarity, positively correlated with novelty and to decrease from beginning to end within a sequence of stops [7,8]. These complications are not yet considered in our model.

The first term measures user i's viewing time of item j relative to his/her average viewing time, and the second term indicates the average time that was spent at item j relative to the average viewing time of the average visitor. Hence, r_{ij} measures whether user i is (relative to his/her average viewing time) more or less interested in item j than the average visitor.[3]

4 Predicting Interests and Expected Viewing Times

Since relative interest is a function of viewing time which measures user interest, it can serve as a predictor of the user's preferences. Assuming that visitors with similar relative interests have similar taste and informational needs, we propose a collaborative user model based on relative interest. This user model is utilised to estimate the user's item preferences and expected viewing times.

Assume we have tracked visitors when they proceeded through the space and accessed information items. These visitor pathways annotated with time stamps are transformed into an *(implicit) rating matrix* $\boldsymbol{R} = (r_{ij})_{M,N} \in \mathbb{R}^{M \times N}$ by calculating the relative interests $r_{ij} = t_{ij}/\overline{t}_{i\cdot} - \overline{t}_{\cdot j}/\overline{t}_{\cdot\cdot}$ for all combinations of users i and items j that occurred.

In a non-personalised approach, the arithmetic mean $\overline{r}_{\cdot j}$ of item j's ratings serves as a prediction $\hat{r}_{aj}$ for the relative interest r_{aj} of user a in item j. In a personalised approach, the active user's relative interest in an item may be predicted as described below (a personalised approach can fall back on $\overline{r}_{\cdot j}$ whenever it is not possible to make a personalised prediction).

Let $M_a = \{i \in I,\, i \neq a : \sum_{j \in J} v_{aj} v_{ij} \geq 3\}$ be the *set of mentors of the active user* a (notation in Table [1]). Let further $\tilde{M}_{aj} = \{i \in M_a : s_{ai} v_{ij} > 0\}$ denote the *set of his/her item mentors for item* j, where s_{ai} is the similarity of user a and user i.[4] Then r_{aj} can be predicted as the weighted mean incorporating user i's rating in proportion to his/her similarity to the active user a [10]:

$$\hat{r}_{aj} = \begin{cases} \frac{\sum_{i \in \tilde{M}_{aj}} s_{ai} r_{ij}}{\sum_{i \in \tilde{M}_{aj}} s_{ai}} & \text{for } \tilde{M}_{aj} \neq \varnothing \\ \overline{r}_{\cdot j} & \text{for } \tilde{M}_{aj} = \varnothing \end{cases} \tag{2}$$

Any consistent measure that is large for close users and small for distant users can be utilised to measure the similarities s_{ai}. One of the most prominent measures is *Pearson's correlation coefficient*, which expresses the linear correlation of the ratings of two users [10,11]:

$$s_{ai} = \begin{cases} \frac{\sum_{j \in J} v_{aj} v_{ij} (r_{aj} - \overline{r}^{i}_{a\cdot})(r_{ij} - \overline{r}^{a}_{i\cdot})}{\sqrt{\sum_{j \in J} v_{aj} v_{ij} (r_{aj} - \overline{r}^{i}_{a\cdot})^2 \sum_{j \in J} v_{aj} v_{ij} (r_{ij} - \overline{r}^{a}_{i\cdot})^2}} & \text{if denominator} > 0 \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

where $\overline{r}^{i}_{a\cdot}$ is the arithmetic mean of all ratings r_{aj} of the active user a for which corresponding ratings were also given by user i, and $\overline{r}^{a}_{i\cdot}$ is defined analogously.

[3] Other measures of interest are possible, e. g., the ratio of the first and second term.

[4] In real-world environments with tens of thousands of users, it is advisable to limit the number of potential mentors [9].

In summary, in order to estimate the relative interest $\hat{r}_{aj}$, we first calculate the user-to-user similarities s_{ai} for each mentor in M_a, and determine user a's set of item mentors $\tilde{M}_{aj}$. The desired estimate is then calculated using Equation 2. However, in order to proceed as described, we first need to calculate the active user's relative interests for items already visited, as they are essential for calculating the similarities s_{ai}. This is done as follows.

Estimating $\hat{r}_{aj}(f_T)$. Let T be the total time which user a has available for the entire visit, and let f_T denote the fraction of time already used up. Hence, the user has $T(1 - f_T)$ time left for the remaining visit. Further, let $n_{i\cdot}(f_T)$ be the number of items accessed by user i in the first f_T's fraction of his/her visit. User a's estimated relative interest $\hat{r}^{\text{visit}}_{aj}(f_T)$ in a visited item j at time f_T then is

$$\hat{r}^{\text{visit}}_{aj}(f_T) = \frac{t_{aj}}{T/\hat{n}_{a\cdot}(f_T)} - \frac{\bar{t}_{\cdot j}}{\bar{t}_{\cdot\cdot}} \tag{4}$$

where $\hat{n}_{a\cdot}(f_T) = n_{a\cdot}(f_T) + \frac{1}{M}\sum_{i\in I}(n_{i\cdot} - n_{i\cdot}(f_T))$ is the *total number of items to be visited estimated at time* f_T, calculated as user a's number of visited items plus the number of items visited in the last $(1 - f_T)$'s visit fraction averaged over all visitors in the database. For all unvisited items j, $\hat{r}_{aj}(f_T)$ can then be predicted by means of collaborative filtering as described in Equation 2. The similarities s_{ai} are calculated by means of Equation 3, using the estimated relative interest values $\hat{r}^{\text{visit}}_{aj}(f_T)$ for the active user a and visited items j, and the respective relative interest values r_{ij} of user i.

We have derived estimators $\hat{n}_{a\cdot}(f_T)$ and $\hat{r}_{aj}(f_T)$ for the number of stops and the relative interests of user a at time f_T, but we still must estimate the time that user a will require in order to view item j: the *expected viewing time* $\hat{t}_{aj}(f_T)$.

Estimating $\hat{t}_{aj}(f_T)$. For all items j not yet visited, the expected viewing times $\hat{t}_{aj}(f_T)$ can be estimated as follows.

$$\hat{t}_{aj}(f_T) = \begin{cases} \frac{T}{\hat{n}_{a\cdot}(f_T)} \cdot \frac{\sum_{i\in\tilde{M}^{(t)}_{aj}} s_{ai}\frac{t_{ij}}{\bar{t}_{i\cdot}}}{\sum_{i\in\tilde{M}^{(t)}_{aj}} s_{ai}} & \text{for } \tilde{M}^{(t)}_{aj} \neq \varnothing \\ \frac{T}{\hat{n}_{a\cdot}(f_T)} \cdot \frac{1}{n_{\cdot j}}\sum_{i\in I} n_{ij}\frac{t_{ij}}{\bar{t}_{i\cdot}} & \text{for } \tilde{M}^{(t)}_{aj} = \varnothing \end{cases} \tag{5}$$

where $\tilde{M}^{(t)}_{aj} = \{i \in \tilde{M}_{aj} : t_{ij} > 0\}$ is the set of user a's item mentors with positive viewing times t_{ij} for item j.

A recommendation to the active user a at time f_T comprises the set of non-visited items with the highest predicted relative interest values $\hat{r}_{aj}(f_T)$, such that the sum of their expected viewing times $\hat{t}_{aj}(f_T)$ does not exceed $T(1 - f_T)$.

5 Evaluation

Thanks to a cooperation with Museum Victoria, we could access data from several observational studies conducted at the Melbourne Museum. For our evaluation we used data collected at the Marine Life exhibition in 2006. The Marine

Life exhibition comprises 56 exhibits in four rooms. For this study, 44 visitors were observationally tracked and their viewing times recorded. Although the Marine Life exhibition data set also contains the demographics of the visitors, we focused on data that could be sensed easily, i. e., the time-annotated pathways. On average, a visitor accessed only 9.11 out of the 56 items. Hence, this data set provides a good basis for evaluating our prediction approach. We now briefly describe our main results.

5.1 Visitor Types and Relative Interest

Several attempts have been made to classify museum visitors into different categories. Adopting a socio-semiotic perspective, Véron and Levasseur [12] distinguish four visitor categories: *ant*, *butterfly*, *fish* and *grasshopper*. Sparacino's alternative classification distinguishes three types [13]: *busy*, *selective* and *greedy*. The busy type aims to get an overview of the principal items and sees a little of everything, whereas the selective visitor selects only a few preferred items but studies those in depth. The greedy visitor wants to see and know as much as possible, and does not have much of a time constraint. The ant visitor roughly corresponds to the greedy visitor and the grasshopper to the selective type, whereas the butterfly and fish are unified in the busy type. The different classifications have been justified through qualitative analyses [3,12,13]. Recently, Zancanaro *et al.* [14] empirically validated Véron and Levasseur's classification in a quantitative study.

We used the Marine Life exhibition data set to verify the visitor classifications. For each visitor we calculated the number of stops ($\#stops$) and the average time spent at an item ($mtime$). After standardising and normalising the tuples, we used hierarchical complete-linkage clustering with Euclidean distance to obtain a cluster tree. The obtained dendrogram—the plot of the hierarchical binary cluster tree—showed three clusters to be the most likely formation. Figure 1 depicts these results. Despite the small sample, we believe that the different visitor types can be identified. A more detailed analysis including more than just two features will be conducted when more data becomes available.

Thus, in order to cater for different visitor types, the user model should be adaptive with respect to the user's average viewing time and total number of stops, and predicted sets of items should reflect these characteristics. The approach described in Sections 3 and 4 implicitly satisfies these requirements.

We calculated the average relative interests $\overline{r}_{i\cdot} = \frac{1}{v_{i\cdot}} \sum_{j \in J} v_{ij} r_{ij}$ for each of the visitors in the Marine Life exhibition data set using a dense rating matrix $\boldsymbol{R}$.[5] We then sorted the visitors in ascending order of their average relative interests $\overline{r}_{i\cdot}$, and compared this sorted list to a domain expert's classification and our own clustering results. The comparison revealed the following relations: busy $\prec$ selective $\prec$ greedy. This indicates that our measure of relative interest is meaningful and separates well the different visitor types.

[5] Section 5.2 describes how values were imputed to obtain a dense rating matrix $\boldsymbol{R}$.

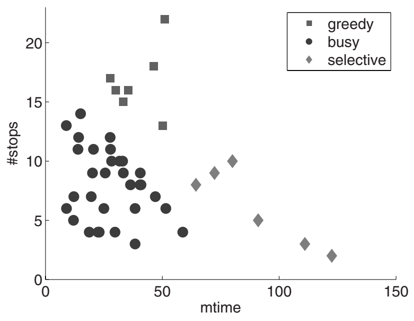


Fig. 1. Clustering results (*mtime* vs. *#stops*)

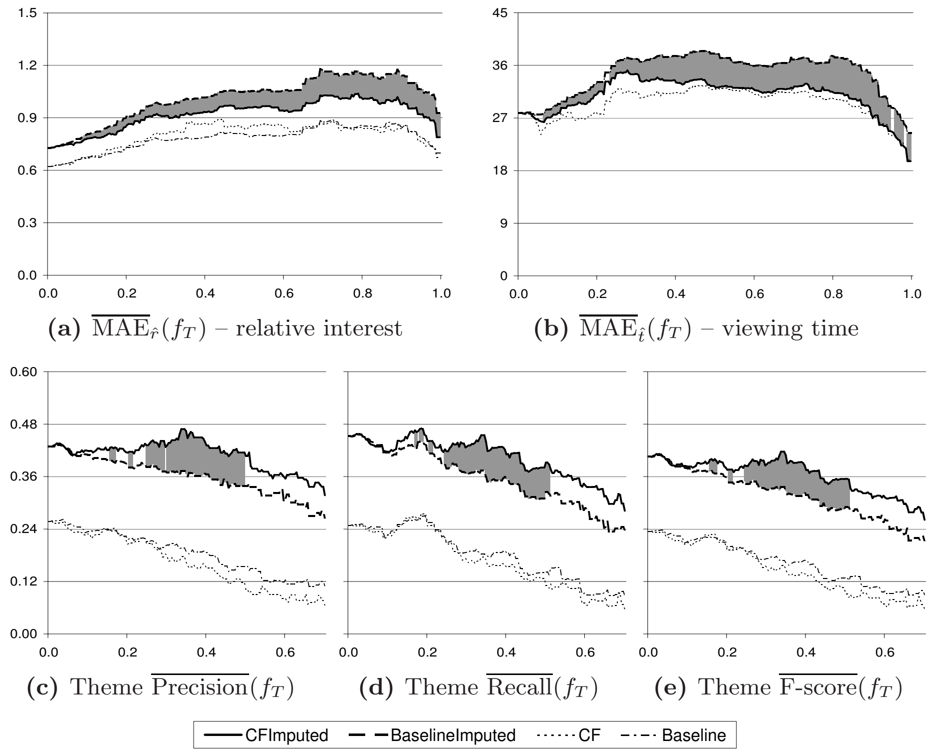
5.2 Quantitative Evaluation of Prediction Quality

The Marine Life exhibition data set was also used to evaluate our predictive approach. Four different algorithms were tested: `CF`, `Baseline`, `CFImputed` and `BaselineImputed`. `CF` is the collaborative filtering approach outlined in Section 4, and `Baseline` is a non-personalised approach based on calculating non-personalised averages (Equations 2 and 5 bottom cases). In both variants we used a sparse rating matrix $\boldsymbol{R}$ obtained by calculating the relative interests $r_{ij} = t_{ij}/\overline{t}_{i\cdot} - \overline{t}_{\cdot j}/\overline{t}_{\cdot\cdot}$ for all combinations of users i and items j that occurred. For the variants `CFImputed` and `BaselineImputed`, calculations were performed as for `CF` and `Baseline` respectively, but we attributed $t_{ij} = 0$ to all non-accessed items. That is, we assumed that not having accessed an item implies no interest, yielding $r_{ij} = -\overline{t}_{\cdot j}/\overline{t}_{\cdot\cdot}$ in case of no access. This assumption leads to a dense matrix $\boldsymbol{R}$ of relative interests (i. e., $v_{ij} = 1$ for all i and j, while $v_{ij} = n_{ij}$ for the non-imputed variants), and hence addresses the sparsity problem (Section 2). Clearly, this is a simplistic assumption, since not having visited an item might not mean no interest but just no time. A more realistic assumption would attribute no interest in item j only if the visitor has walked past this item and disregarded it. However, such an assumption was not necessary for the Marine Life exhibition due to its physical layout.

We used *mean absolute error (MAE)*, and *precision*, *recall* and *F-score* to compare the algorithms.[6] Performance evaluation was carried out using leave-one-out cross validation as follows. For each of the 44 visitor pathways we simulated the visit over time and added visited items to the visitor's item history at the appropriate fraction f_T of the visit according to the recorded pathway.[7] Based on the user's history, we applied our four algorithms to predict the number of items to be viewed $\hat{n}_{a\cdot}(f_T)$, relative interest values $\hat{r}_{aj}(f_T)$, and viewing times $\hat{t}_{aj}(f_T)$. The predicted set of items to be visited (with cardinality N_s) was

[6] Refer to [15] for a detailed analysis and comparison of evaluation strategies and metrics in the context of collaborative recommender systems.

[7] We recorded events at time fractions of the visits (and not absolute times), because different visits have different lengths.



(a) $\overline{\text{MAE}}_{\hat{r}}(f_T)$ – relative interest **(b)** $\overline{\text{MAE}}_{\hat{t}}(f_T)$ – viewing time

(c) Theme $\overline{\text{Precision}}(f_T)$ **(d)** Theme $\overline{\text{Recall}}(f_T)$ **(e)** Theme $\overline{\text{F-score}}(f_T)$

Fig. 2. Performance of the four models over time f_T

the subset of items with the highest predicted relative interests selected from the set of non-visited items, such that the sum of their expected viewing times $\hat{t}_{aj}(f_T)$ did not exceed the visitor's time limit $T(1 - f_T)$. We assumed the set of relevant items (with cardinality N_r) to be the set of items accessed in the remainder of the visit. Hence, the number of selected items N_s and relevant items N_r decreases over time. With N_{sr} being the number of selected relevant items, we calculated precision $P = N_{sr}/N_s$, recall $R = N_{sr}/N_r$ and F-score $= \frac{2PR}{P+R}$ [15] for each f_T. Precision, recall and F-score were averaged over all visitors for each f_T to calculate $\overline{\text{Precision}}(f_T)$, $\overline{\text{Recall}}(f_T)$ and $\overline{\text{F-score}}(f_T)$. Similarly, $\overline{\text{MAE}}_{\hat{r}}(f_T)$ and $\overline{\text{MAE}}_{\hat{t}}(f_T)$ are the averages of the mean absolute errors of $\hat{r}$ and $\hat{t}$ respectively, averaged over all visitors for each fraction f_T.

Figure 2 shows the progression of these quantities over time.[8] For each f_T we performed paired two-tailed t-tests ($\alpha = 0.05$) to test whether the difference between the respective values of `CFImputed` and `BaselineImputed` is statistically significant. The intervals for which `CFImputed` and `BaselineImputed` are significantly different are highlighted in the graphs. Precision, recall and F-score

[8] The variants `Baseline` and `BaselineImputed` predict $\hat{t}_{aj}(f_T)$ identically. Hence, the corresponding mean absolute errors $\overline{\text{MAE}}_{\hat{t}}(f_T)$ overlap (Figure 2b).

are shown only for the first 70% of the visits, because for $f_T > 0.7$ at least one visitor has arrived at the last stop (i. e., the number of relevant items is zero for at least one visitor).

For all variants, the mean absolute error for predicted relative interests and viewing times increases initially, then stabilises, and finally decreases towards the end of the visit (Figures 2a and 2b). This may be explained by the fact that as time progresses, on one hand, fewer items are being predicted from the pool of unvisited items, while on the other hand, the quality of the collaborative model improves as more evidence is gathered. Also, performance appears to degrade, since precision, recall and F-score decrease over time (Figures 2c, 2d and 2e). However, this is a result of calculating ratios between a decreasing numerator (N_{sr}) and decreasing denominators (N_s and N_r). Although it appears that `CF`'s precision, recall and F-score are often lower than `Baseline`'s (Figures 2c, 2d and 2e), these differences are almost always not statistically significant (based on paired two-tailed t-tests with $\alpha = 0.05$ for each f_T).

Overall, our results show that (1) precision, recall and F-score are much higher (nearly double) for the imputed variants than for the non-imputed variants; and (2) `CFImputed` outperforms all other tested approaches, with significantly lower mean absolute errors than `BaselineImputed` for both predicted relative interests and viewing times (Figures 2a and 2b), and significantly better values for precision, recall and F-score for most f_Ts (Figures 2c, 2d and 2e).

6 Conclusions and Future Research

We outlined a collaborative user modelling approach to predict a user's interests and expected viewing times. Our results show that our measure of relative interest is meaningful and captures well the different types of visitors to cultural heritage spaces. A quantitative evaluation of four prediction models on a small data set showed that the collaborative model based on relative interests with imputed ratings yields the best performance.

To date, we have taken into account only two factors correlated with viewing time for calculating relative interest: item complexity and available time. Other factors will be considered in the future. Predictions currently comprise the set of non-visited items with the highest predicted relative interest values such that the sum of their expected viewing times does not exceed the visitor's time limit. This over-simplified model does not consider the physical distances between items. In the future, we plan to take into account the physical layout of the space together with predicted themes (unordered sets of items) in order to recommend personalised tours (ordered sequences of items). Also, in the near future, we will validate our results on larger data sets.

Acknowledgements. This research was supported in part by Australian Research Council grant DP0770931. The authors thank Liz Sonenberg from The University of Melbourne for her input on this research. Thanks also go to

Carolyn Meehan and Alexa Reynolds from Museum Victoria for fruitful discussions, their support of this study, and the data set.

References

1. Petrelli, D., Not, E.: User-centred design of flexible hypermedia for a mobile guide: Reflections on the HyperAudio experience. User Modeling and User-Adapted Interaction 15(3-4), 303–338 (2005)
2. Oppermann, R., Specht, M.: A context-sensitive nomadic information system as an exhibition guide. In: Thomas, P., Gellersen, H.-W. (eds.) HUC 2000. LNCS, vol. 1927, pp. 127–142. Springer, Heidelberg (2000)
3. Hatala, M., Wakkary, R.: Ontology-based user modeling in an augmented audio reality system for museums. User Modeling and User-Adapted Interaction 15(3-4), 339–380 (2005)
4. Zukerman, I., Albrecht, D.W.: Predictive statistical models for user modeling. User Modeling and User-Adapted Interaction 11(1-2), 5–18 (2001)
5. Albrecht, D.W., Zukerman, I.: Introduction to the special issue on statistical and probabilistic methods for user modeling. User Modeling and User-Adapted Interaction 17(1-2), 1–4 (2007)
6. Burke, R.: Hybrid recommender systems: Survey and experiments. User Modeling and User-Adapted Interaction 12(4), 331–370 (2002)
7. Berlyne, D.E.: Conflict, Arousal, and Curiosity. McGraw-Hill, New York (1960)
8. Heinrich, P.: "Free looking time:" Eine Methode zur Präferenzbestimmung. Psychologie und Praxis 14(2), 79–93 (1970)
9. Fink, J., Kobsa, J.: User modeling for personalized city tours. Artificial Intelligence Review 18(1), 33–74 (2002)
10. Shardanand, U., Maes, P.: Social information filtering: Algorithms for automating "word of mouth". In: CHI 1995. Proceedings of the 1995 SIGCHI Conference on Human Factors in Computing Systems, pp. 210–217 (1995)
11. Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., Riedl, J.: GroupLens: An open architecture for collaborative filtering of netnews. In: CSCW 1994. Proceedings of the 1994 ACM Conference on Computer Supported Cooperative Work, pp. 175–186 (1994)
12. Véron, E., Levasseur, M.: Ethnographie de l'exposition. Bibliothèque Publique d'Information, Centre Georges Pompidou, Paris (1983)
13. Sparacino, F.: The Museum Wearable – Real-time sensor-driven understanding of visitors' interests for personalized visually-augmented museum experiences. In: MW 2002. Proceedings of the 11th International Conference on Museums and the Web (2002)
14. Zancanaro, M., Kuflik, T., Boger, Z., Goren-Bar, D., Goldwasser, D.: Analyzing museum visitors' behavior patterns. In: UM 2007. Proceedings of the 11th International Conference on User Modeling (2007)
15. Herlocker, J.L., Konstan, J.A., Terveen, L.G., Riedl, J.T.: Evaluating collaborative filtering recommender systems. ACM Transactions on Information Systems 22(1), 5–53 (2004)

Collaborative Tagging in Recommender Systems

Ae-Ttie Ji, Cheol Yeon, Heung-Nam Kim, and Geun-Sik Jo

Intelligent E-Commerce Systems Laboratory, Inha University, Korea
{aerry13, entireboy, nami}@eslab.inha.ac.kr, gsjo@inha.ac.kr

Abstract. This paper proposes a collaborative filtering method with user-created tags focusing on changes of web content and internet services. Collaborative tagging is employed as an approach in order to grasp and filter users' preferences for items. In addition, we explore several advantages of collaborative tagging for future searching and information sharing which is used for automatic analysis of user preference and recommendation. We present empirical experiments using real dataset from *del.icio.us* to demonstrate our algorithm and evaluate performance compared with existing works.

1 Introduction

The prevalence of digital devices and the development of Internet technologies and services enable end-users to be a producer as well as a consumer of media content. Even in a single day, an enormous amount of content including digital video, blogging, photography and wikis is generated on the web. It's getting more difficult to make a recommendation to a user about what he/she prefers among these items automatically because of not only their huge amount, but also the difficulty of automatically grasping of their meanings.

Collaborative filtering (CF) is an efficient approach to treat the above issues [3, 5, 6, 11, 13]. CF has an advantage over content-based filtering which is the ability to filter any type of items, e.g. text, music, videos and photos [6]. Because the filtering process is only based on historical information about whether or not a target user has preferred an item before, analysis of actual content itself is not necessarily required.

Collaborative tagging describes the process that allows many users to annotate content with descriptive keywords, i.e. tags [1, 12, 15]. Tagging is not new, but has recently become useful and popular as one effective way of classifying items for future search, sharing information and filtering [1, 15]. In terms of user-created tags, they imply users' preferences and opinions about items as well as metadata about them.

In our research, we propose a CF method with user-created tags focusing on changes of web content and internet services. Collaborative tagging is employed as an approach in order to grasp and filter users' preferences for items. The next section describes an overview of recent studies related to tagging. Our main contribution is a novel approach of recommendation systems with collaborative tagging. This approach is described in section 3. Section 4 presents the effectiveness of our approach through the experiments comparing our approach with existing works using *del.icio.us* (http://del.icio.us/) data. We conclude with a discussion and future directions.

2 Collaborative Tagging and Folksonomy

Collaborative tagging is the practice of allowing any user to freely annotate to content with any kind of tags [1, 15]. In some circumstances, such as the web, where there is no "librarian" to classify items or there are too many items to classify by a single authority, collaborative tagging is one of the most useful ways of categorizing or indexing content [1]. Moreover, tags are directly published and discussed on the web and may be applied to any kinds of items, even people [10]. Collaborative tagging can play a key role in sharing content in social networks [15].

Collaborative tagging is described as "folksonomy," in contrast with typical "taxonomy," even though there is some debate about the appropriateness of this term [4]. In contrast with taxonomy, tagging performs a horizontal and inclusive way for classification and therefore can have an advantage over hierarchical taxonomy in some cases. In taxonomy, a category with a more general concept includes more specific ones. Even though a hierarchical category assures a user that all the items exist in one corresponding stable place, the user cannot be sure that all relevant items are returned by a query. To avoid fruitless searching, the user needs to check multiple locations. Unlike a hierarchical search, in a collaborative tagging system such items can be annotated with a variety of terms simultaneously; general tags and specific ones. In addition, tags can filter out all relevant items and return only those items tagged with those tags. As users can provide tags without any intricate implementation, a tagging system can be an effective and easy way to help identify correct items and make search results more relevant. Golder and Huberman have discussed about such advantages over taxonomy as well as the other significant issues of tagging systems [1].

Marlow et al. define several dimensions of tagging system design according to their possible implications [2]. We will review two of them related to our work briefly. From the user's right of tagging behavior, a tagging system can be classified into self-tagging, permission-based and free-for-all. Self-tagging, where users only tag the content they created for future personal retrieval, is provided by *Technorati* (http://www.technorati.com/) and *YouTube* (http://www.youtube.com/). Like in *Flickr* (http://www.flickr.com/), permission-based tagging is provided as specifying different levels of permission to tag. These two forms of tagging are also mentioned as narrow folksonomies [12, 14], and strictly speaking, they partially or do not support *collaborative* tagging [1]. *Del.icio.us* and *Yahoo! MyWeb* (http://myweb.yahoo.com/), which provide free-to-all tagging, allow any user to tag any items. Free-to-all tagging is also known as a broad folksonomy [12, 14]. According to the aggregation of tags, a tagging system is divided into a bag-model and a set-model. A set-model does not allow any repetition of tags, and so the system shows users only the "set" of tags attached for the item (e.g., *Flickr*, *YouTube* and *Technorati*). In contrast with set-model, a bag-model system allows duplicated tags for the same item from different users (e.g., *del.icio.us*, *Yahoo! MyWeb*). Based on the statistical information of tag frequencies, the system is able to present the item with the collective opinions of the taggers.

Especially in broad folksonomy, a tag frequency of an item tends to lead a long-tail curve (i.e., power law curve or power curve) [14], as the majority of the tags attached to the item are popular or common tags of the item. These popular tags with high frequencies can represent the content of the item.

3 Recommender System with Collaborative Tagging

Fig. 1 illustrates our method with two phases: Candidate Tag Set (*CTS*) generation and probabilistic recommendation. *CTS,* which implies the latent preference of a target user, includes the tags filtered with the CF scheme. Based on *CTS*, a *Naïve Bayes Classifier* is applied to decide which items to recommend stochastically.

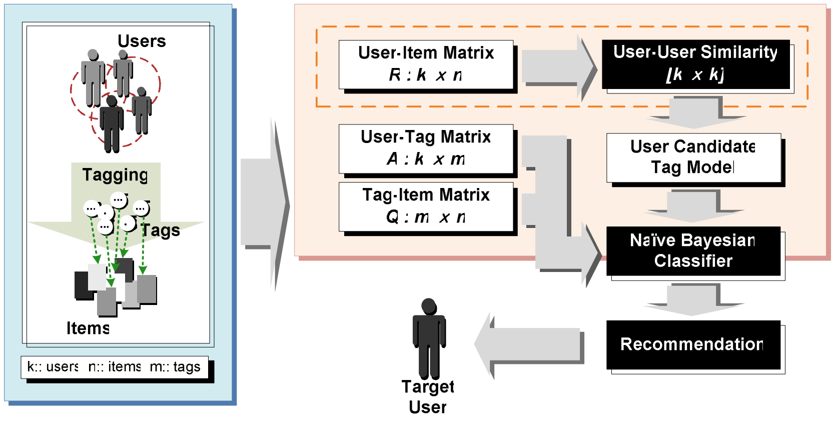


Fig. 1. Architecture of a Tag-based Collaborative Filtering System

Three matrices including preference history of users are used throughout this article.

User-item binary matrix, *R*. A list of k users $U = \{u_1, u_2, \ldots, u_k\}$ and a list of n items $I = \{i_1, i_2, \ldots, i_n\}$ can be represented as a *User-item binary matrix*, $R\ (k \times n)$. Each $R_{u,i}$ has 1 if a user u has selected (or tagged) an item i or 0 otherwise.

User-tag matrix, *A*. For a set of m tags $T = \{t_1, t_2, \ldots, t_m\}$, tag usages of k users can be represented as a *User-tag matrix*, $A\ (k \times m)$. Each $A_{u,t}$ represents the frequency of meaning how many times a user u has been tagging with a tag t.

Tag-item matrix, *Q*. *Tag-item matrix,* $Q\ (m \times n\ )$ includes tag frequency for n items. Each $Q_{t,i}$ implies the number of tag t tagged for item i by users.

3.1 Candidate Tag Set Generation

Candidate Tag Set, *CTS*. The concept of *CTS* starts from assuming that a target user is likely to prefer the items tagged with the tags, that is, the tags that have been used by similar users or by a target user before. *CTS* includes a set of tags which implied a target user's latent preference and is generated using CF scheme based on *user-tag matrix*, *A*. $CTS_w(u) = \{t_x | \ x=1,2, \ldots,w,\ t_x \in T\}$ indicates the *CTS* of user u included in the set of all tags, T and w means the number of candidate tags.

User-user Similarity. In order to find k nearest neighbor (*KNN*), cosine similarities between a target user and each users with tag frequencies of corresponding user in *user-tag matrix, A*. *KNN* includes users who have higher similarity score than the other users and means a set of users who prefer more similar tags with a target user. The similarity relationship between user *u* and *v*, *sim*(*u,v*) is defined as:

$$sim(u,v) = \cos(\vec{u}, \vec{v}) = \frac{\sum_{t\in T} A_{u,t} \cdot A_{v,t}}{\sqrt{\sum_{t\in T} (A_{u,t})^2}\sqrt{\sum_{t\in T} (A_{v,t})^2}} \tag{1}$$

$A_{u,t}$ and $A_{v,t}$ denotes the tag frequencies of user *u* and *v* each in *user-tag matrix, A*. The similarity score between two users takes a real number between 0 and 1 and the higher score a user has, the more similar he is to a target user.

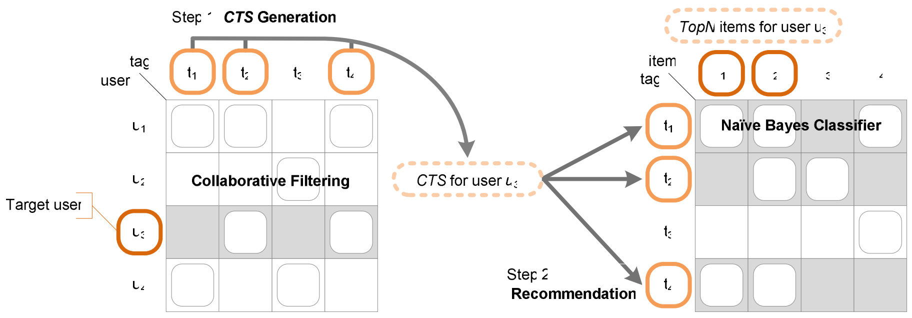


Fig. 2. Candidate Tag Set Generation via Collaborative Filtering

Tag Preference. The measurement of how much a user prefers a tag is given by [5]:

$$S_{u,t} = \sum_{o\in KNN(u)} (A_{o,t}) \cdot sim(u,o) \tag{2}$$

$S_{u,t}$ denotes the prediction value of user *u*'s preference for tag *t*. For each user *o* included in *KNN*(*u*), which is a set of *k* neighbors of user *u*, a sum of tag frequency $A_{u,t}$ weighted by *sim*(*u,o*), which is a similarity score between user *u* and *o*. The weighted sum leads to a higher prediction value for a more similar user to a target user. According to the highest-order of prediction value $S_{u,t}$, *w* tags are selected for generating $CTS_w(u)$. The algorithm for computing user-tag preferences is shown in Algotithm 1.

Recommendation from tags can partially improve the sparsity problem, which is one of the limitations of CF, due to the filtering process depending on the co-occurrence of items. For example, let us assume that Alice bookmarked two web pages, "Subway Map" and "Bus Map", Bob bookmarked "Subway Map", "Bus Map" and "Public Transportation Fares" and Cathy bookmarked "Public Transportation Fares". If Alice is a target user who needs to get a recommendation, the system provide "Public Transportation Fares" page for her from Bob, because Alice and Bob have co-bookmarked items, "Subway Map" and "Bus Map". However, the system cannot provide any items for Cathy or from her because she has no co-bookmarked items with other users, and so any similarity relationship between her and other users cannot be derived. In contrast, providing that Alice tagged "Subway Map" and "Bus Map" with *route map, public transportation* and *traffic* and Cathy tagged "Public Transportation Fares" with *public transportation* and *fare*, the system can provide the tag *fare* for Alice from Cathy because of co-tagged *public transportation*. Now, Alice can get recommendations of items tagged with *fare* as well as her own tags: *route map, public transportation* and *traffic* including “Public Transportation Fares” tagged by Cathy.

In addition, tags can be used as a more efficient means for modeling user’s preferences. Tags are able to support the detailed opinion of a user for a particular item and are easy to be changed and updated regardless of implementation. If a user’s preference is changed as time goes by, he can easily express his own opinion about the item by adding or changing tags.

Algorithm 1. Tag Preference Prediction Algorithm

```
Input: total user list U; size of KNN k; user-tag matrix A;
       user-user similarity matrix D;
Output: user-tag preference matrix S

Procedure computingTagPreference(U, k, A, D, S)
01: set all elements in matrix S to 0
02: for each $u \in U$
03:     for $i \leftarrow 1$ to r // r is row count of matrix U
04:         add $D_{u,i}$ to itemset KNN
05:     for each $x \in$ KNN // get KNN of each user
06:         if $x \neq$ among the k largest values in KNN then
07:            remove x from KNN
08:     for each $t \in T$ // compute the preference matrix S
09:         for each $x \in$ KNN
10:             $S_{u,t} \leftarrow S_{u,t} + (A_{x,t} \times D_{u,x})$
```

3.2 Item Recommendation Via Naïve Bayes Approach

Based on a *CTS* model for each user, *top-N* items are recommended stochastically with a *Naïve Bayes Classifier* [9]. *Top-N recommendation* is one of the recommendation schemes offering a target user u the ordered set of items $TopN_u$ such that $|TopN_u| \leq N$ and $TopN_u \cap I_u = \varnothing$ [7]. I_u denotes a set of items that have been selected by a user u. Given a tag instance t_j in $CTS_w(u)$ as a set of feature variables, a Bayes classifier allows us to compute the *posterior* probability $P(I = i_y | t_j)$ which tag t_j was tagged for item i_y for each possible item i_y in a set of all items $I = \{i_1, i_2, \ldots, i_n\}$. A *priori* probability $P(I = i_y)$ and a item-conditional tag distribution $P(t_j | I = i_y)$ are computed as:

$$P(I = i_y) = \frac{\sum_{u=1}^{k} R_{u,y}}{\sum_{y=1}^{n} \sum_{u=1}^{k} R_{u,y}} \quad , \quad P(t_j | I = i_y) = \frac{1 + Q_{j,y}}{m + \sum_{t=1}^{m} Q_{t,y}} \tag{3}$$

$R_{u,y}$ and $Q_{j,y}$ denotes the binary value of u-th user for y-th item in user-item matrix, R and the tag frequency of j-th tag for y-th item in tag-item matrix, Q respectively. Avoiding that $Q_{j,y}$ turn out to be zero, we use *Laplace correction* in Equation 3 [8].

Let us assume that each feature t_j is conditionally independent of every other feature, the *posterior* probability as a preference probability $P_{u,y}$ of user u with $CTS_w(u)$ for an item i_y is given by:

$$P_{u,y} = P(I = i_y)\prod_{j=1}^{w} P(t_j | I = i_y) \tag{4}$$

Finally, the ordered set of items *TopN*$_u$ with the highest $P_{u,y}$ is recommended for a target user *u* [7]. The entire recommendation process proposed is described in Algotithm 2 with *TopN*$_u$ generation.

Algorithm 2. Recommender System with Collaborative Tagging

```
Input: total user list U; user-item matrix R; user-tag matrix
       A; tag-item matrix Q; size of CTS w; size of Top-N N;
       size of KNN k; items not rated by user u L_u

Method:
01: for each u ∈ U // generate user-user similarity matrix D
02:     for each v ∈ U
03:         if v ≠ u then
04:             D_{u,v} ← sim(u, v)
05: // generate user-tag preference matrix S
06: call generatingTagPreference (U, k, A, D)
07: for each u ∈ U { // recommending items to each user
08:     TopN_u ← generatingTopNItem (u, w, N, L_u, S)
09:     Recommending Top-N items to user u
10: }

Procedure generatingTopNItem(u, w, N, L_u, S)
01: // get CTS of user u from user-tag preference matrix S
02: for i ← 1 to m // m is column count of matrix S
03:     add S_{u,i} to itemset CTS_w(u)
04: for each x ∈ CTS_w(u)
05:     if x ≠ among the w largest values in CTS_w(u) then
06:        remove x from CTS_w(u)
07: for each i_y ∈ L_u
08:     add NaiveBayesClassifier(u, CTS_w(u), i_y, Q) to TopN_u
09: for each z ∈ TopN_u // recommend Top-N items to user u
10:     if P_{u,z} = 0 ∨ P_{u,z} ≠ the N largest values in TopN_u then
11:        remove z from TopN_u
12: return TopN_u
```

4 Experimental Results

In this section, we empirically evaluate the recommendation algorithm via collaborative tagging and compare its performance against the performances of user-based CF [11] and item-based CF [7]. The system prototype was implemented using *JDK* 5.0 and *MySQL* 5.0 and experiments were performed on *Dual Xeon* 3.0 GHz, 2.5GB RAM computers.

4.1 Dataset and Evaluation Metrics

Del.icio.us is a well-known social bookmark service supporting collaborative tagging. We collected our dataset by examining the *del.icio.us* site and constructed a 1544 × 17,390 user-item binary matrix, *R,* a 1544 × 10,077 user-tag matrix, *A* and a 10,077 × 17,390 tag-item matrix, *Q*. Sparsity level, which is defined as 1-(nonzero elements / total elements) [3], of the collected dataset is as follows; user-item matrix is 0.9989 and user-tag matrix is 0.9971. The dataset was divided into two parts; a training set containing 21,653 bookmarks (80%) and a test set containg 5,413 bookmarks (20%).

Table 1. The Dataset from *del.icio.us*

users	items	tags	book marking	Tagging
1,544	17,390	10,077	27,066	44,681

The performance was measured by looking at the number of items in the test set that were also included in $TopN_u$ recommended for a target user *u* by a particular scheme, which is also called *recall* [7, 11]. Hit-ratio for each target user *u* is given by:

$$hit - ratio(u) = \frac{|Test_u \cap TopN_u|}{|Test_u|} \tag{5}$$

$Test_u$ denotes a set of items tagged by a target user u. Average *recall* of all the *k* users in test set is given by:

$$recall = \frac{\sum_{u=1}^{k} hit - ratio(u)}{k} \times 100 \tag{6}$$

4.2 Performance Evaluation of Benchmark Algorithms

The neighborhood size has significant impact on the quality of results from collaborative filtering [4]. We evaluated the quality of user-based CF [11] and item-based CF [7] based on a user-item matrix, *R*, where we varied the size of the *k*-nearest neighbor, *KNN*, from 10 to 100. We set the number of returned items *N* to 10 for each user in the test set and computed *recall* for each algorithm [3].

Fig. 3 illustrates the variation of *recall* for each algorithm. It can be observed that the recommendation quality improves as the number of *k* is increased. After 50, the increase rate of a user-based scheme diminishes. Generally, an item-based scheme performed better than a user-based one. This is caused by the sparsity of the dataset is too high to compute user-user similarities [13]. With a relatively small neighborhood size, an item-based scheme outperformed a user-based one. However, because the number of items is far lager than the number of users, computation of item-item similarities took much longer than user-user similarities.

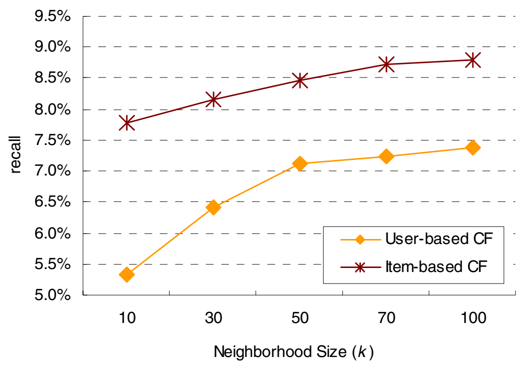


Fig. 3. The Variation of Recalls for User-based CF and Item-based CF

4.3 Experiments with Candidate Tag Set Size

We expected that the size of *CTS*, *w,* can be a significant factor affecting the quality of recommendation in our work, and so we evaluated our algorithm by measuring *recall* according to each size of *CTS* from 10 to 100. In order to obtain *CTS* based on user-tag matrix, *A*, user-based CF was used where *k* is set to 50. *N* is set to 10 through all evaluations.

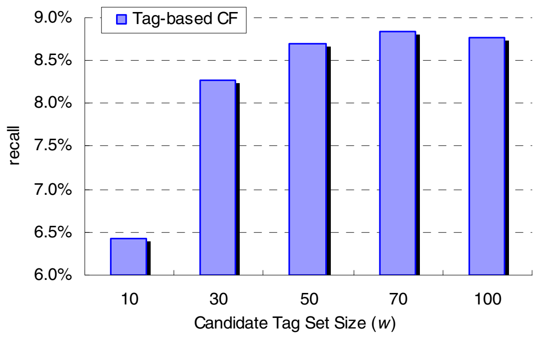


Fig. 4. The Variation of Recall of Tag-based CF

Fig. 4 shows that our algorithm tends to result in better quality as the size of *w* is increased. However, the quality of algorithm was rather worse when *w* was 100 (recall was 8.772%), whereas its recall was 8.839% which was the highest when *w* was 70. This result indicates that superfluous tags which do not represent user's preference can be included in *CTS*. That is, selecting too many numbers of tags can cause not only even bad impact on representing user's preference but also unnecessary cost for computation. For this reason, *w* should be selected within a reasonable level for experimentation, so we set it to 70, which obtained the best quality.

4.4 Comparisons of Performance

To experimentally compare the performance of our algorithm with those of user-based CF and item-based CF, we selectively varied the number of returned items *N* from 10 to 50 in an increment of 10. According to the results of the prior experiments in section 4.2 and 4.3, *k* and *w* were set to 50 and 70 respectively.

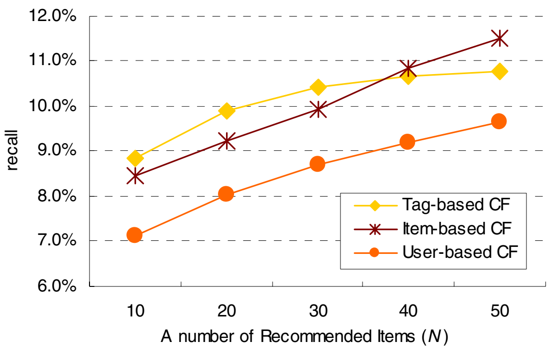


Fig. 5. Comparisons of Recall as the value of *N* increases

As we can see from these experiments, overall recall for all three algorithms was improved according to the increment of *N*. However, due to sparsity of the collected dataset from del.icio.us, all three methods did not perform well. These results were also affected by the number of items (17,390) being 10 times larger than the number of users (1,544). As shown in Fig. 5 nevertheless, tag-based CF outperformed user-based CF. In addition, the proposed algorithm performed better than an item-based one as *N* was increased from 10 to 30, even though after that the result became worse than item-based one. That is, when a relatively small number of items is recommended, the proposed method causes a more proper item to be at a higher rank in the returned item set, $TopN_u$, and so our algorithm can provide better items for a target user than the other algorithms.

5 Conclusions and Future Works

As a part of Web 2.0, collaborative tagging is getting popular as an important tool to classify dynamic content for searching and sharing. We analyzed the potential of collaborative tagging systems, including personalized and biased user preference analysis, and specific and dynamic classification of content for applying to recommendations. Also proposed is a novel recommendation algorithm based on CTS selected from collaborative tags of users using a CF scheme.

As described in our experimental results, the proposed algorithm obtained better recommendation quality compared to a traditional user-based CF algorithm. Moreover, we also observed that our method can provide more suitable items for user preference even though the number of recommended items is small.

However, the empirical result showed that “noise” tags which have bad influence on analyzing user preference can be included in *CTS*. Such tags, due to the characteristics of tag, personalized and content-criticizable (e.g., *bad*, *my work* and *to read*), should be treated effectively for more valuable and personalized analysises. In addition, there remain common issues that have been mentioned in keyword-based analysis; polysemy, synonymy and basic level variation [1]. Semantic tagging is one of the interesting issues that we plan to consider for addressing these problems in the future.

References

1. Golder, S.A., Huberman, B.A.: Usage Patterns of Collaborative Tagging Systems. Journal of Information Science 32(2), 198–208 (2006)
2. Marlow, C., Naaman, M., Boyd, D., Davis, M.: HT06, Tagging Paper, Taxonomy, Flickr, Academic Article, To Read. In: Procs. of the17th Conf. on Hypertext and Hypermedia, pp. 31–40 (2006)
3. Sarwar, B., Karypis, G., Konstan, J., Riedl, J.: Item-based Collaborative Filtering Recommendation Algorithms. In: Procs. of the 10th Int. Conf. on WWW, pp. 285–295 (2001)
4. Mathes, A.: Folksonomies-Coooperative Classification and Communication though Shared Metadata, http://www.adammathes.com/academic/computer-mediated-communication/folksonomies.html
5. Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., Riedl, J.: GroupLens: An Open Architecture for Collaborative Filtering of Netnews. In: Procs of ACM 1994 Conf. on Computer Supported Cooperative Work, pp. 175–186 (1994)
6. Herlocker, J.L, Konstan, J.A., Riedl, J.: Explaining Collaborative Filtering Recommendations. In: Procs of ACM 2000 Conf. on Computer Supported Cooperative Work, pp. 241–250 (2000)
7. Deshpande, M., Karypis, G.: Item-based Top-N Recommendation Algorithms. ACM Transactions on Information Systems (TOIS) 22(1), 143–177 (2004)
8. Han, J., Kamber, M.: Data Mining Concepts and Techniques, 2nd edn. Morgan Kaufmann Publishers, San Francisco (2006)
9. Sahami, M., Dumais, S., Heckerman, D., Horvitz, E.: A Bayesian Approach to Filtering Junk E-Mail. In: Learning for Text Categorization: Papers from the 1998 Workshop (1998)
10. Muller, M.J., Ehrlich, K., Farrell, S.: Social Tagging and Self-Tagging for Impression Management. In: CSCW 2006. Procs of the ACM Conf. on Computer Supported Cooperative Work (2006)
11. Sarwar, B., Karypis, G., Konstan, J., Riedl, J.: Analysis of Recommendation Algorithms for E-Commerce. In: Procs of the 2nd ACM Conf. on E-Commerce, pp. 158–167 (2000)
12. Voss, J.: Collaborative Thesaurus Tagging the Wikipedia Way (2006), http://arxiv.org/abs/cs/0-604036
13. Miller, B.N., Konstan, J.A., Riedl, J.: PocketLens: Toward a Personal Recommender System. ACM Transactions on Information Systems (TOIS) 22(3), 437–476 (2004)
14. Wal, T.V.: Explaining and Showing Broad and Narrow Folksonomies (2005), http://www.perso-nalinfocloud.com/2005/02/explaining_and_.html
15. Kelkar, S., Ajita, J., Doree, S.: An Activity-based Perspective of Collaborative Tagging. In: ICWSM 2007. Procs of the Int. Conf. on Weblogs and Social Media (2007)

Computational Imagination: Research Agenda

Rossitza Setchi[1], Nikolaos Lagos[1], and Danny Froud[2]

[1] School of Engineering, Cardiff University, The Parade, Cardiff CF24 3AA, UK
{Setchi, LagosN}@cf.ac.uk
[2] IAS Lab, University of West of England, Bristol Business Park, Bristol BS16 1QD, UK
{danny.froud}@uwe.ac.uk

Abstract. The goal of this paper is to promote a new research area named Computational Imagination, which is defined as the science of modeling human imagination by creating artificial agents with intelligence, emotions and imagination. The new discipline aims to (i) study the interplay between cognition, emotion and imagery, (ii) analyze the way perceptions, emotions, prior knowledge and context influence imagination, and (iii) design agents capable of forming concepts and images of something that is neither perceived nor sensed as real, and of dealing with the reality by using the creative power of their imagination. The paper outlines some of the most challenging and intriguing research questions, and directions for further research.

Keywords: Computational imagination, cognition, emotion, imagery, descriptionalism, pictorialism, concepts, semantics, AI.

1 Introduction

In one of his rare interviews, Einstein famously suggested: “Imagination…is more important than knowledge. Knowledge is limited. Imagination encircles the world.” Yet imagination has remained so far outside the reach of Artificial Intelligence (AI) research.

AI once captured the imagination of the world. However, many scientists have become disappointed at the lack of progress made since the 1980s [1]. Part of this discontent is the realization that the mind is not completely logical, and that the power of intelligence cannot be entirely identified with logical thinking. Even Marvin Minsky, often referred to as the godfather of AI, advocates the view that ‘there is no such thing as rational thought’ [2] and now suggests that the solution is to make machines more emotional [3].

This paper goes a step further. It goes beyond creating paranoid androids that ‘feel’ sadness or guilt after failing a task. The paper advocates the view that it is timely to establish an entirely new field of research in AI, a new research area called Computational Imagination, which has the ambitious goal of modeling human imagination. Computational Imagination will extend AI’s current role and lead to a greater understanding of humans by investigating a phenomenon that has, until now, remained outside the reach of AI research.

2 State-of-the-Art Review

2.1 Literature Review

As Perlovsky [4] points out, words like mind, thought, imagination, emotion, and concept present a challenge: people use these words in many ways colloquially, but in cognitive science and mathematics of intelligence they have not been uniquely defined and their meaning is a subject of active research and ongoing debates.

Indeed, the Oxford English dictionary [5] defines imagination as "the faculty or action of forming ideas or images in the mind". This definition encapsulates the existing controversy about the nature of imagination: some scholars consider it a distinct faculty of the mind that should be studied in isolation while others view imagination as a sub-process of cognition. In a comprehensive study of the cognitive, pre-cognitive and meta-cognitive aspects of imagination, O'Connor and Aardema [6] outline four different conceptions of imagination:

- Imagination as a distinct faculty (advocated by Aristotle, Saatre, Kant, and Descartes),
- Imagination as memory or a picture in the mind (Hobbes, Aquinas, Furlong, Gibson, and Hume)
- Imagination as originality, creativity, and transcendence (Bacon, Kant, and Fichte), and
- Imageless imagination (Ryle, White and Wittgenstein).

Moreover, there is ongoing argument, known as "the imagery debate", on whether imagination is based on forming ideas or images. The description theory (sometimes referred to as the propositional theory) claims that mental images [7] are structured language-based descriptions consisting of encodings and syntactical instructions [8, 9]. On the other hand, the picture theory associates imagination with the formation of mental images based on "quasi-pictures" or "functional pictures" rather than pictures in a literal sense [10]. The picture theory plausibly assumes that images fade over time. Both description and picture theories agree that imagination uses "deep representations" stored in long term memory.

The earliest attempts to develop computer models on imagery were made in the 1970s and used by the main advocates of the two theories, Kosslyn [10] and Pylyshyn [9], to vigorously promote the debate. The computer model of the picture theory [10] is based on extracting features from images, forming arrays and filling their elements with symbols representing the qualitative features of the image (such as color, 3D depth, and edges) at that point. Several computer models illustrating the description theory were also developed, most of them aimed at very circumscribed task domains such as solving imagery puzzles [11, 12].

In his seminal work [13] Mueller detached himself from the imagery debate and focused on how imagination is influenced by emotions. The link between emotions and human behavior is been further investigated by research in emotional engineering, affective computing, and emotional intelligence. In this context, emotional engineering explores the influence of colors, shapes, and sounds on attention and human behavior [14]. Affective computing mimics human motions and facial expressions and relates them to feelings [15] and social interactions [16], while emotional intelligence advocates that personality, human progress, and future performance can be predicted [17] using emotion related metrics. These findings indicate that imagination may also be linked to human behavior if the synergy between emotions and imagination is understood.

Mueller [13] has also stated that any artificial system capable of imagination should be able to monitor its environment and respond to unexpected situations. This is an active research topic in robotics where imagination is understood as any internally simulated interaction with the environment [18]. This approach is inspired by the perceptual activity theory of imagination, which is based on psychological and philosophical studies. Contrary to any previous research, this theory postulates that perceptual capacities are used to obtain only the necessary pieces of information needed to control behavior in a perceived environment [8, 19]. It also states that perceptual experience consists of the ongoing activity of schema-guided perceptual exploration of the environment. According to the perceptual activity theory, the synthesis of perception and imagination is the core process that creates new knowledge and understanding.

2.2 Research Questions

The review shows increasing interest towards studying human emotions, imagination and cognition, and the lack of formal computational models representing the synergy between them.

Over the past two decades there has been a shift from investigating thinking, i.e. logic and reasoning, to studying behavior [2, 20] partly due to the failure of current systems to exhibit resourcefulness in unexpected situations. The emerging research areas of computational creativity, emotional intelligence, and affective computing demonstrate that models of creativity and emotions can be used to interpret aspects of human behavior that cannot be explained by traditional methods.

An important question which is entirely overlooked by current research relates to the type of reasoning technique(s) needed to simulate imagination. Studies in areas such as computational creativity indicate that the use of connectionism [21], genetic algorithms, and latest theories about dynamical cognitive science and embodied cognition, is a promising direction in augmenting human creative abilities [22]. However, there is no known research that relates imagination to any of these techniques.

The establishment of a new research area dealing with these aspects in a systematic and formal manner is an ambitious step forward. It will develop new knowledge to deliver a better understanding of the role of imagination in developing possible futures using plausible but not necessarily logically sound reasoning.

3 Research Agenda

3.1 Research Goal

Computational Imagination is the science of modeling human imagination by creating artificial agents with intelligence, emotions and imagination.

The goal of Computational Imagination is to:

(i) study the interplay between cognition, emotion and imagery,

(ii) analyze the way perceptions, emotions, prior knowledge and context influence imagination, and

(iii) design agents capable of forming concepts and images of something that is neither perceived nor sensed as real, and of dealing with the reality by using the creative power of their imagination.

The research agenda includes:

1. Establishing the theoretical foundations of Computational Imagination by developing definitions, metrics, conceptual and formal models,
2. Establishing the computational foundations of Computational Imagination by developing reasoning models and algorithms for generating semantically linked scenes, and
3. Validating the developed theoretical and computational models using purpose built simulation testbeds.

3.2 Research Hypotheses

The following research hypotheses based on existing research in connectionism, pictorialism, descriptionalism and perceptual activity theory (see section 2) are the starting point in the development of the theoretical and computational models:

Imagination is a process of forming semantically linked mental images, each representing one or more concepts.

Imagination can be formally represented using both visual and linguistic means.

Imagination is influenced by perceptions, emotions, current context and prior knowledge.

The synergy of emotions, imagery and cognition is in the core of imagination.

3.3 Application Scenarios

There is a lot of evidence of the role imagination plays in business, art, science, engineering and social life. Imagination helps human beings to solve problems, express their creativity, develop hypotheses, advance theories, empathise and understand other people. In this context, the application scenarios which can be used to test and validate Computational Imagination are numerous and reside in a variety of business, engineering, educational and medical domains.

Examples of practical applications include creating and using mental imagery, the process of visualizing pictures, events and behaviour as mental simulation or mental scenario rehearsal to:

- Develop strategic thinking by creating and manipulating vivid projections of complex and ambiguous organizational environments,
- Improve scenario planning and facilitate organizational change by disciplined imagination, which involves designing, conducting and interpreting imaginary experiments,
- Learn new motor skills by supplementing motor training with mental training,
- Better predict and understand human behaviour in unknown or hostile environments,
- Prepare for disaster,
- Improve training by visualizing flight training manoeuvres,
- Assist older people by, for instance, helping them remember to comply with medical advice,
- Engage people in collaboration,
- Stimulate innovative and creative thinking,
- Help people learn from experience by constructing counterfactual ('if only' or 'what if') scenarios.

3.4 Major Challenges

One of the first challenges that research in this new discipline faces is the development of the conceptual model of Computational Imagination. Its aim is to express the synergy between perceptions, emotions, prior knowledge, context, and imaginative behavior. One way of representing the abstract concepts and relations associated with the mental images is by using image schemata [23] and conceptual metaphors. Similar to image schemata which have both pictorial and descriptive representations, imagination could be formally described using visual and linguistic means, which requires the availability of a lexicon based on natural language constructs and a collection of image schemata.

The synergy between emotions, imagery and cognition will have to be further studied by analyzing the way perceptions, emotions, prior knowledge, and context influence imagination. Prior knowledge (procedural and declarative) may stem from several sources: innate, previous interactions with environment, and culture. Context, on the other hand, includes current interactions with the environment, temporal characteristics and social settings.

Another challenge is associated with the development of appropriate reasoning techniques for generating sequences of semantically linked scenes, expressed again both visually and linguistically. A good starting point for this research is analyzing reasoning under inconsistency, associative, and analogical reasoning, and possibly identifying the need for a new reasoning technique.

Furthermore, algorithms for constructing and de-constructing semantically linked sequences of scenes (expressed both visually and linguistically) present a further challenge due to the combinatorial complexity involved.

Computational Imagination will challenge traditional AI as a discipline and will change its nature by introducing 'real-world reasoning' which is highly situated and contextualized. It will investigate alternative ways of reasoning such as associative reasoning (extremely unpredictable and entirely personalized), analogical reasoning (combined with ontologies), and reasoning under inconsistency (paraconsistent logics with contradictions that are typical for irrational scenarios).

The further property of imagination in comparative or constrained emotive situations leads to empathy, a development leading to prediction and understanding of human behavior and needs. This predictive ability could then be used to determine appropriate action, behavior or communication depending on the circumstance. It will also become a useful tool for determining alternative and novel solutions, and a foundation for developing ‘common sense’ in AI for robotics.

4 Conclusions

Over the past two decades there has been a shift from investigating thinking to studying behavior. There is an increasing interest towards studying human emotions, imagination and cognition, and yet no formal computational models exist which show the synergy between them. Emerging research indicates that models of creativity and emotions can be used to interpret aspects of human behavior that cannot be explained by traditional methods. The establishment of a new research area dealing with these aspects in a systematic and formal manner is an ambitious step forward. Computational Imagination possesses some of the most challenging and intriguing questions, which once answered, will make imagination reality.

Acknowledgments. The authors wish to express their gratitude to the European Research Council for encouraging visionary and groundbreaking research in Europe through their IDEAS research initiative. The authors are grateful to Dr S Westernman, Prof J Miles, Mr P Prickett, Prof D T Pham and Prof S Dimov for their suggestions and support.

References

1. Berners-Lee, T., Hall, W., Hendler, J.A., O’Hara, K., Shadbolt, N., Weitzner, D.J.: A Framework for Web Science. Foundations and Trends in Web Science 1, 1–130 (2006)
2. Gefter, A.: Interview: Once More With Feelings. The New Scientist. 193, 48–49 (2007)
3. Minky, M.: The Emotion Machine: Commonsense Thinking. In: Artificial Intelligence, and the Future of the Human Mind, MIT Press, Cambridge (2006)
4. Perlovsky, L.I.: Towards Physics of the Mind: Concepts, Emotions, Consciousness, and Symbols. Physics of Life Reviews 3, 23–55 (1996)
5. Soanes, C., Hawker, S.: Compact Oxford English Dictionary of Current English, 3rd edn. Oxford, UK (2005)
6. O’Connor, K.P., Aardema, F.: The Imagination: Cognitive, Pre-Cognitive, and Meta-Cognitive Aspects. Consciousness and Cognition 14, 233–256 (2005)
7. Johnson-Laird, P.N.: Mental Models: Towards a Cognitive Science of Language, Inference, and Consciousness. Harvard University Press, Cambridge (1983)
8. Ellis, R.D.: Questioning Consciousness: The Interplay of Imagery, Cognition and Emotion in the Human Brain. Advances in Consciousness Research Series 2 (1995)
9. Pylyshyn, Z.W.: Computation and Cognition. MIT Press, Cambridge (1984)

10. Kosslyn, S.M., Pomerantz, J.R.: Imagery, Propositions, and the Form of Internal Representations. Cognitive Psychology 9, 52–76 (1977)
11. Baylor, G.W.: A Treatise on the Mind's Eye. Unpublished doctoral dissertation. Carnegie-Mellon University, Pittsburgh. University Microfilms No. 72-12, 699 (1972)
12. Moran, T.P.: The Symbolic Imagery Hypothesis: A Production System Model. Unpublished Doctoral Dissertation, Carnegie-Mellon University, Pittsburgh. University Microfilms No. 74-14, 657 (1973)
13. Mueller, E.T.: Daydreaming in Humans and Machines: A Computer Model of the Stream of Thought. Ablex Publishing, Greenwich (1990)
14. Korsten, N.J.H, Fragopanagos, N., Harley, M., Taylor, N., Taylor, J.G.: Attention as a Controller. Neural Networks 19, 408–421 (2006)
15. Hone, K.: Empathic Agents to Reduce User Frustration: The Effects of Varying Agent Characteristics. Interacting with Computers 18, 227–245 (2006)
16. Boehner, K., DePaula, R., Dourish, P., Sengers, P.: How Emotion is Made and Measured. International Journal of Human-Computer Studies 65, 275–291 (2007)
17. Mayer, J.D.: A New Field Guide to Emotional Intelligence. In: Emotional Intelligence in Everyday Life, 2nd edn. Psychology Press/Taylor & Francis (2006)
18. Hesslow, G.: Conscious Thought as Simulation of Behavior and Perception. Trends in Cognitive Science 6, 242–247 (2002)
19. Thomas, N.: Are Theories of Imagery Theories of Imagination? An Active Perception Approach to Conscious Mental Content. Cognitive Science. 23, 207–245 (1999)
20. Pfeifer, R., Bongard, J.C.: How the Body Shapes the Way We Think: A New View of Intelligence. MIT Press, Cambridge (2006)
21. Rumelhart, D.E., McClelland, J.L.: Parallel Distributed Processing: Explorations in the Microstructure of Cognition. In: Foundations, vol. 1, MIT Press, Cambridge (1986)
22. Wlodzislaw, D.: Creativity and the Brain. In: A Handbook of Creativity for Teachers, World Scientific Publishing, Singapore (2007)
23. Johnson, M.: The body in the Mind: The Bodily Basis of Meaning, Imagination, and Reason. University of Chicago Press, Chicago (1987)

An Upper Ontology for Event Classifications and Relations

Ken Kaneiwa[1], Michiaki Iwazume[1], and Ken Fukuda[2]

[1] National Institute of Information and Communications Technology
3-5 Hikaridai, Seika, Soraku, Kyoto 619-0289, Japan
{kaneiwa,iwazume}@nict.go.jp

[2] National Institute of Advanced Industrial Science and Technology
2-42 Aomi, Koutou-ku, Tokyo 135-0064, Japan
fukuda-cbrc@aist.go.jp

Abstract. For knowledge representation and reasoning, there is a need to consider the nature of events because event data describe various features and behaviors of the occurrences of actions and changes in the real world. In this paper, we propose to establish an upper event-ontology in order-sorted logic as an infrastructure for event knowledge bases. Our event ontology contains a classification of event entities (e.g., natural events and artificial events) and event relationships (e.g., causal relations and next-event relations). These ontological characterizations are needed for a theoretical basis of applications such as implementation of event databases, detection of event relationships, and annotation of event data.

1 Introduction

Events are entities classified in a formal ontology that are difficult to treat in knowledge base systems. To describe the occurrences of actions and changes in the real world, knowledge base designers have to consider that objects and properties are static, but events are dynamic.

In the fields of logic, linguistics, ontology, artificial intelligence, and deductive databases, the nature of events has been investigated as follows. Allen et al. [1] explained that events were methods used to classify useful and relevant patterns of change rather than entities in the real world. Sowa [12] categorized events as changes that occur in the discrete steps of a process. Active object-oriented databases [3,2] have event specification facilities to describe complex events. Galton and Augusto [2] attempted to combine the two kinds of event definitions of knowledge representation and databases. As an object-oriented approach, Worboys and Hornsby [13] proposed the foundations of modeling objects and events for dynamic geospatial domains.

Upper ontologies have defined the nature of events for information systems. In fact, several standard upper ontologies are distributed on the Web. Guarino's group built the DOLCE (Descriptive Ontology for Linguistic and Cognitive Engineering)[8], in which events are a subclass of perdurant occurrences that are disjoint to the entities of endurant, quality, and abstract. The SUMO

(Suggested Upper Merged Ontology)[9], designed by the IEEE Standard Upper Ontology Working Group consists of a set of concepts, relations, and axioms where abstract and physical entities are divided, in which the physical entities are classified into objects and processes. The OpenCyc [10] contains an upper ontology, in which event entities are defined as temporal and intangible.

In this paper, we propose to extend an event ontology, which is designed in the structural representation of order-sorted logic [5], to an infrastructure for event knowledge bases. First, we classify the types of events (e.g., natural events and artificial events). Our approach attempts to construct event classifications based on our two ontological views: component structures and semantic functions of events. The component structures lead to the knowledge representation of events (e.g., the argument structures of predicate formulas) and the semantic functions imply the logical and ontological semantics of events for reasoning. Second, we introduce event relations (e.g., causal relations and next-event relations) capturing the differences between instances and classes of events. In order that these relations represent a sequence of events and constraints in event knowledge bases, we modify Smith's relationship ontology for bioinformatics [11].

2 Ontology Description in Order-Sorted Logic

First, we briefly explain the basic usage of order-sorted logic for formalizing ontologies and concept definitions. Ontologies are usually represented by concept hierarchies. We build an event ontology as a sort-hierarchy in order-sorted logic where many sort symbols denote concepts as sets of individuals and their ordered relation corresponds to a concept hierarchy. The following form represents an ontology consisting of four sorts where a sort symbol s_1 has two subsort symbols s_2 and s_3, and the sort symbol s_3 has a subsort symbol s_4.

```
Sort $s_1$
    Subsort $s_2$
    Subsort $s_3$
        Subsubsort $s_4$
```

Such a sort-hierarchy is interpreted according to the semantics of order-sorted logic in such a way that every instance of sort s_2 is also an instance of its supersort s_1. Let S be the set of sort symbols. In the syntax of order-sorted logic, a subsort relation $\sqsubseteq$ $(\subseteq S \times S)$ of the above ontology is declared as follows.

$$s_4 \sqsubseteq s_3,\ s_3 \sqsubseteq s_1,\ s_2 \sqsubseteq s_1$$

In the usual manner, each sort symbol expresses a set of physical objects (e.g., humans) or a set of abstract objects (e.g., rational numbers) where the same type of object belongs to a sort. In an extension to the sort representation, we regard event instances as the occurrences of an event that belong to an event sort. This notion is based on the fact that if the same event happens many times, their respective occurrences can be recognized as instances of the same event sort.

Every event sort E has to be distinguished from every object sort. Each physical or abstract object of a sort has an identifier, but each instance of an event sort has a different time and location. Then, two special sorts $Time$ and $Location$ are introduced as the set of times and the set of locations, respectively. In addition, every instance of event sorts can play the role of an n-ary predicate representing a relationship among components of the event (e.g., actor and object). The following sort declaration of an n-ary predicate fixes the component structure of event sort E in a sorted signature.

$$E\colon \langle s_1, \ldots, s_n \rangle$$

where sorts $s_1, \ldots, s_n$ are disjoint to event sort E. This declaration implies that every instance of event sort E is an n-ary predicate whose argument structure is of sorts $s_1, \ldots, s_n$. In the next section, using the sort representation, we define classifications and relations of events in an event ontology.

3 Event Classifications in an Ontology

Components of Events: We show the following ontology that classifies natural events, artificial events, dynamic states, and static states together with their respective component structures.

- Event
  - NaturalEvent
    - $Occurence1\colon \langle Time, Location \rangle$
    - $Occurence2\colon \langle Object, Time, Location \rangle$
  - ArtificialEvent
    - $Action1\colon \langle Agent, Object, Time, Location \rangle$
    - $Action2\colon \langle Agent, Time, Location \rangle$
    - $Action3\colon \langle AgentGroup, Time, Location \rangle$
  - DynamicState
    - $ObjectChange\colon \langle Object, Time, Location \rangle$
    - $EnvironmentChange\colon \langle Time, Location \rangle$
  - StaticState
    - $ObjectState\colon \langle Object, Time, Location \rangle$
    - $EnvironmentState\colon \langle Time, Location \rangle$

In the event ontology, natural events and artificial events are defined as having different component structures. If an event happens naturally and there is no actor as a component of the event, the event is called a natural event. A natural event does not contain any actor, but there may be a main component of the event. For example, a volcanic eruption is caused by a volcano and a flood is caused by a sea or river. In contrast, earthquakes and typhoons are described with locations, without any component. For example, an earthquake happens in a location and a typhoon is moving to a location. In our opinion, this difference in component structures helps in representing events in predicate formulas. More precisely, natural events without any component are represented by a binary predicate formula of time and location. Natural events with a main component are represented by a trinary predicate formula of an object, time, and location.

If there are some actors as components that generate an event, then it is called an artificial event. The three types of artificial events are defined with respect to the actors and other components of the events as follows:

(1) Actions from agents to objects or agents (e.g., murder and environmental disruption)
(2) Intransitive actions (e.g., breathing and moving)
(3) Actions with many agents (e.g., conference and discussion)

There are relationships among agents and objects in the three types of artificial events. The first type of artificial event is an action that is generated by an agent against objects (or agents) in the world. This event is described by a quaternary predicate formula of an actor, an object (or agent), time, and location. The second type of artificial event is an action describing the motion of an agent. This event consists of an actor and does not contain other objects. Hence, the event is described by a trinary predicate formula where the arguments are an actor, time, and location. Unlike the first two event types, the third type of artificial event is the whole entity that is generated and in which many agents participate. This event type does not focus on the action of each agent, although the internal action of the event may contain the first two event types.

In addition to these events, we deal with states that are divided into states of objects and states of environments. As a result of this, the object state is expressed by a trinary predicate formula because the state has the three components of an object, time, and location. The environment state is described by a binary predicate formula that consists of the two components of time and location. We regard states of objects or environments as events because states are events representing the changes of objects or environments in dynamic time and location. In philosophical research, there is a distinction between events and states, but states are categorized as events in the above ontology. In practice, the unified category of events and states is useful for us to formalize their relationships, such as causal relations discussed in Section 4. Related to this, Hobbs et al.[4] defined event concepts including state concepts for the objective of tagging in natural languages.

Definition 1 (Activity of States). *A state is dynamic if the state implies the activity and dynamic change of an object or environment in time. A state is static if the state implies the static property in time.*

For instance, “rolling continues”, “rising/dropping”, “slightly active”, and “become strong,” which represent the changes of states, are dynamic states, but “hot”, “cold”, and “fine” are static states. The dynamic state “become higher” indicates that the temperature of the static state “low temperature” will become higher because of a change of the static state.

Semantic Functions of Events: We next characterize semantic functions of events in addition to the event classification given by event components. Semantic functions formally and semantically indicate that each event implies the functional change and behavior of objects in the real world. Let us introduce six semantic functions of events in the event ontology as follows.

- EventSemanticFunctions
  - StateChange
  - TemporalExistenceChange
  - SpatialExistenceChange
  - CardinarityChange
  - Comparison
  - ObjectIdentificationChange

These semantic functions are strongly related to the meaning of verb expressions in natural languages because each verb is a word to express an action or state.

Related to this end, Kaneiwa and Mizoguchi[6,7] have proposed a property classification in an upper ontology where properties are divided by their rigidity in modal order-sorted logic. Using their formalization of the ontology, the semantic functions of events can be defined by modalities, cardinalities, quantifications, and logical connectives. In the semantic functions, each event affects an object or an environment and therefore changes their property or state in the next time. Therefore, we define the semantic function of an event in logic by the truth value changes of properties and states over dynamic times and locations. To embed such complex events in semantics, we have to introduce equations and generalized quantifiers as well as modal and temporal operators.

Definition 2 (State Change). *An event is a state-change-event if the occurrence yields the change of states from now to the next time as follows:*

$$F_1 \rightarrow \bigcirc F_2$$

The modal operator $\bigcirc F_2$ implies that F_2 is true at the next time. Therefore, the semantic function of event e is defined by the statement $F_1 \rightarrow \bigcirc F_2$, which implies that if a state or property F_1 is true, then the event e yields another state or property F_2 at the next time.

Definition 3 (Existential Change over Time). *An event is a temporal-existence-change-event if its occurrence changes the existence of an object according to a change in time as follows:*

$$\Box_P(\neg E(x)) \wedge E(x)$$

The temporal operator $\Box_P F$ implies that the formula F was always true in the past. The semantic function of event e contains the existential formula $\Box_P(\neg E(x)) \wedge E(x)$ implying that an object x did not exist in the past but it exists now.

Definition 4 (Existential Change over Space). *An event is a spatial-existence-change-event if its occurrence changes the existence of an object depending on movement through space as follows:*

$$\neg E(x) \wedge \blacklozenge E(x)$$

The spatial operator $\blacklozenge F$ implies that the formula F is true in a place accessible from here. Hence, the semantic function of $\neg E(x) \wedge \blacklozenge E(x)$ indicates that an object x does not exist here, but it exists in a place accessible from here.

Definition 5 (Cardinality Change). *An event is a cardinality-change-event if it changes the cardinality of objects as follows:*

$$\exists_i x F(x) \rightarrow \bigcirc \exists_{>i} x F(x)$$

Let n be a natural number. The generalized quantifier $\exists_n x F(x)$ (called counting quantifier) expresses the existence of n objects x such that the formula $F(x)$ is true. In the cardinality change, we have to introduce a variable i of natural numbers for the generalized quantifier $\exists_i$ that is more expressive than the the generalized quantifier $\exists_n$. Hence, the semantic function of $\exists_i x F(x) \rightarrow \bigcirc \exists_{>i} x F(x)$ indicates that if there exist i objects x such that $F(x)$ is true, then there exist more than i objects x such that $F(x)$ is true the next time. This function means that the number of objects increases by the next time because of the event occurrence.

Definition 6 (Comparison). *An event is a comparison event if the attribute value of an object is found to change when comparing that value with the attribute value at the next time as follows:*

$$\exists y(\mathit{Value}(x) = y \rightarrow \bigcirc(\mathit{Value}(x) > y))$$

In the semantic function, $\mathit{Value}(x)$ denotes the attribute value of an object x and the inequality symbol $>$ is used to compare the value of the attribute now to that of the next time. To compare the value of x now to that of the next time, a fixed value y and the changed value $\mathit{Value}(x)$ appear in the formula $\exists y(\mathit{Value}(x) = y \rightarrow \bigcirc(\mathit{Value}(x) > y))$.

Definition 7 (Object Identification Change). *An event is an object-identification-change-event if the essential property of an object is changed and therefore the object cannot be recognized as the former object at the next time as follows:*

$$\exists y(x \equiv y \rightarrow \bigcirc(x \not\equiv y))$$

The object identification is lost at the next time after the occurrence of an object-identification-change-event. Hence, the function of $\exists y(x \equiv y \rightarrow \bigcirc(x \not\equiv y))$ implies that if there exists an object y such that y is identical to object x, then the event makes them different at the next time. This means that an object x is changed into another object y.

There may be several semantic functions for an event. In other words, many logical operations can be used to functionally define an event. Using the above definitions, a correspondence list of event predicates and quantified modal formulas denoting events and their semantic functions is shown in Table 1.

Table 1. Semantic Functions of Events

Event predicates	Quantified modal formulas
$Cure(x,y)$	$\neg Healthy(y) \wedge (Act(x,y) \vee Affect(x,y)) \rightarrow \bigcirc Healthy(y)$
$Drink(x,y)$	$(InMouth(y) \wedge Swallow(x,y) \wedge \neg Bite(x,y)) \rightarrow \bigcirc InBody(y)$
$Stop(x,y)$	$Active(y) \wedge (Action(x,y) \vee Affect(x,y)) \rightarrow \bigcirc \neg Active(y)$
$Die(x)$	$\Diamond_P(\Box_P \neg E(x) \wedge E(x)) \wedge (\neg E(x) \mathcal{S} E(x)) \wedge \Box_F(\neg E(x))$
$Print(x,y)$	$\exists v(Object(v) \wedge Act(x,v) \rightarrow \bigcirc \exists z(Above(z,v) \wedge Press(Mark(y),z)))$
$BeBorn(x)$	$\Box_P(\neg E(x)) \wedge E(x)$
$Go(x)$	$Act(x) \rightarrow \bigcirc(\neg E(x) \wedge \blacklozenge E(x))$
$Separate(x,y)$	$Adjoint(x,y) \vee Overlap(x,y) \rightarrow \bigcirc(\neg Adjoint(x,y) \wedge Overlap(x,y))$
$Understand(x,y)$	$\exists v((v = Fact(x) \vee v = Content(x) \vee v = Meaning(x)) \wedge Get(x,v))$
$Increase(x)$	$\exists i(Nat(i) \wedge \exists_i x\ Countable(x) \rightarrow \bigcirc \exists_{>i} x\ Countable(x))$ $\exists y(Rel(y) \wedge Quantity(x) = y \rightarrow \bigcirc(Quantity(x) > y))$
$Decrease(x)$	$\exists i(Nat(i) \wedge (i > 0) \wedge \exists_i x\ Countable(x) \rightarrow \bigcirc \exists_{i-1} x\ Countable(x))$ $\exists y(Rel(y) \wedge Quantity(x) = y \rightarrow \bigcirc(Quantity(x) < y))$
$Raise(x)$	$\exists y(Location(x) = y \rightarrow \bigcirc(\neg E(x) \wedge \blacklozenge(E(x) \wedge Location(x) > y))$ $\exists y(Value(x) = y \rightarrow \bigcirc(Value(x) > y))$
$High(x,y)$	$\exists r(Rel(r) \wedge Value(x) = r \wedge (Value(y) < r))$
$Change(x)$	$\exists y(x \equiv y \rightarrow \bigcirc(x \not\equiv y))$
$Make(x,y)$	$\exists z(Act(x,z) \rightarrow \bigcirc(BeBorn(y) \wedge y \not\equiv z \wedge Valuable(y)))$ $Act(x) \rightarrow \bigcirc BeBorn(y)$

4 Event Relations: Cause-Effect and Others

We modify Smith's ontology [11] to define relationships between events in knowledge bases, which are important for describing a sequence of many events, e.g., a sequence of causes and effects. For such event relations, two events may be connected by causal, temporal, and spatial constraints, and the distinction between event instances and event classes should be carefully considered.

Before defining the event relations, we generally construct a binary relationship ontology for objects and events as follows:

BinaryRelation
- $ObjectRelation : \langle Object, Object \rangle$
- $EventRelation : \langle Event, Event \rangle$
- $CausalRelation : \langle Object \sqcup Event, Object \sqcup Event \rangle$

A causal relation is a complex relation over various entities such as objects and events (including states). For example, if John sets fire to a house, the cause of the fire is John. Then, we can write the causal relation $John \rightarrow_{cause} fire$. As a part of the above binary relations, we give an event-relationship ontology that contains various event relations as follows:

EventRelation
- EventInstanceRelation
  - EventTemporalRelation

- EventSpatialRelation
- NextEventRelation
- PartOfRelation
- EventInstanceCausalRelation
    - DisjointCausalRelation
    - ContinuousCausalRelation
    - OverlappingCausalRelation
    - PartialCausalRelation

EventClassRelation
- DisjointRelation
- SubclassRelation
- PartOfRelation
- EventClassCausalRelation

To formalize event relations in the above ontology, event instances and event classes have to be distinguished for identifying elements of the relations. An event instance is the actual occurrence of an event (e.g., the occurrence of an earthquake), but an event class is a set of event instances corresponding to a type or feature of events (e.g., the set of event instances of earthquakes). We should note that each event instance involves time and location information in which the event actually occurs, but each event class does not need to have the information. Following the distinction between instances and classes of events, their event relations are respectively defined as follows:

Definition 8 (Event-Instance Relations). *Let e_1, e_2 be two event instances. Then, a binary relation $r(e_1, e_2)$ between e_1 and e_2 is called an event-instance relation, which is defined by the following:*

(1) If an event instance e_1 causes an event instance e_2, then the causal relation $e_1 \rightarrow_{cause} e_2$ holds.
(2) If an event instance e_2 occurs after event instance e_1, then the next-event relation $e_1 \rightarrow_{next} e_2$ holds.
(3) If an event instance e_1 temporally includes an event instance e_2, and e_1 occurs in a spatial part of e_2, then the event-part-of relation $e_1 <_{po} e_2$ holds.

The temporal relations uniquely determined by event instances further divide the event-instance causal relations into disjoint, continuous, overlapping, and partial causal relations (as shown in Fig. 1).

Definition 9 (Event-Class Relations). *Let E_1, E_2 be two event classes. Then, a binary relation $R(E_1, E_2)$ between E_1 and E_2 is called an event-class relation, which is defined by the following:*

(1) If each event instances of E_1 and E_2 cannot occur simultaneously, then the event disjoint relation $E_1 || E_2$ holds.
(2) If every instance of E_1 belongs to E_2, then the event-subclass relation $E_1 \sqsubseteq E_2$ holds.
(3) If for every event instance e of E_1, there is an event instance e' of E_2 such that $e' <_{po} e$, then the event-class part-of relation $E_1 <_{po} E_2$ holds.

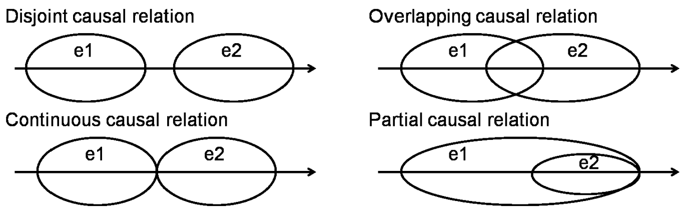


Fig. 1. Causal Relations over Time

(4) If for every event instance e of E_1, there is an event instance e' of E_2 such that $e \rightarrow_{cause} e'$, then the event-class causal relation $E_1 \rightarrow_{cause} E_2$ holds.

For example, the event disjoint relation $heavy\_rain || fine\_weather$ holds because they cannot happen simultaneously. Event classes are structured by an event-subclass relation. The event class *earthquake* is used to describe an event concept as an occurrence set of earthquakes. However, a unique event instance (e.g., Sumatra earthquake as an event instance of *earthquake*) can be regarded as the only instance of the event class, which is a singleton. Then, the event-subclass relation $\sqsubseteq$ is declared as follows:

$$\{Sumatra\_earthquake\} \sqsubseteq earthquake \sqsubseteq natural\_disaster \sqsubseteq disaster$$

Each event-instance causal relation is transitive. That is, if $e_1 \rightarrow_{cause} e_2$ and $e_2 \rightarrow_{cause} e_3$, then $e_1 \rightarrow_{cause} e_3$. However, the event-class causal relation is not transitive as follows:

Proposition 1 (Intransitivity of Causal Relations). *Let E_1, E_2, E_3 be three event classes. Then, there are event-class causal relations $E_1 \rightarrow_{cause} E_2$ and $E_2 \rightarrow_{cause} E_3$ such that $E_1 \rightarrow_{cause} E_3$ does not hold.*

We discuss a part-of relation between events. Each event contains two types of partial events. The first partial event is an essential event that always occurs in the whole event. If we consider an earthquake event, a P-wave and S-wave are essential events of every earthquake. The second partial event is a non-essential event in the whole event. For example, a rescue operation is a non-essential event of an earthquake. The rescue operation is an artificial event, which is an action performed by rescuers on victims. This partial event consists of rescuers, victims, suffering things, and stricken areas that are people, objects, and locations. From the view of instances and classes of events, the following property is derived. The essential event parts are often defined by a part-of relation between event classes. If an event instance has a non-essential event part, then the event instance is defined by a part-of relation between event instances.

5 Conclusion

The result of this research was the characterization of event entities in an upper ontology and logical formalization. For natural/artificial events and dynamic/

static states in the ontology, the event component structures were classified so that they correspond to the argument structures of predicate formulas. As an alternative view, the semantic functions of events were analyzed in expressive logical formulas that enabled us to infer logical conclusions from event occurrences. Furthermore, we explained event relations where four types of causal relations and other relations were introduced by distinguishing event instances and classes. These relations would be necessary for relational descriptions of events in knowledge bases. In future work, we plan to develop a reasoning system for the event relations based on our proposed event ontology.

Acknowledgment

This work was supported by KAKENHI (Grant-in-Aid for Scientific Research) on Priority Areas (No.18016032, "Systems Genomics") from the Ministry of Education, Culture, Sports, Science and Technology of Japan.

References

1. Allen, J.F., Ferguson, G.: Actions and events in interval temporal logic. Journal of Logic and Computation 4(5), 531–579 (1994)
2. Galton, A., Augusto, J.C.: Two approaches to event definition. In: Hameurlain, A., Cicchetti, R., Traunmüller, R. (eds.) DEXA 2002. LNCS, vol. 2453, pp. 547–556. Springer, Heidelberg (2002)
3. Gatziu, S., Dittrich, K.R.: Events in an Active Object-Oriented Database System. In: Proc. of 1st Inter. Work. on Rules in Database Systems (1994)
4. Hobbs, J., Pustejovsky, J.: Annotating and reasoning about time and events. In: The Language of Time, Oxford University Press, Oxford (2005)
5. Kaneiwa, K.: Order-sorted logic programming with predicate hierarchy. Artificial Intelligence 158(2), 155–188 (2004)
6. Kaneiwa, K., Mizoguchi, R.: Ontological knowledge base reasoning with sort-hierarchy and rigidity. In: KR2004. Proc. of the 9th International Conference on the Principles of Knowledge Representation and Reasoning, pp. 278–288 (2004)
7. Kaneiwa, K., Mizoguchi, R.: An order-sorted quantified modal logic for meta-ontology. In: Beckert, B. (ed.) TABLEAUX 2005. LNCS (LNAI), vol. 3702, pp. 169–184. Springer, Heidelberg (2005)
8. Masolo, C., Borgo, S., Gangemi, A., Guarino, N., Oltramari, A., Schneider, L.: Wonderweb deliverable d17. the wonderweb library of foundational ontologies and the dolce ontology
9. Niles, I., Pease, A.: Towards a standard upper ontology. In: FOIS 2001. Proc. of the 2nd Inter. Conf. on Formal Ontology in Information Systems (2001)
10. OpenCyc. http://www.opencyc.org
11. Smith, B., Ceusters, W., Klagges, B., Kohler, J., Kumar, A., Lomax, J., Mungall, C., Neuhaus, F., Rector, A., Rosse, C.: Relations in biomedical ontologies. Genome Biol. 6(5), 46 (2005)
12. Sowa, J.F.: Knowledge Representation. Brooks/Cole (2000)
13. Worboys, M.F., Hornsby, K.: From objects to events: Gem, the geospatial event model. In: Egenhofer, M.J., Freksa, C., Miller, H.J. (eds.) GIScience 2004. LNCS, vol. 3234, pp. 327–344. Springer, Heidelberg (2004)

A Within-Frame Ontological Extension on FrameNet: Application in Predicate Chain Analysis and Question Answering

Bahadorreza Ofoghi, John Yearwood, and Ranadhir Ghosh

[1] Centre for Informatics and Applied Optimization
School of Information Technology and Mathematical Sciences, University of Ballarat
PO Box 663, Ballarat VIC 3353, Australia
bofoghi@students.ballarat.edu.au,
{j.yearwood, l.ma}@ballarat.edu.au

Abstract. An ontological extension on the frames in FrameNet is presented in this paper. The general conceptual relations between frame elements, in conjunction with existing characteristics of this lexical resource, suggest more sophisticated semantic analysis of lexical chains (e.g. predicate chains) exploited in many text understanding applications. In particular, we have investigated its benefit for meaning-aware question answering when combined with an inference strategy. The proposed knowledge representation mechanism on the frame elements of FrameNet has been shown to have an impact on answering natural language questions on the basis of our case analysis.

Keywords: FrameNet, Ontological Relation, Question Answering.

1 Introduction

In many cases, most knowledge underlying a lexical resource can be extracted when the knowledge is represented in a network of relations between the constituents of a resource. In such a network, both the nodes (constituents) and the edges (relations) are knowledge-bearing. Such relations formulate the interactions happening between constituents in a more sophisticated fashion.

Understanding these interactions can help in text understanding applications like *predicate chain analysis*. By predicate chains we refer to a specific type of lexical chains where relations between the lexical units (lexemes) are realized by the concept of the predicates. With such a definition, predicate chains differ from the three main types of lexical chains with WordNet-based relations (i.e. extra-strong, strong, and regular). We have identified a mapping strategy which transforms predicative relations to some more general ontological relations that can be useful in text understanding applications.

Question answering (QA) is one of the areas where predicate chain analysis can help. On some occasions, the answer entity is hidden behind a chain of predicative relations or it can be justified only after pursuing such relations in the answer-bearing texts. For instance, the sentence "*they first met on the bridge which was later named as Johnston and was one of the adventures in the area of Mount Clear...*" for the question "*who first met in Mount Clear?*" contains the actual answer that can be reached when following the relations between the entities "bridge", "Johnston", and "Mount Clear". Accessing such an answer entity requires that an automated system be capable of semantically resolving the constituents and connecting them all together in an inference mechanism to identify the right answer.

We have been investigating *FrameNet* [1] as a lexical resource to achieve better answer extraction. To address some difficulties in answering questions, an ontological extension on FrameNet has been formalized and evaluated. Such an extension consists of ontological relations between the elements within the frames which can reveal more knowledge about the participant roles in a scenario. This ontological extension, in conjunction with an inference engine, elevates the ability of a QA system to more semantically identify correct answers to more questions.

2 FrameNet

FrameNet is a lexical resource for English [1] that relies on *Frame Semantics* [2] [3]. Frame semantics, basically developed from Charles Fillmore's *Case Structure Grammar* [4] [5], emphasizes the continuities between language and human experience. FrameNet, as a result of these efforts, is a framework to encapsulate the semantic information gained via frame semantics [6].

The main entity in FrameNet is a *frame* as a kind of normalization over the concepts semantically related. The real-world knowledge about real scenarios and their related properties are encoded in the frames [3]. Each frame contains some *frame elements* (FEs) as representatives of different semantic and syntactic roles regarding the target word of the frame. The semantic roles are usually common properties among all of the lexical units that are inherited from a frame.

A limited set of frame-to-frame relations has been defined in FrameNet which connects frames to constitute a network of concepts and their pictures [7]. Such relations have been used for event structure identification and inference on complicated story scrutinizing in applications like QA [8].

In addition to the associations between the FEs across the frames necessitated by the frame-to-frame relations, there are other relations between the FEs within a frame. The two major existing such relations are *Requires* and *Excludes* relations which emphasize the philosophy of existence of the participant roles in the scenarios covered by the definitions of the frames.

3 Ontological Extension on FEs

Those existing FE-to-FE relations in FrameNet realized as the consequences of frame-to-frame relations are meaningful since such meta-relationships (over frames) exist. For instance, the FE *Perceiver-agentive* in the frame *Perception-active* is a child of the *Perceiver* FE in the parent frame *Perception* as the scenario of *Perception-active* is in fact a more specific type of *Perception*. These FE-to-FE relations across the frames, initiated by the frame-to-frame relations, have been used in QA in [9] the main advantage of which is that the answering system goes beyond the limitations of the specific frame events via the relations using a reasoning module. However, it still cannot be applied where there is no initial frame-to-frame connection.

Those very few relations between the FEs within a frame (requires and excludes) are limited to the boundaries of the scenarios covered by the frames with no (or very narrow) possibility of being used across the (non-related) frames. They are not related to any status of the events either. For example, the *Figure* FE in the frame *Similarity* requires the *Ground* FE with no reference to any event status. And the *Employer* FE in the frame *Hiring* requires *Employee* as any sentence with explicit employer must have an explicit employee to be grammatical and meaningful.

To overcome such shortcomings an ontological extension of frames with respect to their FEs has been proposed, implemented, and analyzed in our work. The main characteristics of this ontological extension are:

- The relations hold between the FEs in a frame and can be activated across the frames (refer to section 4) using the consequential inference engine even where there is no frame-to-frame relation between the FE-bearing frames,
- At the main time slices of a complicated scenario, certain ontological relations between the FEs in a frame are valid,
- The ontological relations between the FEs in a frame denote conceptual relationship between participating roles (not just the existence of the FEs).

3.1 Ontological Relations

There are a few aspects which should be noted with regard to the relation set. First, the relation set has to be a finite set of meaningful relations understandable for similar natural language processing communities and knowledge representation peers. Second, the relations need to be well-defined and machine and human readable. Third, the relations can be *intuitively* correct from a human point of view or they may be symbolic relations to capture *synthesized* relations. It is considered that a leg of a table is a part of the table (*intuitive* relation) and a book put on the table at a certain time, also, can be a part of that table at the time (*synthesized* relation).

With these in mind, we have studied the ontological relation set formalized in [10] containing the foundational relations which can hold between entities as *individuals* (e.g. I, my car, …), *universals* (e.g. human being, cars, …), and *collections* (e.g. my friends, my previous cars, …). In addition, we have inserted the relation *equal-to* which reflects the equality of the individual entities. Table 1 shows all of the relations where the objects 1 and 2 are the signatures of the object-type dependent ontological relations. For example, the relation *individual-part-of* can only hold between individuals, while the relation *member-of* can only be realized when considering an individual and a collection of individuals as signatures.

There are some important aspects about the relations: i) all these relations (except for the relation *equal-to*) are one-way directed relations, ii) they are extracted specific to the frames. iii) It is possible to have inference over the first level relation instances and infer new relation instances in a single frame (not covered in our first version). iv)

One alternative approach to find the relations could have been to use the existing semantic types (STs) in FrameNet and their mapping [11] to the SUMO [12] nodes; however, as there are very few STs defined over the FEs it was not practical to benefit from this property of FrameNet. The SUMO relations over STs, also, can not be adapted for the different time slices of the complex events.

Table 1. Ontological relation set used to extend FrameNet

Object 1	Object 2	Ontological Relation	Example
Individual	Individual	Individual-part-of	You-Your left hand
Individual	Universal	Instance-of	You-Human being
Individual	Collection	Member-of	You-University people
Universal	Universal	Taxonomic inclusion (is-a)	Tiger-Mammal
Universal	Universal	Partonomic inclusion of universals	Animal-Mammal
Collection	Universal	Extension-of	Australian tigers-Tigers
Collection	Collection	Partonomic inclusion of collections	Body parts-Hand parts
Collection	Individual	Partition-of	Your body elements-You
Individual	*Individual*	*Equal-to*	*You-Your name*

We chose to start with this set because: i) the time-dependent relation instances in this set make the ontology instances tuned with the exact time frame of the events, and ii) the good generalization characteristic of this set is translated into a more effective inference over the text using automated inference engines.

The ontological extension can be formalized with respect to the different statuses of an event. For *stative* frames this is viewed as single-status while for *causative* and *inchoative* frames, which are mainly concerned with ongoing events, this is regarded as multiple-status. For the former, the relations hold with no change over time. Considering the latter, however, we emphasize three steps in the events: the *beginning*, *middle*, and *end* of the scenarios. For instance, consider a brief scenario of the frame *Sending* where a *sender* sends a *theme* to a *recipient* in a *container*. There are all three statuses in the scenario each of which are related to a particular set of ontological relations between the participant roles. At the beginning, the *theme* is with the *sender*, in the middle of the scenario the *container* embraces the *theme*, and at the end, the *recipient* owns the *theme*. With such a perception, the ontological relations in this scenario are: i) beginning: theme *individual-part-of* sender, ii) middle: theme *individual-part-of* container, and iii) end: theme *individual-part-of* recipient.

For the time being, we have formulated the relations realized at the end of the scenarios for all frames in FrameNet 1.2 (when they are multi-status). Table 2 shows some instances of our work. There are both types of intuitive and synthesized relations in this table. The entry with the *Building* and *Type* FEs in the *Buildings* frame shows the intuitive *instance-of* relation while the other entry with *Clothing* and *Wearer* in the frame of *Wearing* denotes a synthesized *individual-part-of* relation.

The first version of the relation instance set (having the end of the scenarios considered) for the FrameNet 1.2 database (containing 609 semantic frames) is now complete and will be publicly available soon. Table 3 shows some statistics about the relations and their frequency of occurrence in this version.

Table 2. Some ontological relations over the FEs in FrameNet 1.2

Frame	FE-1	Ontological Relation	FE-2
Abounding-with	Theme	Individual-part-of	Location
Buildings	Building	Equal-to	Name
Buildings	Building	Instance-of	Type
Cause-expansion	Item	Member-of	Group
Wearing	Clothing	Individual-part-of	Wearer

Table 3. Statistical information of the relation instances extracted for FrameNet 1.2

Ontological Relation	Times Occurred
Individual-part-of	491
Instance-of	64
Member-of	26
Taxonomic inclusion (is-a), Partonomic inclusion of universals, Extension-of, Partonomic inclusion of collections, and Partition-of	0
Equal-to	87

3.2 Representation

The axiomatic formalization explained in [10] is based on a sophisticated theory which can be exploited for the ontological extension of FrameNet, especially with respect to its characteristic of time-dependency which suits the single and multi-status events in the scenarios of FrameNet. For the time being, we have not used this formalization; however, organized and contributed the outcome of the exploration over the frames in the FrameNet XML database. It contains all of the information about the frames plus the ontological relations (i.e. ors field). Table 4 shows an instance ontological relation set added to a frame.

Table 4. Part of the ontological extension on the frame *Abounding-with* between the FEs *Theme* and *Location* at the end of the scenario

Ontological Relation in XML

```
<ors>
  <or id="…" type="individual-part-of" event_status="end">
    <signatures>
      <s1>Theme</s1>
      <s2>Location</s2>
    </signatures>
  </or>
   …
</ors>
```

4 Application: Predicate Chain Analysis

In many applications identifying the entities and understanding their relations in texts is crucial. The entities may be known as *Named Entities (NEs)* as introduced in the literature of *Information Extraction* [13]. Therefore, free texts can be viewed as the collections of NEs and their interactions forming the relations.

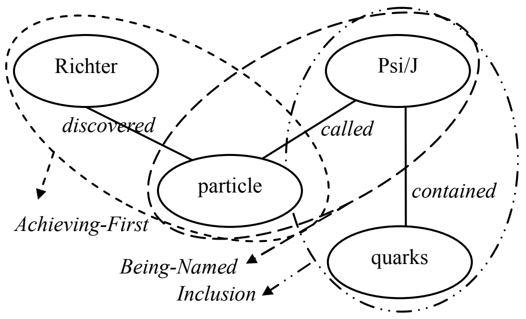


Fig. 1. Entities and their relationship using FrameNet – Entity View

FrameNet has been used for information extraction purposes [14]. However, focusing on the predicate chain analysis to find semantic connections between the informative pieces of the texts, our proposed ontological extension of FrameNet offers a methodology of recognizing relations between the NEs and/or information pieces while processing the texts on the fly. In the following example text: "*In 1974, using beams of electrons and antielectrons, or positrons, Richter discovered [a] particle that came to be called Psi/J. It contained two quarks possessing a previously unknown flavor called charm...*" there are a few entities: "1974", "beams", "electrons", "antielectrons", "positrons", "Richter", "particle", "Psi/J", "quarks", "flavor", and "charm". The identification of such entities is a task of information extraction, while the recognition of their connectivity and semantic relationship is the main problem in predicate chain analysis. These relations mostly realized by the verb phrases between the entities lead us to the use of FrameNet as shown in Figure 1.

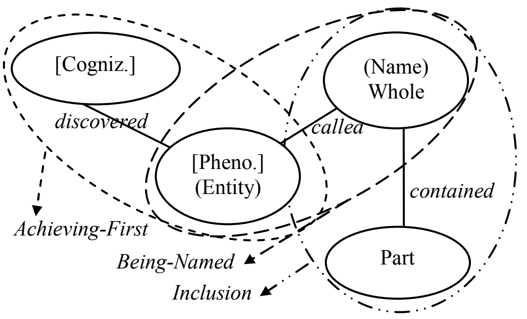


Fig. 2. Entities and their relationship using FrameNet – FE View

Such connectivity between the entities of the text is viewed as semantic relations between the participating roles (i.e. FE instances) in the scenarios of the frames evoked by the target predicates. Those FEs of each frame which share the same instance values make the connectivity between the scenarios. In Figure 1, "particle" is the instance value for the FE of *Phenomenon* of *Achieving-first* frame and at the same time it is the instance value for the *Entity* FE of the frame of *Being-named*. This shared instance value initiates the connectivity between the scenarios. Figure 2 illustrates the same connectivity of the example text from the FEs perspective.

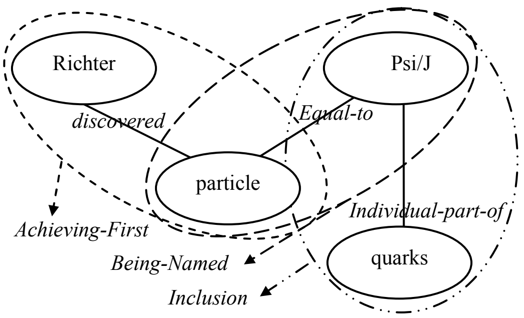


Fig. 3. Entities and their relationship using FrameNet – Ontological View

The connectivity shown in Figure 2, though meaningful for humans, cannot be exploited by a machine to perform any automated reasoning before making the links understandable for such a system. Our proposed ontological view on the FEs of FrameNet offers a sophisticated method of understanding such ordinary connections. Figure 3 demonstrates the ontological view of the example text. The first link "discovered" is not mapped to a more general view as the predicate *discover.v* does not have any relational meaning (according to the relation set). The mappings process is based on the relations already extracted on the FEs of each frame in FrameNet. With this extension, it is now possible for an automated system to infer: i) Richter discovered Psi/J, ii) Richter discovered quarks, and etc. The reasoning elements can have different certainty levels according to the inference engine and mechanism used.

5 Case Analysis in Question Answering

One of the most attractive challenges in text processing is the extraction of focused text snippets as the answers to natural language questions. This is performed by QA systems in lots of different fashions.

FrameNet has been used for QA in a few attempts [8] [15]. However, the main focus of our work is on answering questions whose answer passages contain a network of lexical relations rather than event-driven connections like in [8]. According to a survey on the TREC 2004 factoid questions one of the semantic barriers which interfere with the systems to achieve better performance is the complicated and massive semantic links between the entities in the answer-bearing texts. In a lot of ways, such answers might be extracted as a result of a naïve and semantically poor analysis over the text to find categorically related NEs (with a low level of confidence). For instance, the question "*Who discovered quarks?*" which is related to the answer-bearing passage in the example of the past section can be answered by a list of all *Person* NEs occurring in the passage. Such a list may contain totally irrelevant names of people contained anywhere in the text of the passage.

Using the existing FE-to-FE relations across the frames of FrameNet initiated by their corresponding frame-to-frame relations, it is not possible to answer questions with such answer passages. That is, the strategy explained in [9] cannot identify the candidate answer behind the predicate chain. This is simply due to the fact that there is no frame-to-frame relation (path for reasoning) between the frames *Achieving-first*, *Being-named*, and *Inclusion*.

However, this question may be answered after our proposed predicate chain analysis and performing an inference over the partial knowledge elements. We have used plausible reasoning [16] in order to make these inferences. The process of answering this question starts with the annotation of both the question and answer-bearing passage as follows:

$^{\text{Achieving-first[Cognizer]:}}$ Who $^{\text{Achieving-first[Predicate]:}}$ discovered $^{\text{Achieving-first[Phenomenon]:}}$ quarks?
$^{\text{Achieving-first[Time]:}}$ In 1974, $^{\text{Achieving-first[Instrument]:}}$ using beams of electrons and antielectrons, or positrons, $^{\text{Achieving-first[Cognizer]:}}$ Richter $^{\text{Achieving-first[Predicate]:}}$ discovered $^{\text{Achieving-first[Phenomenon]:}}$ particle $^{\text{Being-named[Entity]:}}$ particle that came to be $^{\text{Being-named[Predicate]:}}$ called $^{\text{Being-named[Name]:}}$ Psi/J. $^{\text{Inclusion[Whole]:}}$ It $^{\text{Inclusion[Predicate]:}}$ contained two $^{\text{Inclusion[part]:}}$ quarks...

Since there is no perfect match between the FEs of the frame *Achieving-first* (their instance values do not match), an inference mechanism is run over the ontological view of the text of the answer passage on the fly. Table 5 and Table 6 show both possible inferences using backward and forward chaining respectively in a plausible reasoning engine. Te approach assumes that the anaphoric references and word sense disambiguation of the predicates in the text are already resolved which makes it possible to match the entities instantiated in the FEs across the frames.

Table 5. Inference over the ontological view of the answer-bearing text – Backward chaining

Backward Chaining
Inf-1: Individual-part-of(Psi/J) = quarks $\rightarrow$ Psi/J = quarks [C_1] (Gen.)
Inf-2: Equal-to(Psi/j) = particle [C_2]
Inf-3: Inf-2 *AND* inf-1 $\rightarrow$ quarks = particle [C_1*C_2]

In the backward chaining process, the inference starts with "quarks" as the target to be matched against the FE instance values of the question. The first row, inf-1, reveals that "Psi/J" can be replaced by "quarks" with the certainty of C_1 according to the type of the inference (i.e. *Generalization*). The second row, Inf-2, results in a simple equality which denotes that "Psi/J" and "particle" are referring to the same entity. Finally, Inf-3, using both Inf-1 and Inf-2 extracts the new fact that "particle" can be replaced by "quarks" with the certainty of C_1*C_2.

The forward chaining process, however, begins with the FE instance value in the passage rather than the targeted string "quarks" in the question. This is because of the fact that the frame Achieving-first is the match frame between the question and the passage. The same result is achieved after three partial inferences.

Table 6. Inference over the ontological view of the answer-bearing text – Forward chaining

Forward Chaining
Inf-1: Equal-to(particle) = Psi/J → particle = Psi/J [C_1]
Inf-2: Individual-part-of(Psi/J) = quarks → Psi/J = quarks [C_2] (Gen.)
Inf-3: Inf-2 *AND* Inf-1 → particle = quarks [C_1*C_2]

Each certainty value, as mentioned, in this approach is calculated based on the type of the inference element of the plausible reasoning applied with respect to the ontological relation type. The highest value (i.e. 1.0) is assigned to the inferences based on the e*qual-to* relations, while the generalization inferences regarding the i*ndividual-part-of* relations get lower certainty values.

Via FE alignment it is now possible to extract the answer as the instance value of the vacant FE of the annotated question in the answer passage. The overall certainty calculated as the product of the partial certainty values, provides a much more meaningful assessment of this as an answer compared to the ones of the shallow information extraction based approaches like NE-oriented answer extraction. The normalization over predicate chains by converting the arbitrary relations to the ontological relations along with the functionality of the reasoner results in answering more questions regarding the passage and its ontological instance (Figure 4).

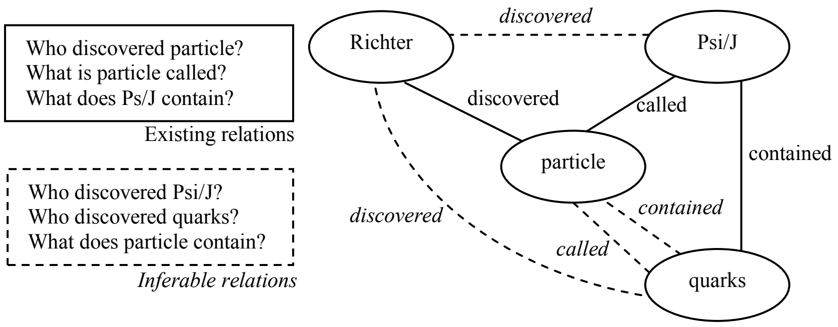


Fig. 4. The more knowledge acquired from the text the more questions to be answered

6 Conclusion

We have developed an ontological extension on FrameNet (version 1.2) with a specific focus on the FEs within the frames. This extension, using a limited number of relations in an XML representation is consistent with and contributed to the current FrameNet database offering a wide range of applications in text processing and understanding. In our work, the benefit of this extension has been explored in predicate chain analysis when texts are viewed as networks of relations (semantic connections) between the entities. We have applied the theory of predicate chain analysis to QA in conjunction with an inference engine (plausible reasoner). The partial knowledge elements acquired from the ontological view of the texts are gathered and yield more knowledge via the reasoning engine. Such extracted knowledge is exploited to more semantically answer questions compared to a shallow semantic and/or statistical NE extraction methodology.

The main direction of our future work is to develop the representation through axiomatic formalization. The completion of the set of relations over the FEs with respect to the different statuses in multi-status events of the FrameNet frames is another part of the future work. We will also extend the first version of the relation instances to the FrameNet database version 1.3.

References

1. Baker, C.F., Fillmore, C.J., Lowe, J.B.: The Berkeley FrameNet Project. In: International Conference On Computational Linguistics, vol. 1, pp. 86–90 (1998)
2. Fillmore, C.J.: Frame Semantics and the Nature of Language. In: Annals of the New York Academy of Sciences: Conference on the Origin and Development of Language and Speech, vol. 280, pp. 20–32 (1976)
3. Lowe, J.B., Baker, C.F., Fillmore, C.J.: A Frame-Semantic Approach to Semantic Annotation. In: SIGLEX Workshop on Tagging Text with Lexical Semantics: Why, What, and How? (1997)
4. Fillmore, C.J.: The Case for Case. Universals in Linguistic Theory, 1–88 (1968)
5. Cook, W.A.: Case Grammar Theory. Georgetown University Press, Washington (1989)
6. Petruck, M.R.L.: Frame Semantics. In: Handbook of Pragmatics Online (1996)
7. Ruppenhofer, J., et al.: FrameNet: Theory and Practice (2005), http://framenet.icsi.berkeley.edu/
8. Narayanan, S., Harabagiu, S.: Question Answering Based on Semantic Structures. In: COLING 2004. International Conference on Computational Linguistics, pp. 693–701 (2004)
9. Scheffczyk, J., Baker, C.F., Narayanan, S.: Ontology-based Reasoning about Lexical Resources. In: Proceedings of OntoLex 2006, pp. 1–8 (2006)
10. Bittner, T., Donnelly, M., Smith, B.: Individuals, Universals, Collections: On the Foundational Relations of Ontology. In: Varzi, A.C., Vieu, L. (eds.) FOIS 2004. Proceedings of the Third Conference on Formal Ontology in Information Systems, pp. 37–48 (2004)
11. Scheffczyk, J., Pease, A., Ellsworth, M.: Linking FrameNet to the Suggested Upper Merged Ontology. In: FOIS 2006. Proceedings of the 2006 International Conference on Formal Ontology in Information Systems (2006)
12. Niles, I., Pease, A.: Towards a Standard Upper Ontology. In: International conference on Formal Ontology in Information Systems, pp. 2–9 (2001)
13. Sang, E.F.T.K., Meulder, F.D.: Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition. In: Proceedings of CoNLL 2003, pp. 142–147 (2003)

14. Moschitti, A., Morarescu, P., Harabagiu, S.M.: Open Domain Information Extraction via Automatic Semantic Labeling. In: Proceedings of FLAIRS Conference 2003, pp. 397–401 (2003)
15. Kaisser, M.: QuALiM at TREC 2005: Web-Question Answering with FrameNet. In: TREC 2005. Proceedings of the Fourteenth Text REtrieval Conference (2005)
16. Collins, A., Michalski, R.: The Logic of Plausible Reasoning A Core Theory. Cognitive Science 13, 1–49 (1989)

Using Clustering for Web Information Extraction

Le Phong Bao Vuong and Xiaoying Gao

School of Mathematics, Statistics and Computer Science
Victoria University of Wellington, PO Box 600, Wellington, New Zealand
{Phong.Le, Xiaoying.Gao}@mcs.vuw.ac.nz

Abstract. This paper introduces an approach that achieves automated data extraction from semi-structured Web pages by clustering. Both HTML tags and the textual features of text tokens are considered for similarity comparison. The first clustering process groups similar text tokens into the same *text clusters,* and the second clustering process groups similar data tuples into *tuple clusters*. A *tuple cluster* is a strong candidate of a repetitive data region.

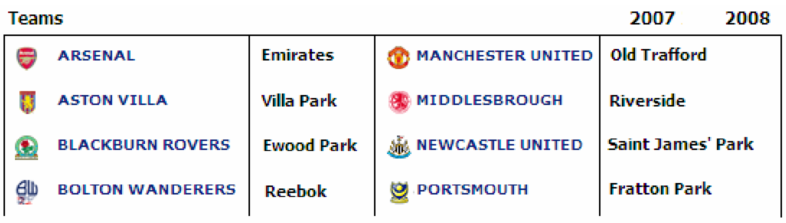
Keywords: Information extraction, clustering, Smith-Waterman algorithm.

1 Introduction

Semi-structured Web pages often contain repetitive data regions such as foreign currency exchange tables or online shopping product lists. Information Extraction (IE) systems (wrappers) automatically extract these useful data items from the HTML sources so that they can be used by software programs for data mining, knowledge discovery and accurate search, etc. Wrapper induction is a technology that uses machine learning to learn IE patterns from training example pages to be used to extract data from Web pages. However, information exaction using wrapper induction technology has some limitations.

Existing wrapper induction systems [1 - 4] require training Web pages from each site. These systems assume that there are multiple Web pages with similar data records in similar data format available on the same site. Human effects (or other programs) are needed to find the training pages [3, 4] and/or to manually label the training pages [1, 2].

There have been some automatic systems that do not require labeling [3, 6] but their performance on repetitive data record detection is not satisfactory. RoadRunner [3] requires at least two training pages with different numbers of data records to identify the individual data records. Both RoadRunner and MDR [6] have difficulties handling Web pages that represent multiple data records on the same rows. As an example, Figure 1 shows a fragment of a Web page which has a table containing two data records on each row. While wrapper induction systems like RoadRunner or MDR can extract data from this page, they have no idea that the four columns should be interpreted as two data records. Instead, a list of quadruples *(teamName, stadiumName, teamName, stadiumName)* is generated as the result.



Teams		2007	2008
ARSENAL	Emirates	MANCHESTER UNITED	Old Trafford
ASTON VILLA	Villa Park	MIDDLESBROUGH	Riverside
BLACKBURN ROVERS	Ewood Park	NEWCASTLE UNITED	Saint James' Park
BOLTON WANDERERS	Reebok	PORTSMOUTH	Fratton Park

Fig. 1. Two data tuples are on the same rows

We propose a different approach which uses clustering to automatically extract data from a single un-labeled semi-structured Web page without any human intervention. In particular, a HTML parser is used to parse a single HTML input page into a list of text tokens. Then, clustering is applied on these text tokens to group similar text tokens into *text clusters*. As a result, similar text tokens are grouped into the same *text clusters* and labeled with the same *text cluster IDs*. To decide whether two text tokens are similar, we employ both HTML tags and the textual features of text tokens such as HTML delimiters, relative 2D coordinates and part-of-speech of the text tokens. Next, data tuples (or data records) are constructed based on a refined DOM tree of the HTML document. Similar data tuples are identified by applying Smith-Waterman algorithm on the sequences of *text cluster IDs* of the text tokens that the data tuples contain. Again, clustering is applied to group similar data tuples into *tuple clusters*, each of which constitute to a candidate repetitive data region.

The remainder of the paper is organized as follows. Section 2 summarizes briefly the related work. Section 3 describes the text token representation and the text token clustering process. Section 4 shows how data tuples are constructed from the refined DOM tree and how repetitive data regions are identified from the results of the data tuple clustering process. Section 5 demonstrates the experimental results on two sets of Web sites and shows comparisons between the proposed approach and another existing approach. Finally, section 6 concludes the paper.

2 Related Work

As shown in a survey paper [5], many wrapper induction systems have been built for learning information extraction patterns to extract data from semi-structured Web pages. Most of earlier wrapper induction systems such as WIEN [1] and Stalker [2] require manually labeled training pages, which need considerable amount of human efforts. One recent wrapper induction system RoadRunner [3] can learn from unlabelled Web pages but it requires at least two training pages generated with the same template, which may not be available for many Web sites. Moreover, from our experiments with RoadRunner on the same sets of test data that we used to test our system, we discovered that repetitive data records were correctly identified by RoadRunner only if at least one of the example pages has a different number of repetitive data records. If the training pages have the same number of data records, which is very common, the *recursive mismatch module* [3] cannot detect the repetitive data records and results in one giant data record containing all data fields. Different from the others, AutoWrapper [4] is a relatively new system that uses one single unlabelled Web page to learn patterns but it is restricted to tabular pages.

Regarding the motivation of this research, the closest related work is MDR [6] which does not require training. The main technology of MDR is based on a string matching algorithm. MDR can handle non-contiguous data records, but it cannot deal with HTML pages similar to the one demonstrated in Figure 1. Regarding to the technologies used by existing systems, the closest related work to our approach is an approach by Lerman et al [7]. It uses a clustering algorithm to exploit data format and data content to classify data, but it uses a different clustering algorithm and totally different similarity measures. Moreover, it uses grammar induction for identifying data record. Their system requires multiple pages, and is restricted to pages with tables and lists.

3 Text Token Clustering

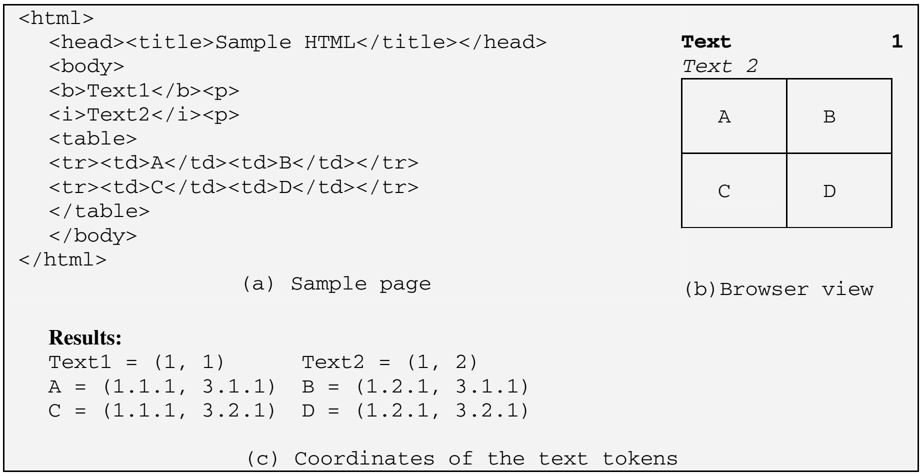
We transform a semi-structured Web page into a sequence of text tokens. This is done by a HTML parser to parse the input page into an intermixed list of HTML tags and strings of characters between the HTML tags. Text tokens capture these strings of characters as main values.

3.1 Text Token Representation

A text token has six main features, three of them are basic elements namely *string value, left delimiter* and *right delimiter*. The string value of a text token represents a string of characters between HTML tags captured by the text token. The left and right delimiters are the sequences of consecutive HTML tags preceding and succeeding the string value, respectively. A tag is the basic unit of a delimiter of a text token. A tag is represented by a tag name which is the name of the HTML tag element, a tag type which is either OPEN or CLOSE, and a set of attributes inside HTML.

The fourth feature *Parent Nodes* is used to capture the hierarchical structure of the text tokens. We construct a refined HTML DOM tree (Document Object Model tree) which contains only block tags such as <TABLE>, <TR> and <UL> from the original DOM tree. Non-block tags such as <B>, <I> and <BR> do not have any contribution into building the hierarchy and therefore are ignored. The fourth feature associated with a text token is its parent nodes which is the sequence of nodes along the path from the root of the tree to the current node that contains the text token.

The last two features are *the X and Y coordinates* of the text tokens. Normally, data of the same types are rendered on the browser at the same X or the same Y coordinate. Therefore, by capturing coordinates of the text tokens, we have two useful parameters to determine whether two text tokens are similar. Furthermore, instead of blindly comparing any pair of text tokens during the text token clustering process, only those having the same X or Y coordinate will be taken into account. Different browsers might render Web pages in different ways, so relative X and Y coordinates of the text tokens are captured instead of their absolute counterparts. We use a tool developed in the WebCoords project [9] to get these values. As an example, Figure 2 displays a sample HTML page and the result returned from the tool which shows the relative coordinates of the text tokens.



```
<html>
  <head><title>Sample HTML</title></head>
  <body>
  <b>Text1</b><p>
  <i>Text2</i><p>
  <table>
  <tr><td>A</td><td>B</td></tr>
  <tr><td>C</td><td>D</td></tr>
  </table>
  </body>
</html>
```

(a) Sample page

Text 1

Text 2

A	B
C	D

(b)Browser view

Results:

```
Text1 = (1, 1)        Text2 = (1, 2)
A = (1.1.1, 3.1.1)  B = (1.2.1, 3.1.1)
C = (1.1.1, 3.2.1)  D = (1.2.1, 3.2.1)
```

(c) Coordinates of the text tokens

Fig. 2. webCoord's example

3.2 Text Token Similarity Comparison

In this section, we describe how to measure the similarities between text tokens. Two text tokens are similar if 1) their delimiters are similar as defined in section 3.2.1, and 2) their text string values are similar as defined in section 3.2.2, and 3) they must have either the same X or the same Y coordinate.

3.2.1 Delimiter Similarity

As each delimiter is actually a sequence of tags, Smith-Waterman algorithm [8] is selected to compute the similarity between two delimiters. Originally, the Smith-Waterman algorithm is used to identify maximally homologous sub-sequences in two molecular sequences. In our case, we are not interested in the returned maximally similar sub-sequences but the best matching score computed in the intermediate step using the similarity measures. The best score is used in our system to determine whether two delimiters are similar.

We use a slightly different similarity function to compute the similarity between two tags. Let *a* and *b* be any elements of the two sequences under consideration, *S(a; b)* is called the similarity function of *a* and *b*. In the original Smith-Waterman algorithm, *S(a; b) = 1* if *a* and *b* match, and *S(a; b) = -1/3* if they do not match. When applying the algorithm to the delimiters, we use a different similarity function defined as follows:

$$\mathrm{S}(\mathrm{Tag}_1;\ \mathrm{Tag}_2) = \begin{cases} 1.0 \ \textit{if } Tag1 \textit{ and } Tag2 \textit{ are identical} \\ ds \ \textit{if } Tag1 \textit{ and } Tag2 \textit{ are similar} \\ \text{-}1/3 \ \textit{otherwise} \end{cases} \tag{1}$$

where Tag$_1$ and Tag$_2$ are any two tags of the delimiters being considered and *ds* is the similarity degree between them which is detailed as follows.

Given two tags Tag_1 and Tag_2, they are considered identical if they have the same tag name and the same tag type (either "OPEN" such as <IMG> or "CLOSE" such as </IMG>), and all their tag attribute names are the same. Two tags are considered similar if they have the same tag name and the same tag type. In this case, the *degree of similarity (ds)* between the two tags is defined as $ds = 2*N/ (N1 + N2)$, where N1, N2 is the number of attributes of the two tags and N is the number of the attributes with the same names.

Let m and n be the number of tags in the two delimiters respectively, $L = min(m; n)$ and H is the best matching score computed by the Smith-Waterman algorithm. We defined the two delimiters are similar if the two conditions hold: $|m\text{-}n|/(m+n+1)< \theta$ and $|L\text{-}H|/(L+H+1)< \theta$.

3.2.2 String Values Similarity

The string value in a text token contains the data content and data contends of the same data fields are often similar. Therefore, it is important for the clustering algorithm to exploit the similarity of the string values to group similar text tokens together.

We use the vector space model (VSM) [10], which is commonly used in document classification and information retrieval, to construct the similarity measure for the string values. VSM represents natural language documents in a formal manner by the use of vectors in a multi-dimensional space. In our case, a document under consideration is the string value of a text token. Each document in the vector space model is represented as a vector based against a vocabulary. The vocabulary is often built from all the words in each document. However, our observation shows that most of the words in the string values of the same data fields (data attributes) are not the same, e.g. "ARSENAL" and "ASTON VILIA" as values of Team Names.

We identify three important features for similarity comparison of string values. 1) String values of the same data fields often share the same textual patterns. Consider two text tokens with string values of email addresses: "*Phong.Le@mcs.vuw.ac.nz*" and "*Xiaoying.Gao@mcs.vuw.ac.nz*". If we compare word by word then we will conclude that they are not similar. For our purpose, however, they should have been considered similar because they both conform to the email format and they represent values of the same data field. These kinds of data with predefined format can be identified easily using regular expression matching techniques. 2) Text data sometimes contain special entities such as team names, club names, person names, organization names, city names and country names that can be identified using named-entity recognition techniques [11], although named-entity recognition is a big research area itself. 3) More importantly, text data contain part-of-speech features which can be retrieved by part-of-speech tagging approach.

We use the three important features to build our own vocabulary. We focus on part-of-speech tagging in building the vocabulary used in VSM. Regular expression matching techniques and named-entity recognition are used as a supplement for the part-of-speech tagging. At first, regular expression matching is used to identify and classify some text data into one of the following predefined terms: email, date, time, currency, number, telephone, mobile number and URL. We then use a named-entity recognition tool to identify recognizable entities such as team name, person name, car branch, country name, etc. The rest of the text tokens are tagged with a part-of-speech term such as noun, adjective, verb, adverb, etc. By identifying and transforming each word within a text token into predefined terms, we have a vocabulary for the VSM model.

We use the standard *Tf-Idf* method to represent each document into a vector. Given two string values represented by two vectors X and Y, we define two string values are similar if the cosine similarity between X and Y is greater than a threshold $\theta \approx 0.6$.

3.3 Text Token Clustering

In our system, Hierarchical Agglomerative Clustering algorithm (HAC) [12] is used for text tokens clustering. The same clustering algorithm is applied in the data tuple clustering process that will be detailed in next section. Note we use the algorithm to compare only pairs of text tokens which have either the same X or the same Y coordinate, so the efficiency is improved significantly.

After the text token clustering process, similar text tokens are expected to be in the same *text clusters* and labeled with the same *text cluster IDs*. These *text cluster IDs* are very useful factors for measuring the similarities between data tuples which are subsequently required by the data tuple clustering process. In the next section, we firstly describe how data tuples are constructed from a refined DOM tree of the HTML document, and then outline how repetitive data regions are identified by applying clustering on the data tuples.

4 Identifying Repetitive Data Tuples by Clustering

Within a Web page, a repetitive data region contains a list of similar data tuples which subsequently contain similar sequences of text tokens. Basically, a data tuple contains one or more related text tokens, each of which can be represented by a *text cluster ID*. By transforming each data tuple into a sequence of *text cluster IDs*, the Smith-Waterman algorithm can be applied on any two of these sequences to determine whether two data tuples are similar. We group similar data tuples into *tuple clusters*, each of which is considered a repetitive data region if its number of elements is greater than one.

4.1 Data Tuple Identification

From observing several Web pages, we concluded that text tokens belonging to the same data tuple often share the same parent node in the DOM tree described in Section 3.1. For example, the tree depicted in Figure 3 contains three data tuples (“*John*”, *45*, “*New Zealand*”), (“*Daniel*”, *43*, “*Germany*”) and (“*Next*”, “*Previous*”). The three text tokens in the first tuple share the same <TR> parent node (on the left) and likewise, the text tokens in the second and third tuple share the same <TR> parent nodes. Therefore, by traversing from each leaf node (which is the node containing a text token) to the root node and finding out the nearest common “joint” nodes of the traversing paths, we can “connect” related text tokens together into data tuples.

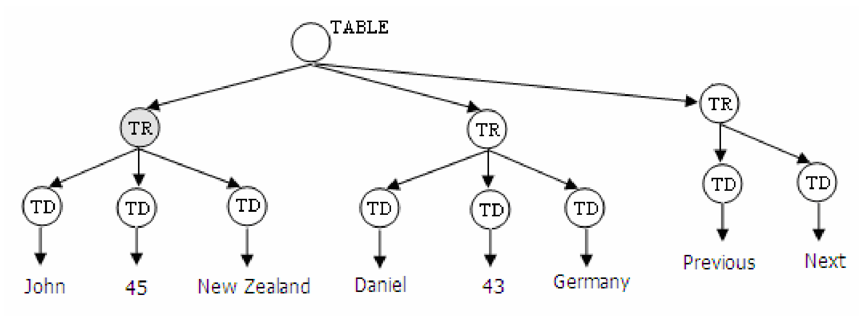


Fig. 3. Data tuple ("John", "45" and "New Zealand") share the same TR parent node

However, there are some cases in which all text tokens within a repetitive data region share the same single parent node, so all text tokens are merged into one giant tuple. In this case, we split the giant data tuple by using string alignment technique to find the repetitive pattern within the sequence of *text cluster IDs* which represents the test tokens in the data tuple. For the example page shown in figure 1, the four text tokens in the same table row share the same parent node, so initially they are connected as a data tuple. Suppose the data tuple is represented as a sequence of *cluster IDs* (#1, #2, #1, #2), the string alignment technique is used to find the repetitive pattern (#1, #2) to split it into two data tuples.

4.2 Data Tuple Clustering

In order to identify repetitive data regions, we group similar data tuples into *tuple clusters*. Again, HAC algorithm is applied, but this time on the data tuples, based on the similarity measure returned by the Smith-Waterman algorithm. Given two data tuples, the Smith Waterman algorithm computes the similarity score between the two sequences of *text cluster IDs* which represent the text tokens in the two data tuples. Two data tuples are defined as similar if the similarity score between them exceeds a threshold.

As an example, assume that the three data tuples in Figure 3 are transformed into (*#1, #2, #3*), (*#1, #2, #3*) and (*#4, #5*) respectively. By applying the Smith-Waterman on these sequences, we can conclude that ("*John*", *45*, "*New Zealand*") and ("*Daniel*", *43*, "*Germany*") are repetitive and similar. These two data tuples and other possibly other similar data tuples form a repetitive data region. ("*Next*", "*Previous*") should be grouped into a different tuple cluster (which may not be a repetitive data region). Depending on the selected threshold, sequences like *(#1, #2, #3, #4*) and *(#1, #2, #6, #4*) may also be considered as similar.

5 Experimental Results

To investigate whether our approach is effective, we developed a prototype system called *ClusterEx*. This system was examined on two groups of Web sites and compared with MDR [6] which is a relatively new system and is available online.

5.1 Experiment 1

The first group of experiments was run on a collection of 60 Web pages which were downloaded directly from the Internet by the time the experiments were conducted. The results of our *ClusterEx* system and another system *MDR* are shown in Table 1. The third column of the table shows the numbers of repetitive data records. Column 4 to 7 show the *Precision* and *Recall* values calculated from the results returned by ClusterEx and MDR respectively.

Table 1. First Experimental Results

No	URL	Objs	ClusterEx		MDR	
			Precision	Recall	Precision	Recall
1	addons.mozilla.org	10	100%	100%	0%	0%
2	blogsearch.google.com	10	100%	100%	100%	100%
3	citeseer.ist.psu.edu	15	100%	93%	0%	0%
4	groups.google.com	10	100%	100%	100%	100%
5	groups.yahoo.com	10	100%	90%	100%	90%
6	ieee.org/conferenceSearch	12	100%	100%	100%	100%
7	jobsearch.gov.au	20	100%	100%	100%	100%
8	news.google.com	10	100%	100%	100%	100%
9	www.kellysearch.com	100	100%	99%	50%	0%
10	scholar.google.com	10	100%	100%	0%	0%
11	search.experts-exchange.com	10	100%	100%	0%	0%
12	search.xtramsn.co.nz	10	100%	100%	83%	100%
13	search.yahoo.com	10	100%	100%	0%	0%
14	sourceforge.net	10	100%	100%	100%	100%
15	travel.travelocity.com	19	100%	100%	0%	0%
16	www.aatravel.co.nz	10	100%	100%	100%	100%
17	www.thomasnet.com	25	100%	100%	100%	100%
18	www.apple.com	40	90%	100%	100%	100%
19	www.aspfree.com	10	100%	90%	100%	100%
20	www.bayleys.co.nz	10	100%	70%	100%	100%
21	www.booking.com	15	100%	100%	0%	0%
22	www.careerbuilder.com	25	100%	100%	100%	100%
23	www.dhtmlgoodies.com	17	100%	100%	25%	100%
24	www.download.com	10	100%	100%	100%	100%
25	www.dse.co.nz	20	100%	100%	100%	100%
26	www.dynamicdrive.com	15	100%	100%	0%	0%
27	www.ferrit.co.nz	10	100%	100%	100%	100%
28	www.fishpond.co.nz	20	100%	100%	100%	100%
29	www.google.com	10	100%	100%	100%	100%
30	vuw.ac.nz - scholarships	39	100%	100%	93%	100%
31	www.history.com	23	100%	96%	53%	100%
32	www.hotscripts.com	20	100%	100%	100%	100%
33	www.infospace.com	15	100%	100%	100%	100%
34	www.open2view.com	10	100%	100%	100%	100%
35	www.oreilly.com	10	100%	100%	0%	0%
36	www.bbc.co.uk/food	15	100%	93%	100%	100%
37	www.planet-source-code.com	10	100%	100%	100%	100%
38	www.portfolios.com	10	100%	100%	100%	100%
39	www.foodieview.com	10	90%	100%	90%	90%
40	study.massey.ac.nz	20	100%	100%	100%	100%
41	www.realestate.co.nz	10	100%	100%	0%	0%
42	www.daypop.com	10	100%	100%	91%	100%
43	www.rsc.org	10	100%	100%	100%	100%
44	www.sharetool.com	15	100%	100%	100%	100%
45	www.tommys.co.nz	10	100%	100%	100%	100%
46	www.trademe.co.nz	50	100%	100%	100%	100%
47	www.tucows.com	20	100%	100%	87%	100%
48	www.webcrawler.com	20	100%	100%	0%	0%
49	www.cooks.com	10	100%	100%	100%	100%
50	www.wrox.com	19	100%	100%	100%	100%
51	www.pconlineshop.co.nz	12	100%	100%	0%	0%
52	www.trademe.co.nz	20	100%	90%	0%	0%
53	www.beautifulwallpaper.com	8	100%	100%	100%	100%
54	www.lyrics.com	20	100%	100%	100%	100%
55	www.smartlyrics.com	50	100%	96%	0%	0%
56	froogle.google.com	10	100%	100%	0%	0%
57	www.dvdunlimited.co.nz	10	100%	100%	100%	100%
58	www.seek.co.nz	20	100%	100%	100%	100%
59	www.parallelimported.co.nz	10	100%	100%	100%	100%
60	www.urcomputer.co.nz	49	100%	100%	100%	100%
		Average	**99%**	**98.6%**	**71.2%**	**73%**

As can be seen from Table 1, our system achieved better average *Precision* and *Recall* values which are 99% and 98.6% respectively. The MDR system, however, only achieved 71.2% for *Precision* and 73% for *Recall*, suggesting that our system outperforms the MDR system on these Web pages.

5.2 Experiment 2

Our second group of experiments was run on a special collection of 12 Web sites in which multiple data records are presented on the same rows. Note that the fourth column of Table 2 shows the numbers of data records that are on the same rows. For example, in example #3, 12 data records are rendered on the browsers as a 3 columns x 4 rows matrix.

Table 2. Second Experimental Results

No	URL	Objs	Columns	ClusterEx		MDR	
				Precision	Recall	Precision	Recall
1	c1com.co.nz	20	2	100%	85%	100%	100%
2	www.nzoczone.com	30	30	100%	100%	0%	0%
3	www.wallpaperbase.com	12	3	100%	100%	0%	0%
4	www.wallpapers.com	13	3	100%	100%	100%	100%
5	www.extremepc.co.nz	13	4	100%	100%	100%	100%
6	www.superpc.co.nz	12	3	100%	100%	100%	100%
7	www.einfo.co.nz	17	2	100%	100%	100%	100%
8	e-zone.co.nz	15	3	100%	100%	100%	100%
9	www.ifocus.co.nz	6	2	100%	100%	100%	100%
10	www.tiscowaikato.co.nz	15	3	100%	100%	100%	100%
11	uitservice.com	19	3	100%	100%	100%	100%
12	www.promax.co.nz	11	3	100%	100%	100%	100%

As shown in Table 2, ClusterEx has successfully extracted data from all example pages. There is only one case (example #1) in which 3 data records are missing because of the poor clustering result. On the other hand, MDR was unable to extract data from example #2 and #3. In these two examples, multiple data records are merged incorrectly, e.g. in example #3, 3 incorrect data records are returned, each of which contains 4 merged data records. With the rest of the examples, MDR failed to return the correct numbers of repetitive data regions. Although all data records have been extracted successfully, they are split into many smaller repetitive data regions while there is actually only one data region.

6 Conclusion

The goal of this paper was to investigate an approach to the use of clustering for identifying repetitive data records from one unlabelled semi-structured Web page. The goal was successfully achieved by: (1) defining a number of similarity measures for judging whether two text tokens are similar, (2) developing a method to construct and group data tuples together in order to detect repetitive data regions.

The approach was examined and compared with another similar approach on two groups of Web sites. The results suggest that our new approach is effective for information extraction from semi-structured Web pages, especially for those having multiple data records on the same rows.

References

1. Kushmerick, N., Weld, D.S., Doorenbos, R.B.: Wrapper induction for information extraction. In: IJCAI. Intl. Joint Conference on Artificial Intelligence, pp. 729–737 (1997)
2. Muslea, I., Minton, S., Knoblock, C.: A hierarchical approach to wrapper induction. In: Etzioni, O., Muller, J.P., Bradshaw, J.M. (eds.) Agents 1999. Proceedings of the Third International Conference on Autonomous Agents, pp. 190–197. ACM Press, New York (1999)
3. Crescenzi, V., Mecca, G., Merialdo, P.: Roadrunner: Towards automatic data extraction from large web sites. In: Proceedings of 27th International Conference on Very Large Data Bases, pp. 109–118 (2001)
4. Gao, X., Andreae, P., Collins, R.: Approximately repetitive structure detection for wrapper induction. In: Zhang, C., W. Guesgen, H., Yeap, W.-K. (eds.) PRICAI 2004. LNCS (LNAI), vol. 3157, pp. 585–594. Springer, Heidelberg (2004)
5. Muslea, I.: Extraction patterns for information extraction tasks: A survey. In: Proceedings of AAAI Workshop on Machine Learning for Information Extraction, Orlando, Florida (July 1999)
6. Liu, B., Grossman, R., Zhai, Y.: Mining data records in web pages. In: Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 601–606 (2003)
7. Lerman, K., Knoblock, C., Minton, S.: Automatic data extraction from lists and tables in web sources. In: IJCAI 2001. Proceedings of the workshop on Advances in Text Extraction and Mining (2001)
8. Smith, T.F., Waterman, M.S.: Identification of common molecular subsequences. Journal of Molecular Biology 147, 195–197 (1981)
9. Gagnon, M., Schoudel, B., Vasilevsky, S., Wisely, A.: WebCoords Project. http://webcoords.sourceforge.net
10. Salton, G.: Introduction to Modern Information Retrieval. McGraw-Hill, New York (1983)
11. Grishman, R., Sundheim, B.: Message Understanding Conference - 6: A Brief History. In: Kopenhagen, I. (ed.) COLING. Proceedings of the 16th International Conference on Computational Linguistics, pp. 466–471 (1996)

A Decision Tree Approach to Sentence Chunking

Samuel W.K. Chan

Dept. of Decision Sciences
The Chinese University of Hong Kong
Hong Kong SAR
swkchan@cuhk.edu.hk

Abstract. This paper proposes an algorithm which can chunk a given sentence into meaningful and coherent segments. The algorithm is based on the assumption that segment boundaries can be identified by analyzing various information-theoretic measures of the part-of-speech (POS) *n*-grams within the sentence. The assumption is supported by a series of experiments using the POS-tagged corpus and Treebank from Academia Sinica. Experimental results show that the combination of different classifiers based on the measures improves the system coverage while maintaining its precision in our evaluation of 10,000 sentences.

1 Introduction

Syntactic parsing has long been considered as one of the critical tasks in Natural Language Processing (NLP). While the parsing based on the powerful grammar formalisms such as Lexical-Functional Grammar (LFG) (Kaplan & Bresnan, 1982), Tree-Adjoining Grammar (TAG) (Joshi, 1987) or Head-Driven Phrase Structure Grammar (HPSG) (Pollard & Sag, 1994) have provided an in-depth analysis of narrowly defined sets of data, it is still formidable to produce a complete parse tree in any real-world sentences. An alternative approach is the shallow parsing which tries to avoid the complexity of full parsing and analyzes a sentence at the level of phrases and the relations between them (Abney, 1991). As a branch in shallow parsing, sentence chunking is generally the first step in a cascade of classifiers intended for shallow parsing. Sentence chunking is defined to partition a sentence into non-overlapping tiles which are usually short phrases. One method of chunking Noun Phrases (NP) with a classification approach is to define the problem as a tagging problem, as suggested by Ramshaw & Marcus (1995). They assign a tag to each word in a sentence to indicate whether the word is inside an NP chunk, marked as tag *I*, or outside, marked as tag *O*. The third tag *B* is reserved for marking the beginning of an NP chunk immediately following another NP chunk. They recognize the non-recursive NPs up to the head, including determiners. These chunks are extracted from the Treebank parses, by selecting NPs that contained no nested NPs. Tjong & Veenstra (1999) combine two different classifiers for annotating the beginning and the end of NP chunks. While the first classifier is based on the IOB representation, the second one contains the Part-of-Speech (POS) tags and the IOB tags of the focus word. Argamon *et al.* (1999) use Memory-Based Sequence Learning (MBSL) to recognize shallow linguistic patterns in a bracketed training corpus. Contrary to above NP chunking approaches that assign IOB tags, the MBSL relies on fragments of POS subsequences that can be combined to cover the complete sentences. A score that measures the degree of their coverage, overlapping and maximum amount of context involved is devised. The highest-scoring candidate subsequences are used to predict all the possible chunks appeared in the sentences.

In this research, we consider sentence chunking is the first and foremost step in shallow parsing. Following the work of Ramshaw & Marcus (1995), we present a shallow sentence chunking technique and illustrate how an input POS-tagged sentence can be chunked into several segments in such a way that segments are coherent portions illustrating all the core phrases of the sentence. It targets not only on the NPs but all the possible chunks that may appear in a fully POS-tagged Treebank. The possible chunks include prepositional chunks, verbal chunks as well as basic grammatical relations such as subject-verb relations or verb-object patterns. We consider whether there is a chunking point between two focus words based on a sequence of POS tags as a classification problem. Given a set of attributes including the POS sequences and their collocation information-theoretic measures, it is to be decided whether there is a chunking point between the two words or not. One well known method for solving such classification problems is the C5.0 algorithm (Quinlan, 2000). C5.0 builds a classification tree in which each non-terminal node corresponds to the examination of features, while the terminal nodes contain class information. The classification tree, or called *classifier*, is built recursively from a set of training data that classes are known. One of our primary concerns is to design a shallow but robust mechanism which can identify the chunking points based on the rules learnt from the classifier. The organization of the paper is as follows. In Section 2, we first describe the information-theoretic measures which are extracted from the part-of-speech (POS) *n*-grams of the parse trees from the Academia Sinica Treebank. The measures indicate the likelihood that two POS fragments will co-occur. In this paper, three different sets of information-theoretic measures are applied in quantifying the POS collocation, namely the mutual information, likelihood ratio and the *t*-statistics. Unlike other similar approach, neither predefined thresholds nor confidence levels will be introduced. Instead, all these measures are used as training data. Different classifiers will be built in the C5.0 as described in Section 3. The system has already been implemented using Java language. In order to demonstrate the capability of our system, we have trained our system using 55,000 parse trees and an evaluation of 10,000 sentences has been conducted. The experimental results of the classifiers are also investigated. They are explained in Section 4 followed by a conclusion.

2 Information-Theoretic Measures of POS Sequence

A straightforward choice of attributes for chunking point detection is certainly the POS tags of the focus words with their neighbors. However, a full chunking of sentences on the basis of the POS sequence alone is generally impossible. In our approach, in addition to the POS sequence, we consider fragments in the sequence and introduce various information measures which reflect the likelihood of the fragment collocation. Let the POS sequence Ω with a window size $2n$ be $x_1..x_nx_{n+1}..x_{2n}$ and assume x_n and x_{n+1} be the POS tags of the focus words. We are concerned with more than just the interdependence between two individual POS tags, but between two POS fragments. Different various adjacent POS fragments at the neighborhood of x_n and x_{n+1} are defined in Table 1. Thus, we consider an n-gram as a 2-gram of an n_1-gram and an n_2-gram, where $n_1 + n_2 = n$ (Magerman & Marcus, 1990).

Table 1. Different measures of collocation between adjacent POS fragments

$\boldsymbol{x}_{n-3}$	$\boldsymbol{x}_{n-2}$	$\boldsymbol{x}_{n-1}$	$\boldsymbol{x}_{n}$	$\boldsymbol{x}_{n+1}$	$\boldsymbol{x}_{n+2}$	$\boldsymbol{x}_{n+3}$	$\boldsymbol{x}_{n+4}$	measure of collocation	n-gram
		$\boldsymbol{x}_{n-1}$	$\boldsymbol{x}_{n}$					d_1: $\zeta(\boldsymbol{x}_{n-1}, \boldsymbol{x}_n)$	2-gram
			$\boldsymbol{x}_{n}$	$\boldsymbol{x}_{n+1}$				d_2: $\zeta(\boldsymbol{x}_{n}, \boldsymbol{x}_{n+1})$	2-gram
				$\boldsymbol{x}_{n+1}$	$\boldsymbol{x}_{n+2}$			d_3: $\zeta(\boldsymbol{x}_{n+1}, \boldsymbol{x}_{n+2})$	2-gram
	$\boldsymbol{x}_{n-2}$	$\boldsymbol{x}_{n-1}$	$\boldsymbol{x}_{n}$					d_4: $\zeta(\boldsymbol{x}_{n-2}\boldsymbol{x}_{n-1}, \boldsymbol{x}_n)$	3-gram
		$\boldsymbol{x}_{n-1}$	$\boldsymbol{x}_{n}$	$\boldsymbol{x}_{n+1}$				d_5: $\zeta(\boldsymbol{x}_{n-1}\boldsymbol{x}_{n}, \boldsymbol{x}_{n+1})$	3-gram
			$\boldsymbol{x}_{n}$	$\boldsymbol{x}_{n+1}$	$\boldsymbol{x}_{n+2}$			d_6: $\zeta(\boldsymbol{x}_{n}, \boldsymbol{x}_{n+1}\boldsymbol{x}_{n+2})$	3-gram
				$\boldsymbol{x}_{n+1}$	$\boldsymbol{x}_{n+2}$	$\boldsymbol{x}_{n+3}$		d_7: $\zeta(\boldsymbol{x}_{n+1}, \boldsymbol{x}_{n+2}\boldsymbol{x}_{n+3})$	3-gram
$\boldsymbol{x}_{n-3}$	$\boldsymbol{x}_{n-2}$	$\boldsymbol{x}_{n-1}$	$\boldsymbol{x}_{n}$					d_8: $\zeta(\boldsymbol{x}_{n-3}\boldsymbol{x}_{n-2}\boldsymbol{x}_{n-1}, \boldsymbol{x}_n)$	4-gram
	$\boldsymbol{x}_{n-2}$	$\boldsymbol{x}_{n-1}$	$\boldsymbol{x}_{n}$	$\boldsymbol{x}_{n+1}$				d_9: $\zeta(\boldsymbol{x}_{n-2}\boldsymbol{x}_{n-1}\boldsymbol{x}_{n}, \boldsymbol{x}_{n+1})$	4-gram
			$\boldsymbol{x}_{n}$	$\boldsymbol{x}_{n+1}$	$\boldsymbol{x}_{n+2}$	$\boldsymbol{x}_{n+3}$		d_{10}: $\zeta(\boldsymbol{x}_{n}, \boldsymbol{x}_{n+1}\boldsymbol{x}_{n+2}\boldsymbol{x}_{n+3})$	4-gram
				$\boldsymbol{x}_{n+1}$	$\boldsymbol{x}_{n+2}$	$\boldsymbol{x}_{n+3}$	$\boldsymbol{x}_{n+4}$	d_{11}: $\zeta(\boldsymbol{x}_{n+1}, \boldsymbol{x}_{n+2}\boldsymbol{x}_{n+3}\boldsymbol{x}_{n+4})$	4-gram
$\boldsymbol{x}_{n-3}$	$\boldsymbol{x}_{n-2}$	$\boldsymbol{x}_{n-1}$	$\boldsymbol{x}_{n}$					d_{12}: $\zeta(\boldsymbol{x}_{n-3}\boldsymbol{x}_{n-2}, \boldsymbol{x}_{n-1}\boldsymbol{x}_n)$	4-gram
		$\boldsymbol{x}_{n-1}$	$\boldsymbol{x}_{n}$	$\boldsymbol{x}_{n+1}$	$\boldsymbol{x}_{n+2}$			d_{13}: $\zeta(\boldsymbol{x}_{n-1}\boldsymbol{x}_{n}, \boldsymbol{x}_{n+1}\boldsymbol{x}_{n+2})$	4-gram
				$\boldsymbol{x}_{n+1}$	$\boldsymbol{x}_{n+2}$	$\boldsymbol{x}_{n+3}$	$\boldsymbol{x}_{n+4}$	d_{14}: $\zeta(\boldsymbol{x}_{n+1}\boldsymbol{x}_{n+2}, \boldsymbol{x}_{n+3}\boldsymbol{x}_{n+4})$	4-gram

In the information-theoretic measures, three different functions ζ in Table 1 are applied in quantifying the fragment collocation, namely the mutual information (*MI*), likelihood ratio (*LR*) and the *t*-statistics (*TS*). Mutual information compares the probability of observing n_1-gram and n_2-gram together to the probability of observing them by chance. The mutual information of the n-gram between two POS fragments is defined as

$$MI(n_1\text{ - gram}, n_2\text{ - gram}) = \log \frac{P(n\text{ - gram})}{P(n_1\text{ - gram}) \times P(n_2\text{ - gram})} \tag{1}$$

For example, the measure of collocation using mutual information for d_{11} in Table 1 is $MI(\boldsymbol{x}_{n+1}, \boldsymbol{x}_{n+2}\boldsymbol{x}_{n+3}\boldsymbol{x}_{n+4})$ where the n-, n_1-, and n_2-gram is $\boldsymbol{x}_{n+1}\boldsymbol{x}_{n+2}\boldsymbol{x}_{n+3}\boldsymbol{x}_{n+4}$, $\boldsymbol{x}_{n+1}$ and $\boldsymbol{x}_{n+2}\boldsymbol{x}_{n+3}\boldsymbol{x}_{n+4}$ respectively. On the other hand, another similar approach of extracting collocation measure is conducted the likelihood ratio test. The likelihood ratio for a hypothesis is the ratio of the maximum value of the likelihood function over the subspace represented by the hypothesis to the maximum value of the likelihood function over the entire parameter space (Manning & Schütze, 1999). The log likelihood ratio is a formalization of independence which provides another good measure for the collocation between two POS fragments. While there is evidence that sparseness is an important problem for mutual information, the likelihood function seems to be the right candidate to complement with. In applying the likelihood ratio

LR to measure the POS collocation, the following two alternative hypotheses are examined.

H_0: $P(n_2\text{-gram} \mid n_1\text{-gram}) = p = P(\neg n_2\text{-gram} \mid n_1\text{-gram})$
H_1: $P(n_2\text{-gram} \mid n_1\text{-gram}) = p_1 \neq p_2 = P(\neg n_2\text{-gram} \mid n_1\text{-gram})$

If c_1, c_2, and c_{12} are the frequencies of n_1-gram, n_2-gram and the n-gram and N is the total number of POS tags found in our corpus, the log of the likelihood ratio *LR* is then calculated as follows:

$$\begin{aligned}\log LR = {} & \log L(c_{12}, c_1, p) + \log L(c_2 - c_{12}, N - c_1, p) - \log L(c_{12}, c_1, p_1) \\ & - \log L(c_2 - c_{12}, N - c_1, p_2)\end{aligned} \qquad (2)$$

$$\text{where } L(k,n,x) = x^k (1-x)^{n-k} \text{ and } p = \frac{c_2}{N} \quad p_1 = \frac{c_{12}}{c_1} \quad p_2 = \frac{c_2 - c_{12}}{N - c_1}$$

While mutual information (*MI*) and likelihood ratio (*LR*) measure the collocation of two POS fragments, the *t*-statistics (*TS*) tests the difference between the observed value and the hypothesized value scaled by the estimated variance. The *t*-test has been utilized by Church (1988) and Pedersen *et al.* (1996) to perform tests for association and bigram differences. In our modeling, the *t*-statistics measures how probable the collocation of POS fragments will occur. Unlike the modeling in *MI* and *LR*, we consider an *n*-gram of POS sequence as a *tri*-gram of an n_1-gram, n_2-gram and n_3-gram where $n_1 + n_2 + n_3 = n$. That is, assume an *n*-gram POS sequence Ω is concatenated from three fragments *x*, *y*, *z* with n_1-gram, n_2-gram and n_3-gram respectively, and let $\Omega = x \propto y \propto z$ which means the fragment *x* precedes the fragment *y* and so forth. Our *TS* score of fragment *y* in its neighbors *x* and *z* is defined as:

$$TS_{xz}(y) = \frac{\left(\dfrac{f(x \propto y)}{f(x)} - \dfrac{f(y \propto z)}{f(y)}\right)}{\sqrt{\dfrac{f(x \propto y)}{f^2(x)} + \dfrac{f(y \propto z)}{f^2(y)}}}, \text{ where } f \text{ is the frequency function} \qquad (3)$$

The *TS* score is a different collocation discovery approach which differentiates the co-occurrence of fragments in its neighborhood. The score is testing the differences of the collocation means between the fragment pairs (*x*, *y*) and (*y*, *z*). In the denominator of Eqn. (3), the variance of the difference of two collocations is the sum of their individual variances and the sample standard deviations are roughly approximated by their means. To determine whether there is a chunking point between $\boldsymbol{x}_n$ and $\boldsymbol{x}_{n+1}$, the following *TS* scores are calculated.

Table 2. Different *t*-statistics measure of collocation between adjacent POS fragments

ID	fragment *x*	fragment *y*	fragment *z*
TS_d_{21}	$\boldsymbol{x}_{n-1}$	$\boldsymbol{x}_n$	$\boldsymbol{x}_{n+1}$
TS_d_{22}	$\boldsymbol{x}_n$	$\boldsymbol{x}_{n+1}$	$\boldsymbol{x}_{n+2}$
TS_d_{23}	$\boldsymbol{x}_{n-2}\boldsymbol{x}_{n-1}$	$\boldsymbol{x}_n$	$\boldsymbol{x}_{n+1}\boldsymbol{x}_{n+2}$
TS_d_{24}	$\boldsymbol{x}_{n-1}\boldsymbol{x}_n$	$\boldsymbol{x}_{n+1}$	$\boldsymbol{x}_{n+2}\boldsymbol{x}_{n+3}$

Unlike other statistical language processing applications, neither predefined thresholds nor confidence levels will be introduced. Instead, all these information measures, between different fragments as shown in Tables 1 & 2, will be computed. They will be used as training data in the C5.0 machine learning program and different classifiers will be built as explained in below.

3 Building Classifiers Using C5.0

The C5.0 decision tree algorithm (Quinlan, 2000) is a well-known data mining technique that has been applied in various text processing applications such as finding associations in a collection of texts (Feldman & Hirsh, 1997) and mining online text (Knight, 1999). A decision tree is a tree structure where at each node a test on a particular attribute of the data is performed, and where the leaves corresponds to a particular class. The path from the root node to a particular leaf is a series of tests on the attributes that classifies the data into various class defined by the particular leaf. Using the information theory approach, an entropy criterion selects the most informative or discriminative feature through training. The best attribute is selected under a statistical property, called information gain, for the testing. It measures how well a given attribute differentiates the training examples according to their target classificatory scheme and to select the most suitable candidate attribute at each step while expanding the tree.

In our training in the C5.0, we follow the POS tagging system and the parse trees developed by the Chinese Knowledge Information Processing (CKIP) group at Academia Sinica. The Sinica Treebank has been developed and released since 2000. The Information-based Case Grammar (ICG) in the Treebank stipulates that each lexical item contains both syntactic and semantic features. The grammar indicates the way that lexical items in the sentences are related to each other.

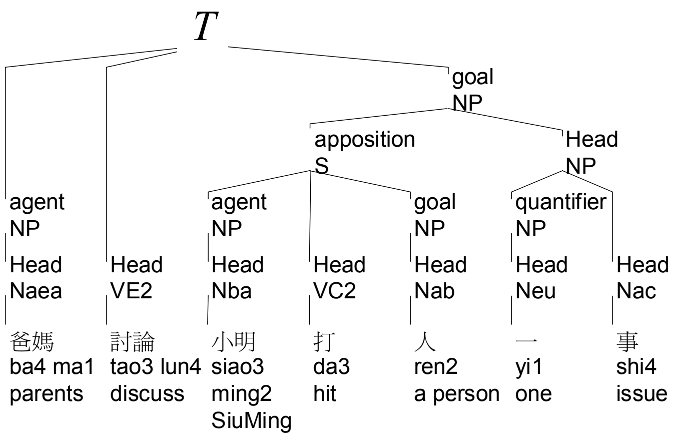


Fig. 1. An example parse tree in the latest Sinica Treebank

The grammatical constraints are expressed in terms of linear order of thematic roles hinted by syntactic and semantic clues. The tree structure has the advantage of maintaining phrase structure rules as well as the syntactic and semantic dependency relations. Thus, the ambiguities of syntactic structures can be resolved by preference checking in the semantic constraints or vice versa (Chen *et al.*, 2000). One of the parse trees found in the Treebank encoded in ICG is shown in the Figure 1. The tree is cast in terms of major POS as well as semantic categories. The latest version of Sinica Treebank (v.3.1), released in early 2006, contains about 65,000 parse trees with more than 300,000 words. The Treebank contains more than 45 different but major POS tags. For every parse tree *T* in the Treebank, a gap between two focus words is defined as a chunking point, marked as `%`, of *T* if the gap

(i) marks the beginning or the end of a phrase, or
(ii) indicates there is a direct terminal node in the parse tree *T*

The following shows the chunking points of the above parse tree *T*.

`% Naea % VE2 % Nba VC2 Nab % Neu Nac %` (P1)

The C5.0 machine learning algorithm has been employed to learn the chunking points from the CKIP Treebank. A sliding window of eight POS tags is defined as the context in the determination of the chunking points. Each instance in training data contains four POS tags both from the left and the right of the gap as well as the information-theoretic measures of collocation as discussed in Section 2. At the same time, we select a set of classifiers with different combination of collocation measures as shown in Table 3.

Table 3. Selected classifiers with different combination of collocation measures

Classifier ID	Attribute List
MI8-B1	MI_d_1, MI_d_2, MI_d_3, MI_d_4, MI_d_5, MI_d_6, MI_d_7
MI8-B2	MI_d_8, MI_d_9, MI_d_{10}, MI_d_{11}, MI_d_{12}, MI_d_{13}, MI_d_{14}
MI8-DB1	$(MI\_d_1 - MI\_d_2)$, $(MI\_d_2 - MI\_d_3)$, $(MI\_d_4 - MI\_d_6)$, $(MI\_d_5 - MI\_d_7)$
LR8-B1	LR_d_1, LR_d_2, LR_d_3, LR_d_4, LR_d_5, LR_d_6, LR_d_7
LR8-B2	LR_d_8, LR_d_9, LR_d_{10}, LR_d_{11}, LR_d_{12}, LR_d_{13}, LR_d_{14}
LR8-DB1	$(LR\_d_1 - LR\_d_2)$, $(LR\_d_2 - LR\_d_3)$, $(LR\_d_4 - LR\_d_6)$, $(LR\_d_5 - LR\_d_7)$
TS8-B1	TS_d_{21}, TS_d_{22}, TS_d_{23}, TS_d_{24}

For example, in studying the gap between the focus words with POS `Nba` and `VC2` in (P1), the attribute MI_d_9 in the classifier MI8-B2 represents the *MI* measure between the POS subsequences `Naea/VE2/Nba` and `VC2`, i.e. *MI*(`Naea/VE2/Nba`, `VC2`). We introduce two extra POS tags, namely `PosHH` and `PosTT`, to signal the head and the tail of parse trees respectively. That is, the attribute MI_d_8 measures the collocation of the POS subsequences `PosHH/Naea/VE2` and `Nba`. Similarly, the attribute $(MI\_d_1 - MI\_d_2)$ in the classifier MI8-DB1 measures the difference between the MI_d_1 and MI_d_2. This MI attribute quantifies the collocation disparity between the pairs (`VE2`, `Nba`) and (`Nba`, `VC2`). Similar classifiers are constructed using the likelihood ratio *LR* such as in LR8-B1, LR8-B2 and LR8-DB1.

While the eight POS tags in the sliding window are all declared as discrete attributes, all the collocation measures are continuous features having numeric values. In any instance of training data, the target attribute has only two possible values, that is, either `ChunkingPoint` or `non-ChunkingPoint`. Obviously, the value `non-ChunkingPoint` is assigned to the target attribute when considering the gap between the POS `Nba` and `VC2` in (P1) in the training of all classifiers.

4 Experimental Results

The algorithm has been implemented using Java language and run on an Intel® Core 2 Duo processor. All the *n*-grams are generated from the Academia Sinica CKIP corpus which contains more than five millions Chinese words with eight millions characters. This fully POS-tagged corpus contains 47 1-*gram*, 2,063 2-*gram*, 42,742 3-*gram* and 288,950 4-*gram* of different POS patterns. All the data are calculated and stored in the MySQL database. Furthermore, 65,000 parse trees in the Sinica Treebank are used for the training and testing of our sentence chunking. We assign 55,000 and 10,000 parse trees to the training and testing respectively. More than 271,000 training cases can be found in the 55,000 parse trees. Since the number of training cases is large, each classifier, as mentioned in Table 3, is trained with 20-fold cross-validation in order to increase the predictive accuracy. In the cross-validation, all the training cases are divided into 20 blocks of roughly the same size and class distribution. For each block in turn, rules are constructed from the cases in the blocks and tested on the cases in the others. In addition, to justify our approach to selecting the classifiers explained in Table 3, we have constructed some other control classifiers. For example, the POS6 classifier only considers the POS tags in a sliding window of size 6 without any other collocation measures. Similar to MI8-B2, the classifier MI4-B2 counts on a context of two POS tags both from the left and the right of the gap as well as all the collocation measures shown in Tables 3. The percentage errors of the classifiers found in the training phase of the C5.0 are shown in Table 4. As with other text analysis, the effectiveness of the system appears to be dictated by recall and precision parameters where recall (*R*) is a percentage of how many correct chunking points can be identified while precision (*P*) is the percentage of chunking points, tackled by the system, which are actually correct. A common parameter *F*-score is used as a single-figure measure of performance as in follows,

$$F = \frac{(\beta^2 + 1) \times P \times R}{\beta^2 \times P + R} \tag{4}$$

We set $\beta = 1$ to give no special preference to either recall or precision. The percentage error found in the training, the recall, precision and *F*-score in the testing of 10,000 parse trees are illustrated in Table 4. As shown in the table, it is worth noting that there is a large variance in performance appeared in our control classifiers. The classifiers based on a smaller window size have all under-performed. The classifiers with window size 4, i.e., MI4*, LR4* and TS4* are in the long run inadequate to provide effective and reliable rule sets for chunking. At the same time, the percentage error found in the classifiers which relied solely on POS without any information measures, such as POS*, are exceptionally high. This supports our assumption that a full chunking of sentences on the basis of the POS sequence alone is generally impossible, even though the classifier POS8 is the only one which narrows the gap between all POS* and our selected classifiers. Among all the selected classifiers shown in Table 3, the average results of the classifiers using *MI* are generally better than the others. Surprisingly, the classifiers originated from *4-gram*, that is MI8-B2 and LR8-B2, produce a reasonable high *F*-score. Similar situation occurs in our control classifiers. This provides another evidence to support the large window size approach in our selected classifiers. One possible explanation is that, due to the limited combination of POS tags, a stronger predictive power can only be obtained using the *4-gram* and larger window size, though there is less complete coverage of linguistic phenomena in our training parse trees.

Table 4. Percentage error found in the training, the recall, precision and *F*-score in the testing of 10,000 parse trees. All classifiers with window size 8 are our selected classifiers.

Classifier ID	*Training* Percentage error	*Testing* Recall	Precision	*F*-score
MI8-B2	5.40%	0.971	0.907	0.938
MI8-B1	5.50%	0.970	0.904	0.936
MI8-DB1	5.70%	0.970	0.903	0.935
LR8-B2	5.70%	0.970	0.902	0.935
LR8-B1	5.80%	0.968	0.901	0.933
LR8-DB1	5.90%	0.968	0.902	0.934
TS8-DB1	5.80%	0.968	0.902	0.934
POS4	15.1%	0.827	0.807	0.817
POS6	9.8%	0.903	0.894	0.898
POS8	7.4%	0.915	0.902	0.908
MI4-B2	7.60%	0.914	0.902	0.908
MI4-B1	10.30%	0.895	0.865	0.880
MI4-DB1	11.60%	0.876	0.868	0.872
LR4-B2	8.00%	0.911	0.907	0.909
LR4-B1	10.60%	0.882	0.872	0.877
LR4-DB1	11.50%	0.878	0.866	0.872
TS4-DB1	10.70%	0.896	0.873	0.884

In addition, we study the cooperative effects of the classifiers, in the sense that we combine the attributes from different groups to form another classifiers. As shown in Table 5 below, all 18 attributes originated from the MI function are assembled in the classifier MI8-18A. Similarly, all the attributes listed in Table 3 are brought together to build the MI8LR8TS8-40A. The results are compared with the benchmark classifier MI8-B2 as shown in Table 5. For example, in the classifier MI8LR8TS8-40A, 40 attributes are used in the training and more than 47,000 rules have been learnt. They are used to evaluate whether the focus gap is the chunking point or not. One might expect the more the attributes we use in the verdict, the higher the accuracy we have in the trial. Surprisingly, the best spin-off is the classifier MI8LR8-36A which involves only the MI* and LR*. At the same time, the overall performances of the classifiers shown in Table 5 are all superior to the results shown in Table 4. In other words, the cooperative effects are significant, further efforts are needed to verify it is the best and optimal solution, though. In this paper, we have articulated one of the plausible ways to combine different, large and heterogeneous sets of attributes in the chunking point detection using the C5.0. While the approach has been implemented successfully using the Academia Sinica Corpus and Treebank as illustrated in our evaluation, the idea delineated certainly does not limit or tailor-made for any particular language. Only a minor modification is needed to apply the technique to other languages.

Table 5. Percentage error found in the training, the recall, precision and *F*-score in the testing of 10,000 parse trees in studying the cooperative effects

Classifier ID	# of Rules Generated	*Training* Percentage error	*Testing* Recall	Precision	*F*-score
MI8-B2 (Benchmark)	45,387	5.40%	0.971	0.907	0.938
MI8-18A	46,925	5.20%	0.972	0.911	0.941
LR8-18A	46,200	5.30%	0.970	0.909	0.938
MI8LR8-36A	48,182	5.00%	0.973	0.915	**0.943**
MI8LR8TS8-40A	47,370	5.00%	0.971	0.914	0.942

5 Conclusions

Even if the claim *language always repeats itself* is too strong, we have to accept that there are many similar patterns in all languages. While we try to capture the regularity of duplication and incorporate them into doctrine from a linguistic perspective, we believe the trace can also be found in our corpus. In this paper, we have illustrated a shallow technique in which sentences are segmented into chunks based on various POS collocation measures. The chunking in its pure form has been developed as a method of classification. Examples found in a Treebank are used to predict the most similar instances. This shallow technique is inspired by research into bio-molecular sequence analysis, which has observed that the vast majority of extant bio-molecular sequences are the result of a continuous series of genetic duplications with subsequent modifications (Doolittle, 1981). This *duplication-with-modification* is the central paradigm of bio-molecular sequence evolution, wherein new sequences and/or new biological functions are fashioned from earlier ones. It is characteristic of biological systems that objects have a certain form that has arisen by evolution from related objects of similar but not identical form. A sentence segment, as a bio-molecular subsequence in a biological system, encodes and reflects the most native, but complicated, language structure that has been described by our linguists. *Redundancy* with high replication is a built-in characteristic of language sequence. It is not surprising that so many new sequences, which are under *duplication-with-modification*, resemble already known sentence segment or various phrases. With the help of the decision tree algorithm, the result demonstrated in this paper shows this mapping is tractable, though partly heuristic. While this paper does not claim to reveal all aspects of the mapping, we suggest an alternate, but plausible, way to parse the real-world sentences.

Acknowledgments. The work described in this paper was partially supported by the grants from the Research Grants Council of the Hong Kong Special Administrative Region, China (Project Nos. CUHK4438/04H and CUHK4706/05H).

References

Abney, S.: Parsing by chunks. In: Berwick, R., Abney, S., Tenny, C. (eds.) Principle-Based Parsing, Kluwer Academic Publishers, Dordrecht (1991)

Argamon-Engelson, S., Dagan, I., Krymolowski, Y.: A memory-based approach to learning shallow natural language patterns. Journal of Experimental Theoretical Artificial Intelligence 11, 369–390 (1999)

Chen, F.-Y., Tsai, P.-F., Chen, K.-J., Huang, C.-R.: Sinica Treebank. Computational Linguistics and Chinese Language Processing 4(2), 87–103 (2000)

Church, K.: A stochastic parts program and noun phrase parser for unrestricted text. In: Proceedings of Second Conference on Applied Natural Language Processing, Texas (1988)

Doolittle, R.F.: Similar amino acid sequences: Chance or common ancestry? Science 214, 149–159 (1981)

Feldman, R., Hirsh, H.: Finding associations in collections of text. In: Michalski, R.S., Bratko, I., Kubat, M. (eds.) Machine Learning and Data Mining: Methods and Applications, pp. 223–240. John Wiley, Chichester (1997)

Joshi, A.: The relevance of tree adjoining grammar to generation. In: Kempen, G. (ed.) Natural language generation: new results in Artificial Intelligence, psychology and linguistics, pp. 233–252. Kluwer Academic Publishers, Dordrecht (1987)

Kaplan, R.M., Bresnan, J.: Lexical-Functional Grammar: A formal system for grammatical representation. In: Bresnan, J. (ed.) The Mental Representation of Grammatical Relations, pp. 173–281. The MIT Press, Cambridge (1982)

Knight, K.: Mining online text. Communications of the ACM 42(11), 58–61 (1999)

Magerman, D.M., Marcus, M.P.: Parsing a natural language using mutual information statistics. In: AAAI 1990. Proceedings of Eighth National Conference on Artificial Intelligence, pp. 984–989 (1990)

Manning, C.D., Schutze, H.: Foundations of Statistical Natural Language Processing. MIT Press, Cambridge (1999)

Pedersen, T., Kayaalp, M., Bruce, R.: Significant lexical relationships. In: Proceedings of the 13th National Conference on Artificial Intelligence, Portland (1996)

Pollard, C., Sag, I.: Head-Driven Phrase Structure Grammar. University of Chicago, Press, Chicago, CSLI Publications, Stanford (1994)

Quinlan, R.: (2000) http://www.rulequest.com/

Ramshaw, L.A., Marcus, M.P.: Text chunking using transformation-based learning. In: Proceedings of the Third Workshop on Very Large Corpora, pp. 82–94 (1995)

Tjong, K.S.E., Veenstra, J.: Representing text chunks. In: EACL 1999. Proceedings of Association for Computational Linguistics, Bergen (1999)

The Semantic Representation of Temporal Expressions in Text

Robert Dale and Paweł Mazur

Centre for Language Technology
Macquarie University
North Ryde, NSW 2109
`rdale,mpawel@ics.mq.edu.edu`

Abstract. Temporal expressions—references to points in time or periods of time—are widespread in text, and their proper interpretation is essential for any natural language processing task that requires the extraction of temporal information. Work on the interpretation of temporal expressions in text has generally been pursued in one of two paradigms: the formal semantics approach, where an attempt is made to provide a well-grounded theoretical basis for the interpretation of these expressions, and the more pragmatically-focused approach represented by the development of the TIMEX2 standard, with its origins in work in information extraction. The former emphasises formal elegance and consistency; the latter emphasises broad coverage for practical applications. In this paper, we report on the development of a framework that attempts to integrate insights from both perspectives, with the aim of achieving broad coverage of the domain in a well-grounded manner from a formal perspective. We focus in particular on the development of a compact notation for representing the semantics of underspecified temporal expressions that enables the component-level evaluation of systems.

1 Introduction

Obtaining a precise semantic representation for utterances that contain references to time is interesting both from a theoretical point of view, as there are many complex phenomena to be addressed, and for purely practical applications such as information extraction, question answering, or the ordering of events on a timeline. In the literature, work on the interpretation of temporal expressions comes from two directions. On the one hand, work in formal semantics (see, for example, [1]) aims to provide a formally well-grounded approach to the representation of the semantics of these expressions, but such approaches are difficult to scale up to the broad coverage required for practical applications. On the other hand, work that has its roots in information extraction, while it emphasizes broad coverage, often results in the use of ad hoc representations. The most developed work in this direction is focused around the TimeML markup language (described, for example, in [2] and in the collection edited by Mani et al. [3]). Some work attempts to bring these two traditions together: notable in this respect is Schilder's [4] work on temporal expressions in German newswire text, and Hobbs and Pan's [5] work on axiomatisation in terms of OWL-Time.

Our contribution towards a truly broad coverage yet semantically well-founded approach is to recognize a principled distinction between (a) the interpretation of the semantics of a temporal expression devoid of its context of use, and (b) the fuller interpretation of that expression when the context is taken into account. The first of these, which we refer to as the **local semantics** of a temporal expression, should be derivable compositionally from the components of the expression; determining the value of the second, which we refer to as the **global semantics** of the expression, may require arbitrary inference and reasoning. Such a distinction is implicit in other accounts: Schilder's [4] use of lambda expressions allows representation of partially specified temporal entities, and the temporary variables that Negri and Marseglia [6] construct during the interpretation of a given temporal expression capture something of the same notion.

Our proposal is to reify this level of intermediate representation based on a formalization in terms of recursive attribute–value matrices, as traditionally used in other areas of natural language processing (see, for example, [7]). This has two distinct advantages: (a) it provides a convenient representation of underspecification, and (b) it leads naturally to a compositional approach to the construction of the semantics of temporal expressions via unification.

We also provide a compact encoding of this representation that is an extension of the existing TIMEX2 representation for temporal expressions. This brings the advantages that (a) existing tools and machinery for evaluation can be used to determine how well a given implementation derives these local semantic values; and (b) performance in the determination of local semantics and global semantics can be tested independently. To ensure breadth of coverage, we initially developed our representation on the basis of all 256 examples of temporal expressions provided in the TIMEX2 guidelines [8]; this has been subsequently refined on the basis of examination of the evaluation data sets provided under the Automatic Context Extraction (ACE) program.[1]

The rest of this paper is structured as follows. In Section 2, we describe the architecture of DANTE, a system which embodies our approach to the interpretation of temporal expressions; in particular, we focus on the architecture we employ, and on the particular levels of representation that it makes use of. In Section 3, we argue for an intermediate representational level that captures the semantics of temporal expressions independent of the context of their interpretation, and introduce the idea of using recursive attribute–value matrices to represent the semantics of temporal expressions. In Section 4, we provide an encoding of these attribute–value matrices in a compact string-based representation that is effectively an extension of the ISO-based date–time format representations used in the TIMEX2 standard, thus enabling easy evaluation of system performance using existing tools. In Section 5, we report on the performance results of DANTE using the ACE 2007 evaluation data. Finally, in Section 6 we draw some conclusions and point to future work.

[1] See `http://www.nist.gov/speech/tests/ace`

2 The DANTE System

2.1 Processing Steps

In our work, our goal is very close to that for which the TIMEX2 standard was developed: we want to annotate each temporal expression in a document with an indication of its interpretation, in the form of an extended ISO-format date and time string, normalised to some time zone. So, for example, suppose we have the following italicised temporal expression in an email message that was sent from Sydney on Friday 30th November 2007:

(1) We will be able to present this at the meeting on *Monday at 2pm*.

In our application, this temporal expression should be marked up as follows:

(2) `<TIMEX2 VAL="2007-12-03T03:00GMT">Monday at 2pm</TIMEX2>`

We have to do three things to achieve the desired result:

– First, we have to detect the extent of the temporal expression in the text. We refer to this process as **temporal expression recognition**.
– Then, we have to use information from the document context to turn the recognized expression into a fully specified date and time. We refer to this as **temporal expression interpretation**.
– Finally, we have to normalise this fully specified date and time to a predefined time zone, which in the case of the present example is Greenwich Mean Time. We refer to this as **temporal expression normalisation**.[2]

We observe that, at the time the extent of a temporal expression within a text is determined, it is already possible to derive a semantic representation of that expression irrespective of the wider context required for a full interpretation: for example, having recognized an occurrence of the string *Friday* in a text, we already know that this is a reference to a specific day of the week. It is likely that most existing systems for the interpretation of temporal expressions make use of such a level of representation implicitly. Schilder's [4] approach captures the semantics here in terms of a lambda expression like $\lambda x \text{Friday}(x)$; Negri and Marseglia [6] capture information at this stage of processing via a collection of temporary attributes.

In our system, each of the three steps above corresponds to a distinct processing component in the DANTE system architecture. These components communicate in terms of a number of distinct representations, which we now describe.

Local Semantics: We use this term to refer to a level of representation that corresponds to the semantic content that is derivable directly from the text

[2] This third step is not required by the TIMEX guidelines, but is an additional requirement in the context of our particular application. This also means that our use of the term 'normalisation' here is not consistent with the standard usage in the TIMEX context; however, we would argue that our distinction between interpretation and normalisation describes more accurately the nature of the processes involved here.

representation; in the case of temporal expressions that are arguments to prepositions, this includes the interpretation of the preposition. Such representations are often incomplete, in that they do not in themselves denote a specific point or period on the time line; however, usually they do partially specify points or periods, and constrain the further interpretation of the string.

In-Document Semantics: This refers to the fully explicit interpretation of the text string, to the extent that this can be determined from the document itself, in conjunction with any metadata associated with the document (in particular, the time and date of publication). This level of representation corresponds to the information encoded in the attributes of the TIMEX2 tag as defined in the TIMEX guidelines.

Global Semantics: The TIMEX guidelines do not have anything to say beyond the representation just described. In our application, however, we are also required to normalise all temporal expressions to a specific time zone. This requires that some further temporal arithmetic be applied to the semantics of the found expressions. To calculate this, we have to determine the difference between the time zone of the document containing the temporal reference and the target time zone, here Greenwich Mean Time.

3 Representing Temporal Expressions

3.1 Temporal Entities

As is conventional in this area of research, we view the temporal world as consisting of two basic types of entities, these being **points in time** and **durations**; each of these has an internal hierarchical structure. In the present paper we focus on the representation of points in time; the representation of durations is a natural extension of this representation. Figure 1 shows the representation of a reference to a specific point in time; in the ISO date and time format used in the TIMEX2 standard, this would be written as follows:

(3) `2006-05-13T15:00:00Z`

Each atomic feature in the attribute–value structure thus corresponds to a specific position in the ISO format date–time string.

3.2 Underspecification

Of course, very few temporal expressions in text are so fully specified. The attribute–value matrix representation makes it easy to represent the content of underspecified temporal expressions. For example, the content of the temporal expression *Thursday* in a sentence like *We met on Thursday* can be expressed as:

$$\begin{bmatrix} \textit{point} & \\ \text{TIMEANDDATE} & \begin{bmatrix} \text{DATE} & \begin{bmatrix} \text{DAY} & \begin{bmatrix} \text{DAYNAME} & D4 \end{bmatrix} \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

$$\begin{bmatrix} \textit{point} & \\ \text{TIMEANDDATE} & \begin{bmatrix} \text{TIME} & \begin{bmatrix}\text{HOUR} & \textit{15}\\ \text{MINS} & \textit{00}\\ \text{SECS} & \textit{00}\end{bmatrix} \\ \text{DATE} & \begin{bmatrix} \text{DAY} & \begin{bmatrix}\text{DAYNAME} & \textit{D4}\\ \text{DAYNUM} & \textit{13}\end{bmatrix} \\ \text{MONTH} & \textit{5}\\ \text{YEAR} & \textit{2006}\end{bmatrix}\end{bmatrix} \\ \text{ZONE} & Z\end{bmatrix}$$

Fig. 1. The semantics of the expression *3pm Thursday 13th May 2006 GMT*

On the other hand, a reference to *13th May* in a sentence like *We will meet on 13th May* has this representation:

$$\begin{bmatrix} \textit{point} & \\ \text{TIMEANDDATE} & \begin{bmatrix} \text{DATE} & \begin{bmatrix} \text{DAY} & \begin{bmatrix}\text{DAYNUM} & \textit{13}\end{bmatrix} \\ \text{MONTH} & \textit{05}\end{bmatrix}\end{bmatrix}\end{bmatrix}$$

In the cases just described, the semantic representation corresponds to the entire temporal noun phrase in each case. The same form of representation is easy to use in a compositional semantic framework: each constituent in a larger temporal expression provides a structure that can be unified with the structures corresponding to the other constituents of the expression to provide a semantics for the expression as a whole. The values of the atomic elements in such an expression come from the lexicon; multiword sequences that are best considered atomic (such as, for example, idioms) can also be assigned semantic representations in the same way. The value of a composite structure is produced by unifying the values of its constituents. Unifying the two structures above, for example, gives us the following representation for *Thursday 13th May*:

$$\begin{bmatrix} \textit{point} & \\ \text{TIMEANDDATE} & \begin{bmatrix} \text{DATE} & \begin{bmatrix} \text{DAY} & \begin{bmatrix}\text{DAYNAME} & \textit{D4}\\ \text{DAYNUM} & \textit{13}\end{bmatrix} \\ \text{MONTH} & \textit{05}\end{bmatrix}\end{bmatrix}\end{bmatrix}$$

So, these structures provide a convenient representation for what we have referred to above as the local semantics of a temporal expression, and correspond to the output of the recognition stage of our processing architecture.

3.3 Interpretation

We can now define the task of interpretation in terms of the content of these structures. We assume a **granularity ordering** over what we might think of as the **defining attributes** in a temporal representation:

(4) year > month > daynum > hour > minute > second

These are, of course, precisely the elements that are represented explicitly in an ISO date–time expression.

Interpretation of a partially specified temporal expression then requires ensuring that there is a value for every defining attribute that is of greater granularity than the smallest granularity present in the partially specified representation. We refer to this as the **granularity rule** in interpretation.

In the case of the example in the previous section, the granularity rule tells us that in order to compute the full semantic value of the expression we have to determine a value for YEAR, but not for HOUR, MINS or SECS. This interpretation process may require various forms of reasoning and inference, and is qualitatively different from the computation of the local semantics.

In the context of our application, a third stage, the normalisation process, then requires taking the further step of adding a ZONE attribute with a specific value, and translating the rest of the construction into this time zone if it represents a time in another time zone.

4 A Compact Encoding

The structures described in the previous section are relatively unwieldy in comparison to the simple string structures used as values in the TIMEX standard. To enable easy evaluation of a system's ability to construct these intermediate semantic representations, we would like to use a representation that is immediately usable by existing evaluation tools. To achieve this goal, we define a number of extensions to the standard TIMEX2 string representation for values of the VAL attribute; these extensions allow us to capture the range of distinctions we need. To save space, we also use these representations here in preference to using attribute–value matrices to show the coverage of the annotation scheme that results.

4.1 Partially Specified Dates and Times

As noted above, many references to dates or times are not fully specified in a text, with the result that some parts will have to be computed from the context during the interpretation stage. Typical examples are as follows:

(5) a. We'll see you in *November*.
b. I expect to see you at *half past eight*.

Table 1. Underspecified Dates and Times

#	String	Representation	#	String	Representation
1	9 pm	xxxx-xx-xxT21	6	the nineteenth	xxxx-xx-19
2	11:59 pm	xxxx-xx-xxT23:59	7	January 3	xxxx-01-03
3	eleven in the morning	xxxx-xx-xxT11:00	8	November	xxxx-11
4	ten minutes to 3	xxxx-xx-xxt02:50	9	summer	xxxx-SU
5	15 minutes after the hour	xxxx-xx-xxtxx:15	10	the '60s	xx6

In the recursive attribute–value notation introduced above, the missing information in each case corresponds to those features that are absent in the structure as determined by the granularity rule introduced in Section 3.3.

In our string-based notation, we use lowercase xs to indicate those elements for which a value needs to be found, but which are not available at the time the local semantics are computed; and we capture the granularity requirement by omitting from the string representation those elements that do not require a value.[3] Table 1 provides a range of examples that demonstrate various forms of underspecification.

A lowercase x thus corresponds to a variable. By analogy, we also use a lowercase t instead of the normal ISO date–time separator of T to indicate that the time needs further specification: consider the fourth and fifth examples in Table 1, where it is unclear whether the time specified is a.m. or p.m.

For partially-specified dates and times, the string-based encoding thus both captures the local semantic content of the temporal expression, and provides a specification of what information the interpretation process has to add. The interpretation process makes use of a notion of **temporal focus**, the temporal point of reference that generally shifts through the course of a text. If the temporal focus is encoded in the same form of representation, then producing the final interpretation is often a simple process of merging the two structures, with the values already specified in the intermediate representation taking precedence over those in the representation of the temporal focus.

4.2 Relative Dates and Times

A relative date or time reference is one that requires a calendar arithmetic operation to be carried out with respect to some temporal focus in the text. Typical examples are as follows:

(6) a. We'll see him *tomorrow*.
    b. We saw him *last year*.
    c. We'll see him *next Thursday*.
    d. We saw him *last November*.

[3] Note that this does not mean the same thing as the use of an uppercase X in the TIMEX2 guidelines: there, an uppercase X means that no value can be determined. In our approach, if no value can be found for a variable element during the interpretation process, then the corresponding x will be replaced by an X.

Table 2. Relative dates and times in ISO-like format

#	String	Representation	#	String	Representation
1	today	+0000-00-00	6	sixty seconds later	+0000-00-00T+00:00:60
2	tomorrow	+0000-00-01	7	five minutes ago	+0000-00-00T−00:05
3	yesterday	−0000-00-01	8	in six hours time	+0000-00-00T+06:00
4	last month	−0000-01	9	at 6 a.m. today	+0000-00-00T06:00
5	three years ago	−0003	10	last night	−0000-00-01TNI

To handle relative references, we extend the ISO format with a preceding '+' or '−' to indicate the direction from the current temporal focus. Some examples of dates and times are provided in Table 2. Note the both the date and time elements in a relative reference can be independently either absolute or relative: compare the representations for *in six hours time* and *at 6am today*.

This representation leads to a very intuitive coordinate-based arithmetic for computing the final semantic interpretation of a given expression: the interpreter merely adds the temporal focus and the intermediate value element-by-element from the smallest unit upwards, using carry arithmetic where appropriate.

5 Evaluation

The primary focus of the work described here has been to develop a representation for local semantics that covers as wide a range of linguistic phenomena as possible; for this reason, the representation has been developed using the TIMEX2 guidelines as a target data set, since the examples presented there are intended to cover the broad range of phenomena that a human annotator is likely to encounter.

However, it is also important, of course, that the representational mechanism then be tested for its performance in a real task evaluation. To this end, we subsequently fine-tuned the representations on the basis of the ACE 2005 development data, and in January 2007, DANTE participated in the ACE Time Expression Recognition and Normalization (TERN) task.

In the ACE evaluations a correctly recognized time expression is one which has a strictly accurate extent and correct values for all the TIMEX2 attributes. An annotation generated by the system is classified as matched with an annotation from the gold standard if there is minimum 30% text span overlap between them

The ACE 2007 evaluation data included 2028 time expressions to be recognized and interpreted. Across all domains we currently achieve 54.7, 57.6 and 56.1 for precision, recall and F-measure, respectively, for correct recognition of temporal expressions. After applying weights to particular elements which are subject to evaluation,[4] the scores are 69.7, 69.2 and 69.4 respectively; the overall

[4] In the ACE 2007 TERN evaluations the weights were as follows: 1.0 for type VAL, 0.5 for ANCHOR_VAL, 0.25 for ANCHOR_DIR, 0.1 for MOD, 0.1 for SET, 0.1 for extent (where there is at least a 30% overlap between matched elements; otherwise elements are not mapped at all). The cost for spurious TIMEX2 mentions was −0.75.

Table 3. The results of evaluating DANTE on the ACE 2007 evaluation data set

Domain	Entities in gold standard	Spurious	Missing	Error	Precision	Recall	F-score	ACE Value
Broadcast Conversation	142	33	29	43	47.9	49.3	48.6	46.5
Broadcast News	322	103	38	69	55.6	66.8	60.6	55.2
Newswire	894	128	110	273	56.0	57.2	56.6	58.8
Telephone Conversation	70	23	11	25	41.5	48.6	44.7	51.4
Usenet Newsgroups	167	20	22	43	61.8	61.1	61.4	65.3
Weblogs	433	68	58	139	53.3	54.5	53.9	57.3
Total	2028	375	268	592	54.7	57.6	56.1	57.2

ACE TERN score for DANTE is 57.2. These results indicate that DANTE's performance is already very close to state-of-the-art systems: the best performing system in the ACE evaluations received an overall score of 61.6.

Table 3 shows the results of the system on the ACE 2007 data broken down by text domain. An analysis of the errors indicates that the representation remains robust when faced with real data; the bulk of our errors are due to limitations in our grammatical coverage (often due to the long tail of low-frequency lexical items and phrases used in temporal expressions) and deficiencies in our focus tracking mechanism.

6 Conclusions

In this paper, we have argued that, in the context of interpreting temporal expressions, there is value in identifying a level of semantic representation that corresponds to the meaning of these expressions outside of any particular document context. Many existing systems appear to make use of such representations at least implicitly. However, we have proposed that this level of representation be made explicit; and by providing an encoding of this level of representation that is an extension of the existing TIMEX2 annotations in terms of element attributes and their values, we make it possible to assess the performance of systems with respect to intermediate values, final values, or both, using standard evaluation tools.

We have developed the representation described here on the basis of the set of 265 examples provided in the TIMEX2 guidelines [8], and tested the performance of DANTE, a system using the representation, on the ACE 2005 and 2007 gold standard data sets. We achieve results that are broadly comparable with the state of the art.

DANTE's temporal expression recognizer is implemented using GATE's JAPE grammar formalism [9]; about 80 macros and 250 rules are used in the current implementation. Recognition is thus by means of finite-state patterns; however, an examination of those cases where we fail to determine the correct extents for temporal expressions indicates that we suffer when the expressions in question are quite complex syntactically, as in *four days after Americans first penetrated the Baghdad outskirts*. Here, DANTE only recognises the string *four days* as a temporal expression and fails to detect the attached prepositional phrase. A major focus for future work is to integrate a syntax-based mechanism for detecting the extents of arbitrarily complex temporal expressions. This will also allow us to more rigorously test the compositional properties of our intermediate representation.

Acknowledgements

We acknowledge the support of the Defence Science and Technology Organisation in carrying out the work described here.

References

1. Pratt, I., Francez, N.: Temporal prepositions and temporal generalized quantifiers. Linguistics and Philosophy 24, 187–222 (2001)
2. Pustejovsky, J., Castaño, J., Ingria, R., Saur, R.G., Setzer, A., Katz, G.: TimeML: Robust Specification of Event and Temporal Expressions in Text. In: IWCS-5. Fifth International Workshop on Computational Semantics (2003)
3. Mani, I., Pustejovsky, J., Gaizauskas, R. (eds.): The Language of Time. Oxford Univ. Press, Oxford (2005)
4. Schilder, F.: Extracting meaning from temporal nouns and temporal prepositions. ACM Transactions on Asian Language Information Processing 3(1), 33–50 (2004)
5. Hobbs, J.R., Pan, F.: An ontology of time for the semantic web. ACM Transactions on Asian Language Information Processing 3(1), 66–85 (2004)
6. Negri, M., Marseglia, L.: Recognition and normalization of time expressions: ITC-IRST at TERN 2004. Technical Report WP3.7, Information Society Technologies (2005)
7. Shieber, S.M.: An Introduction to Unification-Based Approaches to Grammar. Center for the Study of Language and Information (1986)
8. Ferro, L., Gerber, L., Mani, I., Sundheim, B., Wilson, G.: TIDES 2005 Standard for the Annotation of Temporal Expressions. Technical report, MITRE (2005)
9. Cunningham, H., Maynard, D., Bontcheva, K., Tablan, V.: GATE: A framework and graphical development environment for robust NLP tools and applications. In: Proceedings of the 40th Anniversary Meeting of the ACL (2002)

Effectiveness of Methods for Syntactic and Semantic Recognition of Numeral Strings: Tradeoffs Between Number of Features and Length of Word N-Grams

Kyongho Min[1], William H. Wilson[1], and Byeong-Ho Kang[2]

[1] School of Computer Science and Engineering
University of New South Wales, Sydney, Australia
{min, billw}@cse.unsw.edu.au
[2] School of Computing
University of Tasmania, Hobart, Australia
bhkang@utas.edu.au

Abstract. This paper describes and compares the use of methods based on N-grams (specifically trigrams and pentagrams), together with five features, to recognise the syntactic and semantic categories of numeral strings representing money, number, date, etc., in texts. The system employs three interpretation processes: word N-grams construction with a tokeniser; rule-based processing of numeral strings; and N-gram-based classification. We extracted numeral strings from 1,111 online newspaper articles. For numeral strings interpretation, we chose 112 (10%) of 1,111 articles to provide unseen test data (1,278 numeral strings), and used the remaining 999 articles to provide 11,525 numeral strings for use in extracting N-gram-based constraints to disambiguate meanings of the numeral strings. The word trigrams method resulted in 83.8% precision, 81.2% recall ratio, and 82.5% in F-measurement ratio. The word pentagrams method resulted in 86.6% precision, 82.9% recall ratio, and 84.7% in F-measurement ratio.

Keywords: numeral strings, N-grams, named entity recognition, natural language processing.

1 Introduction

Numbers (e.g. “801 voters”) and Alpha-numeric strings (e.g. “9:30am”) with or without Roman numerals (e.g. “ENIAC-II”) and with or without special symbols (e.g. “$” in “$2.5 million”) are essential components of written or spoken human languages. Without understanding numeral strings, it is often hard to understand the sentences and texts that they appear in. Dale [6] discussed types of numerical expressions and their corresponding meanings in his 1997 paper on tokenisation of text: numeral strings are on average much more ambiguous than words. For example, the number “2” could mean AGE, RANK, ORDER, Section number, QUANT, DAY, NUMBER, part of a Named Entity, and etc.

Current NLP systems treat such strings as either a numeral (e.g. "801 voters") or as a named entity (e.g. MONEY for "$2.5 million"). However, ambiguity in their interpretation may arise without semantic/contextual information: for example, "28" in the phrase "he turns 28 today" can on the surface be interpreted as any of the following - (a) as NUMBER; (b) as DAY of a date expression; or (c) as AGE at the lexical meaning level. Such numeral strings are called *separate numeral strings* in this paper. There are also *affixed numeral strings* (e.g. "*10m* yacht"); these have meaningful semantic units attached (e.g. "m" in "10m") that reduce interpretation ambiguity. The natural interpretation in this case is LENGTH, but note that "10m" could signify a model number, or, if embedded in "$10m yacht", MONEY. In another context, "m" could signify "minute", so that "10m" would be TIME.

A special method related to N-grams, called s-grams (Skipped grams), was studied in [8] in the context of approximate string matching in an Information Retrieval area for European languages. The complex use of numeral strings in biomedical texts for protein name identification has been studied [19] [21]. In the domain of text classification, meanings of numeral strings (e.g. Time, Date, Money, Phone number etc.) based on bigrams were used as contextual features in call-for-tender documents [16]. Agrawal and Srikant discussed a keyword searching method for {attribute, numeric_value} pairs in the database (e.g. a query like "address set-up speed 20 ns power 500mW CMOS") [1]. Even though their task is not directly related to our task, this is an instance of the importance of meanings of numerals in information retrieval areas.

In the information extraction and named entity recognition areas [3], [5], basic semantic categories (e.g. PERSON, ORGANISATION, DATE, TIME, and MONEY) have been recognised. FACILE [3] in MUC used a rule-based named entity recognition system incorporating a chart-parsing technique with semantic categories. Semantic categories and semantic tags were used for a Chinese classification system in [12], [22] and for Japanese documents in [2], [20]. The ICE-GB grammar [15] used *cardinal*, *ordinal*, *fraction*, *hyphenated*, or *multiplier,* with two number features - singular and plural – for numeral categories. Zhou and Su [23] employed an HMM-based chunk tagger to recognise numerical quantities with surface and semantic features like FourDigitNum (e.g. 1990) as a year form, and SuffixTime (e.g. a.m.) as a time suffix (see also [18] for time phrases in weather forecasts). Polanyi and van den Berg [17] studied anaphoric resolution of quantifiers and cardinals and employed a quantifier logic framework. Min et. al. [13], [14] studied a manually generated rule-based numeral-string interpretation method and also a system based on automatically-generated bigrams constraints.

We have now implemented ENUMS (English NUMeral understanding System) with trigrams and also with 5-grams (e.g. 3L1R1 – a trigram consisting of one token to the left of the numeral string and one to the right, and 5L2R2) using constraints based on five features: POS, syntactic features, prefix, suffix, and special information. The ENUMS system is composed of a tokeniser, word N-grams retrieval, constraints retrieval, and numeral strings recognition and disambiguation modules in detail, with four sources of knowledge bases: dictionary, morphological rules, syntactic rules of a numeral strings, and word N-gram constraints. The understanding of numeral strings depends on their type: affixed numeral strings (e.g. "24km/h") require rule-based processing (i.e. syntactic rules for affixed numeral strings) after deep tokenisation of the numeral string (e.g. "24km/h" $\rightarrow$ "24" + "km" + "/" + "h"). In this paper, we will focus on the tradeoffs between size of N-grams (e.g. 3 in 3L1R1 vs 5 in 5L2R2) and number of features used for constraint retrieval (in the range 1 to 5).

In the next section, the architecture of the ENUMS system will be described. In section 3, we will describe the ENUMS recognition algorithm. Section 4 will describe experimental results obtained with trigrams and pentagrams variants of ENUMS, and discussion and conclusions follow.

2 Architecture of the ENUMS System

An ENUMS system is organised into modules using different knowledge sources: a dictionary, morphological rules for words, syntactic rules for numeral strings, word-N-gram-based constraints using a number of features (e.g. POS, syntactic features, prefix, suffix). ENUMS is implemented in Common Lisp with its IDE. In the next section, we will focus on the recognition of numeral strings using ENUMS.

We discuss categories and rules used in the ENUMS system briefly in this section. Table 1 shows some examples of numeral strings, and syntactic and semantic categories used in the ENUMS system. We used 40 numeral string categories – some are of a semantic nature, such as MONEY and DATE, and some are basically syntactic, such as NUMBER, FLOATNUMBER, for numeral strings with no more specific category. The category FMNUMBER (ForMatted Number) signifies numbers that include commas every 3 digits to the left of the unit digit for ease of reading, as in "*12,000* peacekeepers."

Table 1. Some categories and examples of them

Category	Example	Category	Example
Day	"August *11*"	Age	"mature *20-year-old* contender
Floatnumber	"support at *26.8* per cent"	Daytime	"between *9:30am* and *2am*"
Number	"*8000* of the Asian plants"	Length	"a *10m* yacht"
Quant	"survey of *801* voters"	Money	"spend *US$1.4* billion"
Year	"by September *2026*"	Scores	"a narrow *3-6* away loss to Otago"

The ENUMS dictionary includes meaningful objects useful for interpretation, such as symbolic tokens (e.g. "(", ")"), lexical words (e.g. "year"), and units (e.g. "km", "m", etc.). For example, the lexical information for "m" is ("m" ((:POS TU :SEM MINUTE) (:POS LU :SEM METER) (:POS NU :SEM MILLION))) where TU stands for Time Unit, LU Length Unit, NU Number Unit, and SEM SEMantics.

Fig. 1 describes the architecture of the ENUMS system. There are four modules with four major types of knowledge bases. The Tokeniser module reads input documents and analyses each document into sentences and tokens based on a word (i.e. a string delimited on the left and right by spaces). This module creates an object called DOCUMENT including information such as the boundary position of each sentence and the tokens in each sentence, with their positional information. Each token has much information, including, importantly, the type of token: WORD, NUMERAL, or SYMBOL.

The token classification is based on the following simple rule:

1) If a string includes any digit, then it is a NUMERAL string;
2) If a string only includes alphabets, then it is a WORD; or
3) Otherwise, it is a SYMBOL.

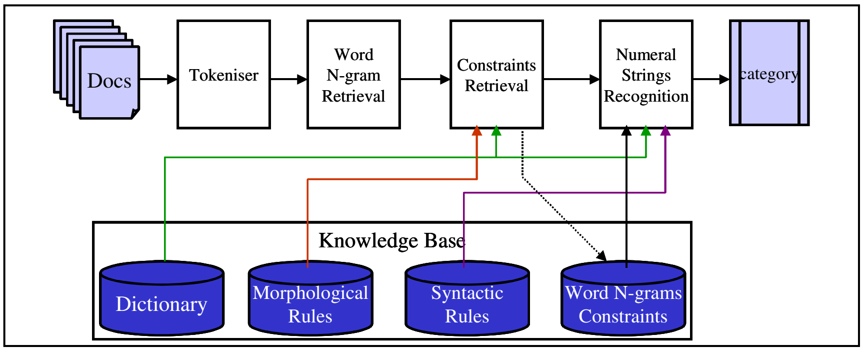


Fig. 1. Architecture of an ENUMS system

For the purpose of sentence boundary understanding, the Tokeniser makes two dummy tokens: BOS – Beginning Of Sentence - and EOS – End Of Sentence. When retrieving the word N-grams for a numeral string, BOS or EOS is the leftmost/rightmost token of word N-grams.

Table 2 shows an example of N-grams constraints. The constraint with ID 3 would hold for a word that is one position (:OFFSET 1) to the right (:D R) of the numeral string, if the word was an adverb (:POS ADV) and it was in capitals (:SPEC CAPITAL).

Table 2. Examples of N-grams constraints based on five features

ID	:D	:OFFSET	:POS	:SYNF	:PREFIX	:SUFFIX	:SPEC
1	L	1	NOUN	PLUR	NIL	NIL	NIL
2	R	1	PREP	“by”	NIL	NIL	CAPITAL
3	R	1	ADV	NIL	NIL	NIL	CAPITAL
4	L	1	PREP	“to”	NIL	NIL	NIL
5	R	1	PREP	“per”	NIL	NIL	PERCENT

The Word N-gram Retrieval module retrieves N consecutive words surrounding each numeral string, including the numeral string itself. The N-gram is described using a NL_iR_j format, signifying N words, i of them to the left and j to the right of the numeral string – so N = i + j – 1. So 3L1R1 means a trigram including the numeral string and one word to its left and right. Similarly, 4L1R2 signifies a tetragram with one word to the left and two to the right of the numeral string. We say that an N-gram of format NL_iR_j is *balanced* if i = j, otherwise it is *unbalanced*. In this paper, we focus on balanced N-grams of formats 3L1R1 and 5L2R2.

The third module is a Constraints Retrieval module. This module retrieves constraints relating to the numeral string's word N-grams with features of words in the N-grams. Each constraint is composed of one to five features as shown in Table 2: POS, syntactic feature, prefix of a numeral string (e.g. the "(" in the numeral string "(02)" that can signify a long distance prefix in a phone number), suffix of a numeral string, and special features (e.g. CAPITAL in "USD" and "GBP", some semantic concepts like MONTH for "January").

Table 3. Examples of category frequency of N-gram constraints (for 3L1R1)

ID	Category Frequency of N-grams Constraints
(1 2)	((MONEY 2) (RATE 1) (AGE 1) (QUANT 1) (NUMBER 1) (DATE 17))
(1 3)	((DATE 51))
(1 5)	((SCORES 3) (CENT 3) (AGE 2) (NUMBER 3) (YEAR 8) (QUANT 3) (DATE 55))
(4 1)	((SCORES 3) (YEAR 1) (AGE 2) (NUMBER 2) (PLURAL 1) (DAY 5) (QUANT 4))
(4 2)	((RANGE 1) (FLOATNUMBER 32) (QUANT 67))

Each constraint will be associated with a particular category frequency distribution in the training data, and this is illustrated in Table 3. Note that actual IDs used are pairs of numbers, like (1 3). It can be seen that constraint (1 3) is always associated with the category DATE in the data, whereas (1 5) is mostly DATE, but also has a scattering of other outcomes. The categories of the numeral strings in the training data (and the test data) were obtained by hand-annotation.

3 ENUMS Recognition Algorithm

In this section, the algorithm of the ENUMS system is discussed, based on word N-grams and associated constraints extracted from training data using up to five features. When a numeral string is found, its N-gram is determined, and this information is passed to the recognition module. The recognition algorithm is described by the pseudocode in Fig. 2.

With a numeral string that contains an affix, the string is also processed by deep tokenisation (e.g. "20.08.2003" → "20 + "." + "08" + "." + "2003"). Then a chart parsing technique [7] using 91 context-free rules that represent the structural form of affixed numeral strings is employed. Each rule is composed of a LHS (left hand side), RHS (right hand side), and constraints on the RHS. For example,

Rule7	RULE-ID:	R7
	LHS:	DATE
	RHS:	(DAY DOT MONTH DOT YEAR)
	Constraints:	((LEAPDATEP DAY MONTH YEAR))

If the recognition module is unable to choose between two or more categories, then word N-gram constraints extracted from the training data are used to select the best single category. For example, if the test numeral string "91 rules", whose annotated category is QUANT, generated word trigrams constraints like (1 2) and (4 2) in Table 3, then the simple addition of categorical frequency of both constraints would be (MONEY 2) (RATE 1) (AGE 1) (QUANT 68) (NUMBER 1) (DATE 17) (RANGE 1) (FLOATNUMBER 32). The most frequent category is QUANT and this would be the disambiguated category for the test numeral string. We employed simple addition of categorical frequency obtained from training data and we plan to implement complex similarity (e.g. Cosine similarity), classification, probability (Bayes rule), or machine learning methods later, and compare these with our current results.

```
Recognise a numeral string's syntactic/semantic category with the following knowledge bases:
dictionary, morphological rules, syntactic rules, and word N-gram constraints.

Input definition: Numeral strings with their word N-grams.
    with detected type of a numeral string: affixed (e.g. "20km/h"), or separate ("2007"),
    IF numeral string type = affixed
    THEN
        Apply numeral string tokeniser (e.g. "20" + "km" + "/" + "h")
        Apply a chart parsing technique to the tokenised strings with rules for numeral string
        processing (e.g. syntactic rules)
        IF Resulting category is NOT ambiguous,
        THEN return the resulting syntactic/semantic category,
        ELSE disambiguate the categories by using word N-gram constraints (i.e. the most
        frequent category is chosen using word N-gram constraints collected from train data),
        END IF

    ELSE (numeral string type = separate)
        Apply separate numeral string understanding rules (which will suggest categories like
        AGE, NUMBER, DAY, etc.)
        IF Resulting category is NOT ambiguous,
        THEN return the resulting syntactic/semantic category,
        ELSE disambiguate the categories by using word N-gram constraints (e.g. the most
        frequent category is chosen using word N-gram constraints collected from train data).
        END IF
    END IF
```

Fig. 2. Recognition algorithm for numeral strings

4 Experimental Results

We collected online newspaper articles (1,111 articles) for a month that cover a range of topics such as domestic, international, sports, economy, and etc. The articles included 12,803 (2.2%) numeral strings among a total of 594,588 strings. To test this system, we produced 10 sets of training and test data. For each set, we randomly selected 10% of test documents (112 among 1,111 documents). Documents in each test set did not overlap each other. Then, for each document set, testing was done using constraints derived using 1, 2, 3, 4, or all 5 features – that is, with just :POS, with :POS and :SYNF, with :POS, :SYNF, and :PREFIX, with :POS, :SYNF, :PREFIX, and :SUFFIX, and with all 5 features. Thus there were in all 10×5 = 50 test setups.

Table 4. Average data size of N-gram approaches

Date Name	Total Articles		Total Numerals		Total Strings	
	Train	Test	Train	Test	Train	Test
Rule-based (average)	91	287 (41)	886	3,215 (459)	48,498	144,030 (20,576)
2L1R0/2L0R1	83	295	915	3,222	* 192,528	
3L1R1	999	111	11,525	1,278	534,525	60,063
5L2R2	999	111	11,525	1,278	534,525	60,063

(* the training and test data were not computed separately.)

Table 4 shows the data size of articles, total numeral strings, and total strings used for training and testing the ENUMS system. The proportion of numeral strings belonging to each category based on 1,111 articles were QUANT (2,245 of 12,803, 17.5%, e.g. "survey of *801* voters"), MONEY (1,282, 10.0%, e.g. "$15m", "$2.55"), DATE (1,114, 98.7%, e.g. "02.12.2003"), FLOATNUMBER (1,099, 8.6%, e.g. "12.5 per cent"), YEAR (1,066, 8.3%, e.g. "in 2003"), NUMBER (1,056, 8.2%, e.g. "800 of the Asian plants"), and DAYTIME (704, 5.5%, e.g. "at 2:30pm") in order. In this paper, the size of total data was tripled compared to [14].

Table 5. Recall/Precision/F-measurement ratios of word N-grams with five features

Test Methods	Precision ratio (%)	Recall ratio (%)	F-measure (%)
Feat-1-3L1R1 Method (Average)	84.4	76.3	80.1
Feat-2-3L1R1 Method (Average)	84.0	79.4	81.6
Feat-3-3L1R1 Method (Average)	83.1	79.5	81.2
Feat-4-3L1R1 Method (Average)	83.7	80.7	82.2
Feat-5-3L1R1 Method (Average)	83.8	81.2	82.5
Feat-1-5L2R2 Method (Average)	85.5	80.8	83.1
Feat-2-5L2R2 Method (Average)	86.2	82.3	84.2
Feat-3-5L2R2 Method (Average)	86.2	82.3	84.2
Feat-4-5L2R2 Method (Average)	86.4	82.7	84.5
Feat-5-5L2R2 Method (Average)	86.6	82.9	84.7

Table 5 shows the performance using word trigrams (i.e. 3L1R1) and pentagrams (i.e. 5L2R2) with five features. The performance depended on the number of features and the number of N-grams. The performance of Feat-1-3L1R1 is the worst among five 3-L1R1 methods and the performance of Feat-1-5L2R2 is the worst among five 5L2R2 methods. The interesting thing in our test is that the number of features did not guarantee improved performance. In the word trigrams, Feat-2 to Feat-5 showed worse performance in precision ratio than Feat-1 (i.e. the differences are 0.4%, 1.3%, 0.7%, and 0.6% respectively).

In the case of the word pentagrams, the performance based on number of features improved monotonically. The addition of :SYNF information (i.e. going from Feat-1-5L2R2 to Feat-2-5L2R2) improved the F-measure by 1.1%. However, the addition of prefix information of a numeral string (i.e. Feat-3-5L2R2) did not improve (or decrease) F-measure. Fig. 3 to Fig. 8 shows the performance of 10 tests with five feature sets in terms of precision, recall, and F-measurement ratios. The performance greatly depended on source knowledge of training data.

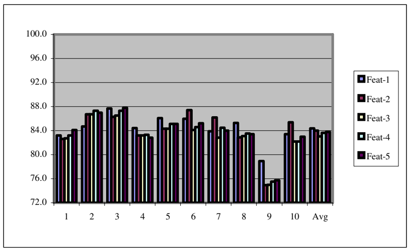


Fig. 3. Precision ratios (%) of 10 tests based on a 3L1R1 method

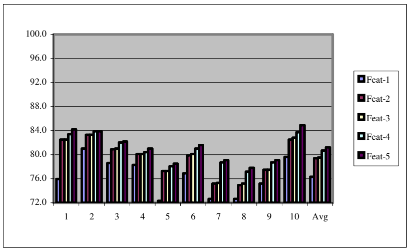


Fig. 4. Recall ratios (%) of 10 tests based on a 3L1R1 method

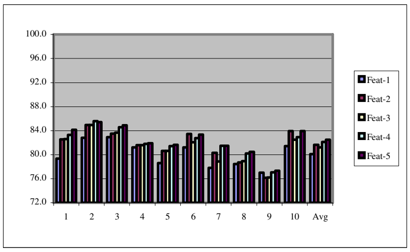


Fig. 5. F-measurement ratios (%) of 10 tests based on a 3L1R1 method

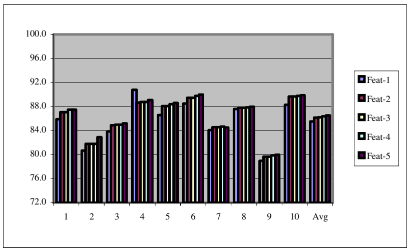


Fig. 6. Precision ratios (%) of 10 tests based on a 5L2R2 method

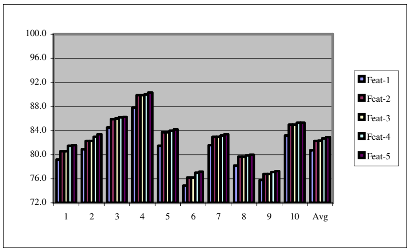


Fig. 7. Recall ratios (%) of 10 tests based on a 5L2R2 method

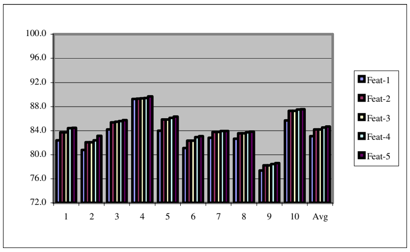


Fig. 8. F-measurement ratios (%) of 10 tests based on a 5L2R2 method

Table 6 shows the performance comparison between rule-based methods [13] and bigrams [14] and our current results with trigrams and pentagrams. The system in [13] was based on a manually generated rule-based method and the system in [14] was based on an automatically generated tabular feature-based method based on two types of bigrams (2L1R0 and 2L0R1).

Table 6. Performance comparison between methods with five features

Test Methods	Precision ratio (%)	Recall ratio (%)	F-measure (%)
Rule-based Method [13]	**86.8**	77.1	81.6
Bigrams Method (Average) [14]	74.5	68.1	71.2
Bigrams Method (best) [14]	83.1	74.5	78.6
Bigrams Method (worst) [14]	57.2	61.1	59.1
Trigrams Method with five features	83.8	81.2	82.5
Pentagrams Method with five features	86.6	**82.9**	**84.7**

The current systems were based on word trigrams (3L1R1), and word pentagrams (5L2R2). The rule-based method did best in precision ratio (86.6%) though the difference compared with pentagrams is just 0.2%. On Recall and F-measurement ratios, pentagrams performed better than other methods. Conclusively, the performance of the time-consuming and manually obtained rule-based method is not better than that of the easy automatic pentagrams method and the performance of the rule-based method greatly depends on the number of rules encoded.

5 Discussion and Conclusions

We focused on interpretation of varieties of numeral strings. Thus direct comparison of our system with other Named Entity recognition systems is not very meaningful, because the systems in e.g. MUC-7 [3] and CoNLL2003 [5] focused on the general recognition task for named entities, using a more limited set of categories: person, location, date, money, and organization, whereas we used 40 categories. In addition, the systems in MUC-7 were trained and tuned by using a training corpus with document preprocessing (e.g. tagging and machine learning). However, performances of MUC-7 systems ranged from 73% to 97% in precision, while our system correctly interpreted 86.6% of the numeral strings using the larger set of categories of numeral string.

For further improvement, different methods related to classification, similarity, and probability will be considered (e.g. application of cosine similarity, Bayes rules, decision tree, etc.). Secondly, the representation of knowledge (i.e. N-gram constraints) will be considered in order to adapt the system to knowledge acquisition (KA) methods like MCRDR (Multiple Classification Ripple-Down Rules) [9], [10]. Third, the integrated method of KA and machine learning algorithms could be applied to extracting domain knowledge (i.e. optimised N-grams constraints) and its model from training data [11]. In this paper, no optimising or learning techniques were applied to get best/optimised N-grams constraints. Fourth, the best value for N in word N-grams (the present system can handle 2 to Max where Max is the maximum string length of the longest sentence) and more effective features will be studied. In addition, the extension of this system to other data like standard biomedical corpora [19] [21] will help to test the effectiveness of our approach.

In conclusion, separate and affixed numeral strings are frequently used in real text. However, there is no system that interprets numeral strings systematically; they are frequently treated as either numerals or nominal entities. In this paper, we discussed two N-gram methods and compared their performance with a manually obtained rule-based method [13] and a word bigrams method [14]. The word trigrams method with five features performs better than rule-based and word bigrams methods in terms of the F-measurement ratio. The word pentagrams method with five features was 0.2% behind in precision ratio but 2.9% ahead in F-measurement ratio compared with the rule-based method. We also found that the addition of extra features to obtain N-gram constraints did not always improve the recognition performance.

References

1. Agrawal, R., Srikant, R.: Searching with Numbers. In: Proceedings of WWW2002, pp. 190–196 (2002)
2. Asahara, M., Matsumoto, Y.: Japanese Named Entity Extraction with Redundant Morphological Analysis. In: Proceedings of HLT-NAACL 2003, pp. 8–15 (2003)
3. Black, W., Rinaldi, F., Mowatt, D.: Description of the NE system used for MUC-7. In: Proceedings of MUC-7 (1998)
4. Chieu, L., Ng, T.: Named Entity Recognition: A Maximum Entropy Approach Using Global Information. In: Proceedings of the 19th COLING, pp. 190–196 (2002)
5. CoNLL-2003 Language-Independent Named Entity Recognition. http://www.cnts.uia.ac.be/conll2003/ner/2 (2003)
6. Dale, R.: A Framework for Complex Tokenisation and its Application to Newspaper Text. In: Proceedings of the second Australian Document Computing Symposium (1997)
7. Earley, J.: An Efficient Context-Free Parsing Algorithm. CACM 13(2), 94–102 (1970)
8. Jarvelin, A., Jarvelin, A., Jarvelin, K.: s-grams: Defining Generalised n-grams for Information Retrieval. Information Processing and Management. 43, 1005–1019 (2007)
9. Kim, Y., Park, S., Kang, B., Choi, Y.: Incremental Knowledge Management of Web Community Groups on Web Portals. In: Karagiannis, D., Reimer, U. (eds.) PAKM 2004. LNCS (LNAI), vol. 3336, pp. 198–207. Springer, Heidelberg (2004)
10. Kim, Y., Park, S., Kang, B., Deards, E.: Adaptive Web Document Classification with MCRDR. In: ITCC 2004. Proceedings of International Conference on Information Technology, Las Vegas USA, pp. 198–207 (2004)
11. Mahidadia, A., Compton, P.: Knowledge Management in Data and Knowledge Intensive Environments. In: Karagiannis, D., Reimer, U. (eds.) PAKM 2004. LNCS (LNAI), vol. 3336, pp. 10–116. Springer, Heidelberg (2004)
12. Maynard, D., Tablan, V., Ursu, C., Cunningham, H., Wilks, Y.: Named Entity Recognition from Diverse Text Types. In: Proceedings of Recent Advances in NLP (2001)
13. Min, K., Wilson, W.H., Moon, Y.: Syntactic and Semantic Disambiguation of Numeral Strings Using an N-gram Method. In: Zhang, S., Jarvis, R. (eds.) AI 2005. LNCS (LNAI), vol. 3809, pp. 82–91. Springer, Heidelberg (2005)
14. Min, K., Wilson, W.H.: Comparison of Numeral Strings Interpretation: Rule-base and Feature-Based N-gram Methods. In: Sattar, A., Kang, B.H. (eds.) AI 2006. LNCS (LNAI), vol. 4304, pp. 1226–1230. Springer, Heidelberg (2006)
15. Nelson, G., Wallis, S., Aarts, B.: Exploring Natural Language - Working with the British Component of the International Corpus of English, John Benjamins, Netherlands (2002)
16. Paradis, F., Nie, J.: Contextual Feature Selection for Text Classification. Information Processing and Management. 43, 344–352 (2007)
17. Polanyi, L., van den Berg, M.: Logical Structure and Discourse Anaphora Resolution. In: ACL 1999. Proceedings of Workshop on The Relation of Discourse/Dialogue Structure and Reference, pp. 110–117 (1999)

18. Reiter, E., Sripada, S.: Learning the Meaning and Usage of Time Phrases from a parallel Text-Data Corpus. In: HLT-NAACL2003. Proceedings of Workshop on Learning Word Meaning from Non-Linguistic Data, pp. 78–85 (2003)
19. Seki, K., Mostafa, J.: A Hybrid Approach to Protein Name Identification in Biomedical Texts. Information Processing and Management. 41, 723–743 (2005)
20. Siegel, M., Bender, E.M.: Efficient Deep Processing of Japanese. In: Proceedings of the 3rd Workshop on Asian Language Resources and International Standardization (2002)
21. Torii, M., Kamboj, S., Vijay-Shanker, K.: An investigation of Various Information Sources for Classifying Biological Names. In: Dignum, F.P.M. (ed.) ACL 2003. LNCS (LNAI), vol. 2922, pp. 113–120. Springer, Heidelberg (2004)
22. Wang, H., Yu, S.: The Semantic Knowledge-base of Contemporary Chinese and its Apllication in WSD. In: Proceedings of the Second SIGHAN Workshop on Chinese Language Processing, pp. 112–118 (2003)
23. Zhou, G., Su, J.: Named Entity Recognition using an HMM-based Chunk Tagger. In: Proceedings of ACL2002, pp. 473–480 (2002)

Using Automated Error Profiling of Texts for Improved Selection of Correction Candidates for Garbled Tokens

Stoyan Mihov[1], Petar Mitankin[1], Annette Gotscharek[2], Ulrich Reffle[2], Klaus U. Schulz[2], and Christoph Ringlstetter[3]

[1] IPP, Bulgarian Academy of Sciences
[2] CIS, University of Munich
[3] AICML, University of Alberta

Abstract. Lexical text correction systems are typically based on a central step: when finding a malformed token in the input text, a set of correction candidates for the token is retrieved from the given background dictionary. In previous work we introduced a method for the selection of correction candidates which is fast and leads to small candidate sets with high recall. As a prerequisite, ground truth data were used to find a set of important substitutions, merges and splits that represent characteristic errors found in the text. This prior knowledge was then used to fine-tune the meaningful selection of correction candidates. Here we show that an appropriate set of possible substitutions, merges and splits for the input text can be retrieved *without any ground truth data*. In the new approach, we compute an error profile of the erroneous input text in a fully automated way, using so-called error dictionaries. From this profile, suitable sets of substitutions, merges and splits are derived. Error profiling with error dictionaries is simple and very fast. As an overall result we obtain an adaptive form of candidate selection which is very efficient, does not need ground truth data and leads to small candidate sets with high recall.

1 Introduction

When faced with an ill-formed input token, lexical text correction systems typically compute a fine-ranked list of correction suggestions from the dictionary. The ranking takes word similarity, word frequencies and other aspects into account [1]. Until today, no efficient methods are known for computing such a ranked list in a direct, one-step manner. Hence, actual systems are typically based on a first step where a set of correction candidates for the ill-formed token is selected from the given background dictionary. At this point, in order to guarantee maximal efficiency, a coarse similarity measure is used. In a second step, candidates are ranked, now using a fine-grained word similarity measure and other scores [1,2,3,4].

In earlier work [5,6], we introduced a direct method for efficiently selecting for a given token V all entries W of the background dictionary where the Levenshtein distance d_L (cf. [7]) between V and W does not exceed a given bound n. These entries are then used as a selection of correction candidates. In tests we found that the method is not fully optimal in the following sense. In a non-neglectable number of cases, the set of candidates does not contain the correct word (reduced recall), and candidate sets may become large (reduced precision). More recently, we introduced an improved method [8] which is based on a refinement of the Levenshtein distance with *restricted sets of substitutions, merges and splits* as edit operations. The new method selects very small candidate sets which nevertheless contain the proper correction in all cases but a few exceptions. The method is very efficient and thus well-suited both for automated and interactive text correction. As an input for the new method, restricted sets of substitutions, merges and splits have to be specified. In [8], these sets are obtained via training, aligning ground truth data with the erroneous text to recognize error types.

In many practical application scenarios, ground truth data are not available. This explains why adaptive techniques become more and more relevant that automatically compute a kind of profile of the given input text. These profiles can be used to improve the behaviour of text correction systems in many ways. In this paper, we show how to efficiently compute restricted sets of substitutions, merges and splits that mirror the errors found in the text without analyzing any ground truth data. We use error dictionaries of a special form to compute an error profile of the given erroneous input text. This form of error profiling is simple and very efficient. From the profile of a text, suitable restricted sets of edit operations are obtained immediately. In our experiments we found that the new method for candidate selection leads to results directly comparable to those obtained from supervised training.

The paper, which combines and refines methods presented in [8,9], has the following structure. Section 2 describes the construction of the error dictionaries used in our experiments. Section 3 then shows how to compute an error profile for a given input text using error dictionaries and how to obtain restricted sets of edit operations from these profiles. Section 4 formally introduces the refinement of the Levenshtein distance with restricted sets of edit operations that is used for candidate selection in our experiments. Comparing candidate selection with standard Levensthein distance and with the new metrics, the latter leads both to better recall and better precision. Section 5 summarizes techniques presented in [8] and shows how to realize candidate selection with the new metrics in a very efficient manner. In Section 6 we present our evaluation results.

2 Constructing Annotated Error Dictionaries

By an *error dictionary* ([10,11]), we mean a collection $\mathcal{E}$ of strings that is generated from a set of correct words $\mathcal{D}$, systematically applying error patterns of a particular form. In an *annotated error dictionary* we store with each erroneous token the corresponding correct word and the transformation (here: edit operation) that was used to generate the entry. The idea behind this new form of a dictionary is simple. Assuming that the base set $\mathcal{D}$ covers most of the tokens of the correct version of the text, and assuming that most errors found in the erroneous text are based on patterns that are taken into account in the error dictionary, error dictionaries help to obtain knowledge on the error distribution found in the input text. This knowledge can be used to improve and fine-tune text correction, in particular the meaningful selection of correction candidates for malformed tokens. One obvious warning is in order: since the same erroneous token might be produced in distinct ways the mapping from errors to correct words in general gives rise to ambiguities.

The following construction focuses on error dictionaries for OCRed texts that help to estimate the frequency of edit operations in OCR errors. The design of such an error dictionary is a delicate matter. We first used a simple brute-force construction, based on a set of typical OCR errors. For some test sets, satisfactory results were obtained. For others, it turned out that some of the relevant error patterns had not been taken into account, and results were disappointing. After a series of other tests we arrived at the following construction.

(a) To the 100,000 most frequent words of our English dictionary we applied all single letter substitutions, a selected list of "typical" merges and splits, and deletion of a letter i, l, t, or f.
(b) In addition, to the 25,000 most frequent words we applied all (other) merges, and
(c) to the 5,000 most frequent words we applied all (other) splits.

Entries of the error dictionary $\mathcal{E}$ were produced by applying only one error transformation at one position. Hence, entries contain just one error. For example, assuming an alphabet with 26 letters, applying all single-letter substitutions to the word "hand" leads to 25^4 errors. Among these, correct words are excluded, see below. The large number of errors obtained from each correct word explains why not all error patterns are applied to all words. The resulting error dictionaries would simply become too large.

For each error W' obtained in this way, we only stored the "prefered" retranslation, i.e., the retranslation that leads to the *most frequent correct* word W. At this point, when garbling a word with one of the non-typical merges and splits (Cases b, c), we reduce its frequency by a penalty factor of $1/100$. We deleted all erroneous tokens with a length ≤ 3 since acronyms and special names of length ≤ 3 can easily be misinterpreted as errors. The error dictionary represents a list of entries of the form (W, W', op) where W' is one of the produced errors, W is the prefered retranslation into a correct word, and op represents the edit operation that leads from W to W'.

For obvious reasons we excluded all errors that by accident correspond to some word in a large collection of standard dictionaries with an overall sum of 3.2 million entries of various languages as well as of person and geographic names. Since each correct word that is left in the error dictionary may lead to a misclassification, it is important to use a collection of dictionaries with very high coverage. The error dictionary $\mathcal{E}$ produced in this way contains $38,794,294$ entries.

3 Computing Error Profiles Using Annotated Error Dictionaries

In the present context, as a profile of a given input text with errors we wish to compute the set of substitutions, merges and splits that represent errors found in the text with a relative occurrence frequency larger than some standard threshold.

Alignment lists. Given an erroneous input text and the annotated error dictionary, the first step is to produce an "alignment list" $\mathcal{L}_{al}$ containing triples (W, W', op). The strings W' are the normal tokens found in the input text. A token is called "normal" if it is composed of standard letters only. If token W' represents an error catched in $\mathcal{E}$, the associated triple (W, W', op) as specified in $\mathcal{E}$ is added to $\mathcal{L}_{al}$. If W' is not found in $\mathcal{E}$ but in the base lexicon of correct words $\mathcal{D}$, W' is considered to be a correct token: we add $(W', W', \emptyset)$ to $\mathcal{L}_{al}$. Tokens that are found neither in $\mathcal{E}$ nor in $\mathcal{D}$ are not used for the alignment list. This procedure takes into account that the error dictionary only partially covers the errors and also prevents that erroneous tokens missing in $\mathcal{E}$ are misinterpreted as correct words.

It is important to note that no groundtruth data is involved in this procedure. Furthermore the error dictionary is made for the purpose of OCR postcorrection but is in no way adjusted to a certain OCR engine or to some document type.

Obtaining the relevant set of restricted edit operations. From $\mathcal{L}_{al}$ we can easily compute estimated frequencies for all symbol dependent substitutions, merges and splits $f(\sigma \mapsto \sigma')$ as well as $f(\sigma)$ for all symbols and 2-grams. The score for each edit operation is its relative frequency: $f_{rel}(\sigma \mapsto \sigma') = f(\sigma \mapsto \sigma')/f(\sigma)$. By $f(\sigma)$, we denote the number of occurrences of σ in the left components W of entries (W, W', op) or $(W', W', \emptyset)$ of $\mathcal{L}_{al}$. Thresholds *subs*, *merge*, and *split* define the set of restricted substitutions, merges, and splits. A substitution $\sigma \mapsto \sigma'$ is only used as a restricted substitution during the selection of answer candidates if the estimated relative frequency of $\sigma \mapsto \sigma'$ is larger than *subs*. *merge* is the corresponding threshold for merges, *split* is the threshold for splits.

Implementation and efficiency aspects. The just under 40 million entries of the error dictionary are stored in a minimal deterministic finite-state automaton to grant a compact representation and very fast lookup times. In an offline step, it takes about 3 minutes and 30 seconds to compute the automaton with approx. 1.5 million states, its size in memory (including the annotations) is 513 MB. The estimation at runtime is basically a sequence of lookups for all normal tokens of the text and thus very fast. After loading the automaton, 1,000 tokens of text are processed in $\sim$ 35ms.

4 Refined Levenshtein Distances for Candidate Selection

We assume that words are composed over a fixed finite alphabet, Σ. Recall that the standard *Levenshtein distance* between two words $V, W \in \Sigma^*$ is the minimal number of edit operations needed to rewrite V into W. Standard edit operations are the substitution of a letter by another letter, the deletion of a letter, and the insertion of a letter.

Definition 1. *Let three sets $S_{1,1} \subseteq \Sigma \times \Sigma$, $M_{2,1} \subseteq \Sigma^2 \times \Sigma$, $S_{1,2} \subseteq \Sigma \times \Sigma^2$ be given. A* restricted substitution *is an operation where we replace one occurrence of a letter σ in a word W by another letter σ' where $\langle \sigma, \sigma' \rangle \in S_{1,1}$. A* restricted merge *is defined similarly, replacing an occurrence of a bigram $\sigma\sigma'$ of consecutive letters in W by σ'' for $\langle \sigma\sigma', \sigma'' \rangle \in M_{2,1}$. By a* restricted split *we mean an operation where one occurrence of a letter σ of W is replaced by a bigram $\sigma'\sigma''$ such that $\langle \sigma, \sigma'\sigma'' \rangle \in S_{1,2}$. Insertions and deletions are defined as usual. Each restricted substitution, merge, split, and each insertion and deletion is called a* refined edit operation.

Definition 2. *Let $S_{1,1}$, $M_{2,1}$, and $S_{1,2}$ as above. Let $V, W \in \Sigma^*$. The* refined Levenshtein distance *between V and W, denoted $d_{ref}(V, W)$, is the minimal number of non-overlapping refined edit operations that are needed to transform V into W.*

A sequence of edit operations is non-overlapping if substrings introduced by edit operations are not modified by later operations. It is trivial to see that the refined Levenshtein distance between two words can be computed via dynamic programming, using a variant of the well-known Wagner-Fischer scheme [12].

Given an erroneous token V of the input text and a bound n, we wish to retrieve the set of all dictionary entries W such that $d_{ref}(V, W) \leq n$. Before we describe our experiments with this form of candidate selection, we show how the candidate sets can be computed in an efficient way.

5 Universal Levenshtein Automata and Fast Candidate Selection in Dictionaries

The description of our algorithm for candidate selection proceeds in two steps. We first sketch candidate selection based on the ordinary Levenshtein distance. It is impossible to give a full description here, for details we refer to [6]. Afterwards we sketch the modifications that are needed for candidate selection with the new metrics.

Universal automaton for standard Levenshtein distance. Candidate selection for the traditional Levenshtein distance is strongly based on a special kind of automaton, called *universal Levenshtein automaton* ([6]). For each distance bound n, there is exactly one universal Levenshtein automaton $\mathcal{A}_n^\forall$, which is computed offline only once. The automaton is universal in the sense that it can be applied to any pair of words V, W. One of the words, say V, is treated as input. The distributions of the letters of V in "relevant" subwords of W, formalized as bitvectors, then represent the input for $\mathcal{A}_n^\forall$. The exact form of the input, given V and W, is explained in Remark 1. The sequence of bitvectors is accepted if and only if $d_L(V, W) \leq n$. Details about the construction of the universal Levenshtein automaton can be found in [6].

Candidate selection in dictionaries for standard Levenshtein distance. Assume now we have a dictionary D, encoded as a deterministic finite state automaton $\mathcal{A}_D$. Given an input string W, we want to use $\mathcal{A}_n^\forall$ to find all entries $V \in D$ such that $d_L(V, W) \leq n$. The task is solved by a simple parallel backtracking procedure. Beginning at the initial state we traverse $\mathcal{A}_D$, visiting all prefixes $P = \sigma_1 \ldots \sigma_{k-1}$ of dictionary words. At each forward step, the label σ_k of the transition is matched against the relevant subword of W to produce a bitvector χ_k which serves as input for $\mathcal{A}_n^\forall$. Note that in general there are distinct options for selecting σ_k. If the transition in the "control" automaton $\mathcal{A}_n^\forall$ fails, we choose another possible transition in $\mathcal{A}_D$ with label σ'_k. If no choices are left, we go back to the previous state and prefix $P = \sigma_1 \ldots \sigma_{k-2}$. We output the prefix P once we reach a final state in both automata.

Forward-backward method. It should be mentioned that the evaluation results presented below make use of a highly effective optimization ([6]) in order to speed up the search. Typically, automata encoding a dictionary D for a given natural language have a characteristic structure. The branching degree of nodes is very high in the initial part of the automaton close to the start state ("initial wall"). A huge number of distinct states can be reached with the first two, three transitions. Deeper inside, when 3, 4 nodes are traversed, most nodes only have a very small number of successors. The "forward-backward method" ([6]) uses this observation to speed up approximate search. Given D, two automata $\mathcal{A}_D$ and $\mathcal{A}_D^{rev}$ are used. The former (latter) encodes all words of D in the usual, left-to-right (reverse, right-to-left) order. An erroneous token W given as input is split into two parts W_1, W_2 of approx. the same length. In order to find all lexical neighbours of $W = W_1W_2$ in a given edit distance, we first assume that the majority (at least half) of all errors is found in W_2. Since in W_1 the number of possible errors is reduced, not too much backtracking is necessary when traversing the initial wall of $\mathcal{A}_D$, looking for possible variants of W_1. Variants of W_2 are also found quickly since in this part of the automaton $\mathcal{A}_D$ the branching degree of nodes is very small. The second subsearch is dual. We assume that most errors are concentrated on W_1. We now traverse $\mathcal{A}_D^{rev}$, using the reversed input string $W^{rev} = W_2^{rev}W_1^{rev}$. Again we quickly pass through the initial wall of $\mathcal{A}_D^{rev}$, and possible variations of W_1^{rev} are found easily. The speed-up of this method is striking, see [6].

Universal Levenshtein automata for refined Levenshtein distances. For candidate selection with the new metrics, we only have to replace the universal Levenshtein automaton $\mathcal{A}_n^{\forall}$ by a new type of universal automaton for bound n, all other techniques from above remain the same. In what follows, by $d_{S_{1,1},M_{2,1},S_{1,2}}$ we denote the refined Levenshtein distance for the sets $S_{1,1}$, $M_{2,1}$, and $S_{1,2}$ as specified in Definition 1. The index is omitted if the sets are clear from the context. By $d_{S_{1,1}}$ we denote the simplified version of the refined Levenshtein distance where $M_{2,1} = S_{1,2} = \emptyset$. In what follows we illustrate the universal Levenshtein automata $\mathcal{A}_n$ for the special metrics $d_{S_{1,1}}$, and we show how these automata are applied. Due to space limitations, the very technical computation of the automata $\mathcal{A}_n$ cannot be described here. At the end of the section we indicate the modifications that are necessary to cope with the general situation where the sets $M_{2,1}$ and $S_{1,2}$ may be non-empty.

$\mathcal{A}_n$ only depends on the maximum number of refined edit operations n that we want to tolerate, not on the choice of $S_{1,1}$. The automaton $\mathcal{A}_1$ for $d_{S_{1,1}}$ is shown in Figure 1. Given two words V and W, the set $S_{1,1}$ is used to calculate the actual input $i(V, W)$ for the automaton $\mathcal{A}_n$. $\mathcal{A}_n$ accepts $i(V, W)$ only if $d_{S_{1,1}}(V, W) \leq n$. We build $i(V, W)$ using the so-called characteristic vectors.

Definition 3. *For each letter $c \in \Sigma$ and each word $a_1a_2...a_r \in \Sigma^*$ we define the* characteristic vectors

1. $\chi(c, a_1a_2...a_r) = b_1b_2...b_r$, *where* $b_j = 1$ *if* $c = a_j$ *and* $b_j = 0$ *otherwise,*
2. $\chi_{sub}(c, a_1a_2...a_r) = f_1f_2...f_r$, *where* $f_j = 1$ *if* $\langle a_j, c\rangle \in S_{1,1}$ *and* $f_j = 0$ *otherwise.*

Definition 4. *Given the two words $V, W \in \Sigma^*$ we define the input $i(V, W)$ for $\mathcal{A}_n$ as the sequence of pairs of characteristic vectors $\alpha_1\alpha_2...\alpha_{|W|}$ where $\alpha_i = \langle\beta_i, \beta_i^{sub}\rangle$ and*

1. $\beta_i = \chi(W_i, V_{i-n}V_{i-n+1}...V_k)$, *where* $k =$ min$(|V|, i + n + 1)$, $V_{-n+1} = \ldots = V_0 = \$$ *for* $n > 0$. *Here* $\$ \notin \Sigma$ *is a new symbol.*
2. $\beta_i^{sub} = \chi_{sub}(W_i, V_{i-n+1}V_{i-n+2}...V_k)$, *where* $k =$ min$(|V|, i + n - 1)$, $V_{-n+2} = \ldots = V_0 = \$$ *for* $n > 1$.

For example, let $\Sigma = \{a, b, c, ..., z\}$ and $S_{1,1} = \{\langle a, d\rangle, \langle d, a\rangle, \langle h, k\rangle, \langle h, n\rangle\}$. Let $V = hahd$ and $W = hand$. Then $i(V, W) = \alpha_1\alpha_2\alpha_3\alpha_4$ where

$$\begin{aligned}
\alpha_1 &= \langle\beta_1, \beta_1^{sub}\rangle = \langle\chi(h, \$hah), \chi_{sub}(h, h)\rangle = \langle 0101, 0\rangle,\\
\alpha_2 &= \langle\beta_2, \beta_2^{sub}\rangle = \langle\chi(a, hahd), \chi_{sub}(a, a)\rangle = \langle 0100, 0\rangle,\\
\alpha_3 &= \langle\beta_3, \beta_3^{sub}\rangle = \langle\chi(n, ahd), \chi_{sub}(n, h)\rangle = \langle 000, 1\rangle,\\
\alpha_4 &= \langle\beta_4, \beta_4^{sub}\rangle = \langle\chi(d, hd), \chi_{sub}(d, d)\rangle = \langle 01, 0\rangle.
\end{aligned}$$

Remark 1. The corresponding input for the universal Levenshtein automaton (standard Levenshtein distance) is just given by the sequence of the first components of the input tuples described above. In the example, the input for V and W would be $\langle 0101\rangle\langle 0100\rangle\langle 000\rangle\langle 01\rangle$.

The automaton $\mathcal{A}_1$ is shown in Fig. 1. In the figure, any x represents a don't care symbol that can be interpreted as 0 or 1, and expressions in round brackets are optional. For instance, from the state $\{I{-}1^{\#1}, I^{\#1}, I{+}1^{\#1}\}$ with $\langle 010(x), x\rangle$ we can reach the state $\{I^{\#1}\}$. This means that from $\{I{-}1^{\#1}, I^{\#1}, I{+}1^{\#1}\}$ we can reach $\{I^{\#1}\}$ with $\langle 010, 0\rangle$, $\langle 010, 1\rangle$, $\langle 0100, 0\rangle$, $\langle 0100, 1\rangle$, $\langle 0101, 0\rangle$ and $\langle 0101, 1\rangle$. In the above example we start from the initial state $\{I^{\#0}\}$; with input $\langle 0101, 0\rangle$, $\langle 0100, 0\rangle$, $\langle 000, 1\rangle$ and $\langle 01, 0\rangle$ we visit the states $\{I^{\#0}\}$, $\{I^{\#0}\}$, $\{I{-}1^{\#1}, I^{\#1}\}$, and $\{M^{\#1}\}$. $\{M^{\#1}\}$ is a final state. Hence $d_{S_{1,1}}(V, W) \leq 1$.

The universal automaton $\mathcal{A}'_n$ for the refined Levenshtein distance $d_{S_{1,1},M_{2,1},S_{1,2}}$ is similar to $\mathcal{A}_n$. The difference is that the input $i'(V, W)$ is a sequence of quadruples of bitvectors, because in addition to χ and χ_{sub} we use two other characteristic vectors - one for the restricted merge and one for the restricted split. The sets $M_{2,1}$ and $S_{1,2}$ are used for the computation of these two additional vectors respectively.

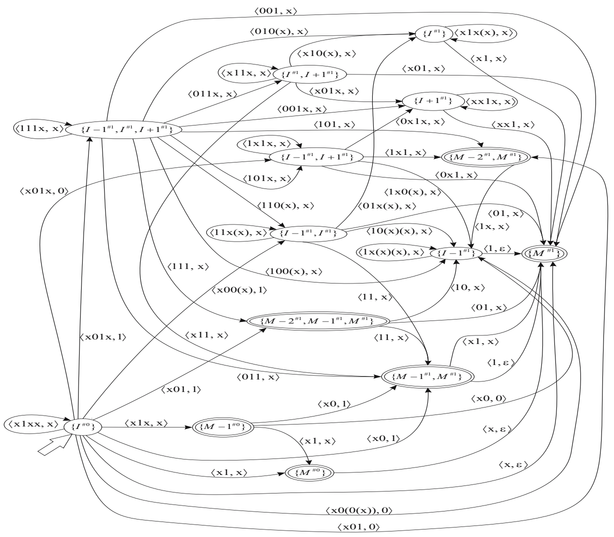


Fig. 1. Universal Levenshtein automaton $\mathcal{A}_1$ for refined Levenshtein metrics

6 Evaluation Results

As a test set for the new method we used the TREC-5 corpus with character error rate 5% (http://trec.nist.gov/data/t5_confusion.html) and an English dictionary with 264,061 words. Via dynamic programming we extracted from the corpus couples of the type $\langle \mathit{pattern}, \mathit{original} \rangle$, where *pattern* is an OCRed word and *original* is its corresponding original. The problem of case errors was ignored for these tests.

From one half of the test set, we extracted the couples $\langle \mathit{pattern}, \mathit{original} \rangle$ where $\mathit{pattern} \neq \mathit{original}$ and where *original* was found in the dictionary. This subset represents all recognition errors that can possibly be corrected with the applied dictionary. For each *pattern* we selected a set of correction candidates from the dictionary.

To have a baseline comparison, we first made experiments with Levenshtein automata for the standard Levenshtein distance (lines (a) in Table 1) and automata allowing unrestricted substitutions, merges and splits (lines (b)). We then repeated the tests with restricted edit operations. The sets of allowed operations were determined by the thresholds [1] given in Table 1 and relative frequencies derived either from supervised training (lines (c)) or from an error profile obtained with error dictionaries as described above (lines (d)). Only for (c) we used the remaining half of the test set for supervised training.

In Table 1, n denotes the maximal number of (restricted) edit operations. For $n = 1$, we used $100,000$ test patterns for evaluation. For $n = 2$, the number of test patterns is $50,000$. For $n = 3$, the number of test patterns is $23,000$. With *len*, we denote the possible length for an input pattern. In practice, a distance bound $n = 1$ ($n = 2, n = 3$) is mainly interesting for words of length $l \leq 6$ ($7 \leq l \leq 12, 13 \leq l$), which explains our focus. The number *cand* gives the average number of correction candidates per pattern found in the dictionary. The value of *cand* depends on the distance bound, the length of the input pattern and the thresholds for the substitutions, merges and splits. With *time*, we denote the average time per pattern needed to find and output all correction candidates. With *recall*, we denote the percentage of patterns where the correct *original* is found in the selected set of answer candidates.

As lines (d) show, the new metrics based on the estimation with error dictionaries lead to substantially improved results: For $n = 1$ and also for $n = 3$, the candidate sets obtained with standard Levenshtein distance were larger but lead to poorer recall. When we allowed unrestricted substitutions, merges and splits, the recall was basically the same, but the extraction is much slower and the size of the candidate sets is unacceptably high. For $n = 2$, a significantly higher recall (99.97%, compared to 95.18% with standard Levenshtein distance) is obtained at the cost of larger candidate sets: 18.73 (as compared to 12.24) candidates on average. Only for $n = 2$ the estimation with error dictionaries was clearly outperformed by supervised training with ground truth data where the candidate sets are much smaller (3.8 on average). This might be due to an important edit operation that went unnoticed by the automatic error profiling.

The CPU of the machine used for the experiments is Pentium 4, 2.4 GHz with 1GB RAM.

[1] The thresholds are standard values that proved suitable for the corpus.

Table 1. Test for distance bounds $n = 1, 2, 3$

		subs	*merge*	*split*	*cand*	*recall*	*time*
$n = 1, len = 1 - 6$							
(a)	standard	0	1	1	7.78	70.565%	0.032 ms
(b)	unrestricted	0	0	0	45.73	94.52%	0.107 ms
(c)	training	0.0006	0.0325	0.0005	5.48	94.519%	0.049 ms
(d)	err.dics.	0.0005	0.01	0.0005	6.85	94.519%	0.068 ms
$n = 2, len = 7 - 12$							
(a)	standard	0	1	1	12.24	95.178%	0.37 ms
(b)	unrestricted	0	0	0	96.88	99.988%	2.448 ms
(c)	training	0.01	0.0002	0.0004	3.8	99.986%	0.487 ms
(d)	err.dics.	0.01	0.02	0	18.73	99.966%	1.67 ms
$n = 3, len > 12$							
(a)	standard	0	1	1	4.1	99.81%	1.214 ms
(b)	unrestricted	0	0	0	43.971	100%	6.623 ms
(c)	training	0.04	0.03	1	4.029	100%	1.08 ms
(d)	err.dics.	0.01	0.01	1	4.029	100%	1.08 ms

7 Conclusion and Future Work

In this paper we showed that error dictionaries may be used to compute profiles for erroneous input texts that help to fine-tune the selection of correction candidates for malformed tokens from a dictionary. The computation of the profile for a given input text is simple, fast, and completely automatic. Combining this technique with a recently introduced method for accessing dictionaries, we obtain an automated procedure for candidate selection which is adaptive in the sense that the specifics of the input text are taken into account. The method leads to very small candidate sets with high recall.

Acknowledgements. This work was supported by AICML, iCore, DFG and VolkswagenStiftung.

References

1. Kukich, K.: Techniques for automatically correcting words in texts. ACM Computing Surveys, 377–439 (1992)
2. Owolabi, O., McGregor, D.: Fast approximate string matching. Software Practice and Experience 18(4), 387–393 (1988)
3. Zobel, J., Dart, P.: Finding approximate matches in large lexicons. Software Practice and Experience 25(3), 331–345 (1995)
4. Strohmaier, C., Ringlstetter, C., Schulz, K.U., Mihov, S.: A visual and interactive tool for optimizing lexical postcorrection of OCR results. In: DIAR 2003. Proceedings of the IEEE Workshop on Document Image Analysis and Recognition (2003)
5. Schulz, K.U., Mihov, S.: Fast String Correction with Levenshtein-Automata. International Journal of Document Analysis and Recognition 5(1), 67–85 (2002)
6. Mihov, S., Schulz, K.U.: Fast approximate search in large dictionaries. Computational Linguistics 30(4), 451–477 (2004)
7. Levenshtein, V.I.: Binary codes capable of correcting deletions, insertions, and reversals. Sov. Phys. Dokl. (1966)

8. Schulz, K.U., Mihov, S., Mitankin, P.: Fast selection of small and precise candidate sets from dictionaries for text correction tasks. In: ICDAR. Proceedings of the ninth International Conference on Document Analysis and Recognition (to appear, 2007)
9. Ringlstetter, C., Reffle, U., Gotscharek, A., Schulz, K.U.: Deriving symbol dependent edit weights for text correction - the use of error dictionaries. In: ICDAR. Proceedings of the ninth International Conference on Document Analysis and Recognition (to appear, 2007)
10. Arning, A.: Fehlersuche in großen Datenmengen unter Verwendung der in den Daten vorhandenen Redundanz. PhD thesis, University of Osnabrück (1995)
11. Ringlstetter, C., Schulz, K.U., Mihov, S.: Orthographic errors in web pages: Towards cleaner web corpora. Computational Linguistics 32(3), 295–340 (2006)
12. Wagner, R.A., Fisher, M.: The string-to-string correction problem. Journal of the ACM 21(1), 168–173 (1974)

Hypothesis Generation and Maintenance in the Interpretation of Spoken Utterances

M. Niemann, I. Zukerman, E. Makalic, and S. George

Faculty of Information Technology, Monash University
Clayton, Victoria 3800, Australia
{niemann,ingrid,enesm,sarahg}@infotech.monash.edu.au

Abstract. The DORIS project (Dialogue Oriented Roaming Interactive System) aims to develop a spoken dialogue module for an autonomous robotic agent. This paper examines the techniques used by *Scusi?*, the speech interpretation component of DORIS, to postulate and assess hypotheses regarding the meaning of a spoken utterance. The results of our evaluation are encouraging, yielding good interpretation performance for utterances of different types and lengths.

1 Introduction

The DORIS project (Dialogue Oriented Roaming Interactive System) aims to develop a spoken dialogue module for an autonomous robotic agent. The purpose of this module is to engage in a dialogue with users, and generate plans for the robot that require physical as well as dialogue actions. These plans are created by interfacing with an external planner module. In this paper, we describe *Scusi?*, DORIS's speech interpretation module, focusing on the techniques used to postulate and assess hypotheses regarding the meaning of a spoken utterance.

It is widely accepted that spoken dialogue computer systems are more prone to errors than similar text-based systems. This may be attributed to the inaccuracy of current Automatic Speech Recognition (ASR) systems, disfluencies common in spoken discourse (e.g., false starts, repeats, etc), and the fact that spoken discourse is often less grammatical than written discourse.

In general, during discourse interpretation, a system must be able to (1) postulate promising interpretations, (2) adjust these interpretations dynamically as new information becomes available, and (3) recover from erroneous interpretations. The adjustment and recovery activities are particularly important for the interpretation of spoken discourse. Further, dialogue systems can interact with users to obtain additional information when so warranted by the state of the interpretation process and the dialogue conditions. For example, imagine a situation where an ASR mishears the last part of the request "get me a mug", producing texts that differ mainly in their last portion. If only a few texts are generated, the dialogue module could ask a specific question, e.g., "did you say mug or mud?", but if there are several options, an open-ended question such as "what did you want me to get?" would be more suitable. Similarly, if there are two possible mugs, a reasonable clarification question would be "the green one or the red one?", while if there are several options, then a more appropriate question would be "which mug should I get?". A robot-mounted dialogue system has yet another option — it can direct the robot to perform a physical action (getting closer to a referenced object) instead of asking a question.

The *Scusi?* dialogue interpretation module is designed to support these capabilities. It implements an interpretation process comprising three stages: speech recognition, parsing and semantic interpretation. Each stage in this process produces multiple candidate options, which are ranked according to the probability of them matching the speaker's intention (Section 3). This probabilistic framework supports the (re-)ranking of interpretations as new information becomes available, as well as the identification of 'trusted' (high probability) and 'untrusted' (low probability) regions in an interpretation. The generation and maintenance of multiple interpretations at each stage of the process allows *Scusi?* to recover from erroneous interpretations, and to abstract features of the state of the dialogue which support the generation of appropriate dialogue or physical actions. Features of interest (for each stage of the interpretation process) include: the number of highly ranked interpretations, the similarity between these interpretations, and the probability of each interpretation.

This paper is organized as follows. Section 2 presents an overview of the interpretation process. The estimation of the probability of an interpretation is presented in Section 3, and the process for generating a final interpretation is described in Section 4. Section 5 details the evaluation of the interpretation mechanism. Related research and concluding remarks are given in Sections 6 and 7 respectively.

2 Multi-stage Processing

Scusi? processes spoken input in three stages: speech recognition, parsing and semantic interpretation. In the first stage, the dialogue module runs the ASR software (Microsoft Speech SDK 5.1) to generate candidate sequences of words (i.e., plain text) from a speech signal. Each sentence is assigned a score that reflects the probability of the words given the speech wave. The second stage applies Charniak's probabilistic parser (`ftp://ftp.cs.brown.edu/pub/nlparser/`) to generate parse trees from the plain text sequences. The parser generates n possible parse trees for each text, where each parse tree is associated with a probability. During the semantic interpretation phase, parse trees are successively mapped into two representations based on Conceptual Graphs (CGs) [1]: first *Uninstantiated Concept Graphs (UCGs)*, and then *Instantiated Concept Graphs (ICGs)*. UCGs are obtained from parse trees deterministically — one parse tree generates one UCG (but a UCG can have several parent parse trees). A UCG represents syntactic information, where the concepts correspond to the words in the parent parse tree, and the relations between the concepts are directly derived from syntactic information in the parse tree and prepositions. This process is similar to the assignment of semantic role labels [2].

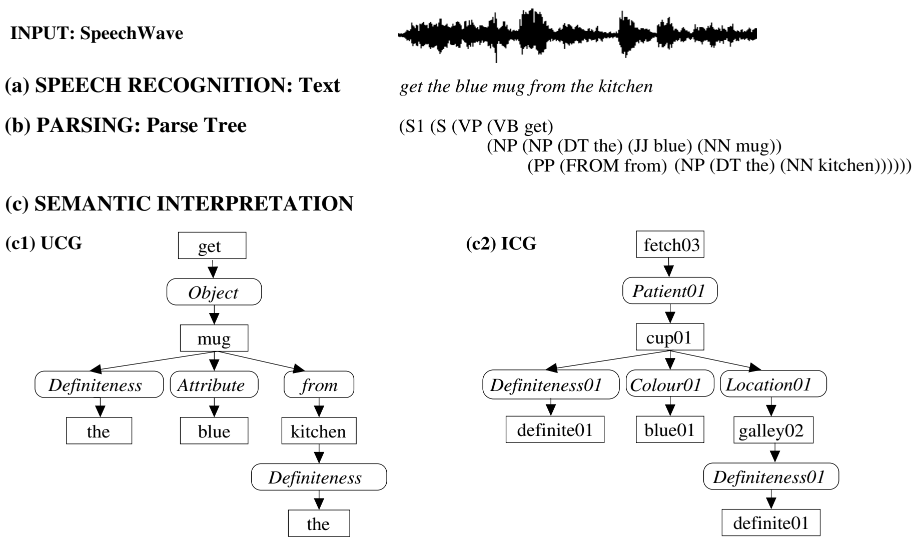


Fig. 1. Structures for the interpretation stages

A UCG can generate several ICGs (and an ICG can have several parent UCGs). An ICG is built by nominating an *Instantiated Concept* from DORIS's knowledge base as a potential realization for each *Uninstantiated Concept* in a UCG (Section 4). Figure 1 illustrates this process with respect to the input sentence "get the blue mug from the kitchen". For instance, the noun 'mug' in the parse tree is mapped to the word `mug` in the UCG, which in turn is mapped to the instantiated concept `cup01` in the ICG.

The consideration of all possible options at each stage of the interpretation process is computationally intractable. Hence, *Scusi?* uses an *anytime* algorithm [3] which applies a selection-expansion cycle to build a search graph. This process is repeated until one of the following is true: all options are fully expanded, a time limit is reached, or the system runs out of memory. At any point after completing an expansion, *Scusi?* can return a list of ranked interpretations with their parent sub-interpretations.

3 Estimating the Probability of an Interpretation

Scusi? ranks candidate ICGs according to their probability of matching the intended meaning of a spoken utterance. The probability of an ICG is estimated from the probabilities of its parent UCGs and the context. Currently, the context encompasses domain knowledge and salience from dialogue history. In the future, the context will also include data from the robot's vision system.

Formally, we denote the user's speech wave by $W \in \mathcal{W}$, where $\mathcal{W}$ is a countable set of spoken utterances. The task is to estimate $\Pr(I|W, \mathcal{C})$, the probability of an ICG I given the speech signal W and the context $\mathcal{C}$. The probability function $\Pr(I|W,\mathcal{C})$ is estimated as

$$\Pr(I|W,\mathcal{C}) \propto \sum_{\Gamma=\{P,U\}} \Pr(I|U,\mathcal{C}) \cdot \Pr(U|P) \cdot \Pr(P|T) \cdot \Pr(T|W) \tag{1}$$

where U, P and T denote UCG, parse tree and plain text interpretations respectively. The summation is taken over all possible paths $\Gamma = \{P,U\}$ from the parse tree to the ICG. This is necessary since a UCG and an ICG can have more than one parent. Charniak's parser and the ASR return an estimate of $\Pr(P|T)$ and $\Pr(T|W)$ respectively. In addition, it is known that $\Pr(U|P) = 1$, since the process of converting a parse tree into a UCG is deterministic. Hence, *Scusi?* only has to estimate $\Pr(I|U,\mathcal{C})$.

Consider an ICG I containing concepts $n^{ICG} \in \Omega_n$ and relations $r^{ICG} \in \Omega_r$ (Ω_n and Ω_r are the concepts and relations in the domain knowledge respectively). A parent UCG is denoted by U and comprises concepts $n^{UCG} \in \Gamma_n$ and relations $r^{UCG} \in \Gamma_r$ (Γ_n and Γ_r are the words and relations from which UCGs are built). The probability of I given U and context $\mathcal{C}$ can then be stated as follows.

$$\begin{aligned}\Pr(I|U,\mathcal{C}) &= \prod_{n^{ICG}\in I, r^{ICG}\in I} \Pr\left(n^{ICG}, r^{ICG}|n^{UCG}, r^{UCG}, \Omega_n^-, \Omega_r^-, \mathcal{C}\right)\\ &= \prod_{n^{ICG}\in I, r^{ICG}\in I} \left\{\begin{array}{l}\Pr\left(r^{ICG}|n^{ICG}, n^{UCG}, r^{UCG}, \Omega_n^-, \Omega_r^-, \mathcal{C}\right) \times\\ \Pr\left(n^{ICG}|n^{UCG}, r^{UCG}, \Omega_n^-, \Omega_r^-, \mathcal{C}\right)\end{array}\right\}\end{aligned} \tag{2}$$

where the superscript in Ω_n^- and Ω_r^- denotes the sets Ω_n and Ω_r without the concept n^{ICG} and relation r^{ICG} respectively. It is difficult to estimate Equation 2, as each concept and relation in an ICG depends on the other ICG concepts and relations. We therefore make the following simplifying assumptions: (1) the probability of an ICG relation depends only on the corresponding UCG relation, the neighbouring ICG concepts, and the context; and (2) the probability of an ICG concept depends only on the corresponding UCG concept, the neighbouring ICG relations, and the context. These assumptions yield

$$\Pr(I|U,\mathcal{C}) \approx \prod_{r^{ICG}\in I}\Pr\left(r^{ICG}|r^{UCG}, n_p^{ICG}, n_c^{ICG}, \mathcal{C}\right) \prod_{n^{ICG}\in I}\Pr\left(n^{ICG}|n^{UCG}, r_p^{ICG}, r_c^{ICG}, \mathcal{C}\right) \tag{3}$$

where n^{UCG} and r^{UCG} denote the UCG concept and relation corresponding to the ICG concept n^{ICG} and relation r^{ICG} respectively. We consider the neighbouring concepts of relation $r^{ICG} \in I$ to be its parent concept $n_p^{ICG} \in I \bigcap \Omega_n$ and its child concept $n_c^{ICG} \in I \bigcap \Omega_n$, and the neighbouring relations of concept $n^{ICG} \in I$ to be its parent relation $r_p^{ICG} \in I \bigcap \Omega_r$ and its child relation $r_c^{ICG} \in I \bigcap \Omega_r$.

After applying Bayes rule, and making additional simplifying assumptions about conditional dependencies, we obtain

$$\begin{aligned}\Pr(I|U,\mathcal{C}) \approx &\prod_{r^{ICG}\in I}\Pr\left(r^{UCG}|r^{ICG}\right)\Pr\left(r^{ICG}|n_p^{ICG}, n_c^{ICG}, \mathcal{C}\right)\Pr\left(n_p^{ICG}|\mathcal{C}\right)\Pr\left(n_c^{ICG}|\mathcal{C}\right) \times\\ &\prod_{n^{ICG}\in I}\Pr\left(n^{UCG}|n^{ICG}\right)\Pr\left(n^{ICG}|r_p^{ICG}, r_c^{ICG}, \mathcal{C}\right)\Pr\left(r_p^{ICG}|\mathcal{C}\right)\Pr\left(r_c^{ICG}|\mathcal{C}\right)\end{aligned} \tag{4}$$

The last two factors in the first (second) row of Equation 4 represent the prior probabilities of the concepts (relations) in an ICG in light of the context. Hence, moving the concept (relation) factors to the second (first) row will not change the equation. We also collapse the two concept (relation) factors into a single one in order to avoid double counting a node that is both a parent and a child (which is a result of the simplification of the structure of an ICG performed for the above calculations). This yields the following equation.

$$\begin{aligned}\Pr(I|U,\mathcal{C}) \approx & \prod_{r^{ICG}\in I} \Pr\left(r^{UCG}|r^{ICG}\right) \Pr\left(r^{ICG}|n_p^{ICG}, n_c^{ICG}, \mathcal{C}\right) \Pr\left(r^{ICG}|\mathcal{C}\right) \times \\ & \prod_{n^{ICG}\in I} \Pr\left(n^{UCG}|n^{ICG}\right) \Pr\left(n^{ICG}|r_p^{ICG}, r_c^{ICG}, \mathcal{C}\right) \Pr\left(n^{ICG}|\mathcal{C}\right) \end{aligned} \quad (5)$$

Ideally, the probabilities in Equation 5 should be estimated from data, which would require the development of a large database of target ICG models corresponding to different speech signals. Since such a database is currently unavailable, we employ a heuristic approach to estimate the necessary probabilities.

Scusi? associates with each ICG I a confidence score which represents the probability of the ICG matching its parent UCG U and the context. This score is a heuristic approximation of Equation 5, and is used by *Scusi?* to rank competing ICGs. Throughout the ICG-generation process, *Scusi?* uses heuristics to boost or reduce the confidence score (Section 4). Three types of penalties (or rewards) can be applied to this score: small, medium and large. Clearly, the effect of applying a reward is to boost the confidence score, whereas a penalty reduces the score. The actual values for the penalties and rewards are empirically estimated.

4 Generating ICGs

The process of generating ICGs from a UCG and estimating their probability is carried out by Algorithm 1, which is composed of two main stages: *concept and relation postulation* (Steps 2–12), and *ICG construction* (Steps 13–19).

Concept and relation postulation. The algorithm proposes suitable candidate ICG concepts and relations from the knowledge base for each UCG concept and relation respectively (Step 3). This yields a list of ICG concepts L_{c_u} for each UCG concept c_u, and a list of ICG relations L_{r_u} for each UCG relation r_u. The algorithm then estimates the probability of the third and first factor of both rows in Equation 5 for each candidate concept and relation. The third factor represents the prior probability of an ICG concept or relation, and the first factor represents the match between an ICG concept or relation and its UCG counterpart.

The third factor is estimated in Step 6 of Algorithm 1. $\Pr(n^{ICG}|\mathcal{C})$, the prior probability of an ICG concept, is estimated using a function of its salience score. This score is obtained from dialogue history (concepts that were recently mentioned are salient). Relations are not influenced by the dialogue history, hence

Algorithm 1. Generating candidate ICGs given a UCG

```
Require: UCG $U$ comprising unknown concepts $c_u$ and relations $r_u$, $k_{\max} = 400$
 1: Initialize $\mathcal{I}$ {buffer to hold $k_{\max}$ ICGs}
 2: for all concepts $c_u$ (relations $r_u$) in the UCG $U$ do
 3:    Initialize $L_{c_u}$ ($L_{r_u}$) {list of suitable candidate domain concepts (relations)}
 4: end for
 5: for all concepts $c_k \in L_{c_u}$ (relations $r_k \in L_{r_u}$) do
 6:    Initialize the probability of $c_k$ to $\mathcal{F}$(salience score from dialogue history)
 7:    for all comparison functions $f$ associated with concept $c_k$ (relation $r_k$) do
 8:       Use $f$ to compare $c_u$ to $c_k$ ($r_u$ to $r_k$) $\rightarrow$ reward (or penalty) for $c_k$ ($r_k$)
 9:    end for
10:    Add concept $c_k$ (relation $r_k$) to the list $L_{c_u}$ ($L_{r_u}$)
11: end for
12: Sort list $L_{c_u}$ ($L_{r_u}$) in descending order of probability
13: for $i = 1$ to $k_{\max}$ do
14:    Generate the next best ICG $I_j$ by going down each list $L_{c_u}$ and $L_{r_u}$
15:    Add $I_j$ to the buffer $\mathcal{I}$
16:    Estimate $\Pr(I_j)$ {product of probabilities of all the concepts and relations}
17:    Check internal consistency of $I_j$ {the probability $\Pr(I_j)$ may change}
18:    Sort buffer $\mathcal{I}$ in descending order of the probability of ICG
19: end for
```

at present their prior probability is set to 1. In the future, their priors will be obtained from a corpus.

The first factor is estimated in Steps 7 and 8. $\Pr(n^{UCG}|n^{ICG})$ (and $\Pr(r^{UCG}|r^{ICG})$), the probability of the match between each concept (and relation) in the UCG and a candidate ICG concept (relation), is estimated by means of comparison functions that give penalties or rewards depending on the result of the comparison. At present, the comparisons are based on four attributes: cg-role (relation or concept), called (a list of domain names for a UCG concept or relation), part-of-speech, and relation (e.g., subject, object). For instance, given the UCG concept *cup*, the ICG concepts `mug01, ..., mug05` and `cup01, ..., cup04` match the UCG concept for the first three attributes (concepts have no relation attribute).

ICG construction. The algorithm generates candidate ICGs by selecting one candidate concept and relation from each of the lists L_{c_u} and L_{r_u} in descending order of probability (Step 14). For example, the ICG generated first is composed of the top (highest probability) element of each list, the next ICG contains the next element in one list and the top elements of the other lists, and so on.

Step 16 calculates the initial probability of an ICG: the product of the first factor in Equation 5 (the match between a UCG concept or relation and a candidate ICG concept or relation) and the third factor (the prior probability of concepts and relations). Step 17 estimates the second factor in Equation 5, which reflects the reasonableness of the relationships between neighbouring concepts and relations. This is done by comparing concepts and relations in the ICG to templates in the knowledge base. For example, the knowledge base contains a template entry for a possessive relationship between two concepts. Hence, ICGs that contain a possessive relationship between, say, an action concept and an object concept are penalized.

Algorithm 1 generates a buffer containing k_{max} ICGs the first time a UCG is expanded. It then successively returns the next ranked ICG candidate every time a new ICG is requested for that UCG.

5 Evaluation

A proper evaluation of the *Scusi?* dialogue module is a nontrivial task. Initially, the goal was to compare the performance of *Scusi?* to that of other dialogue systems. However, this was soon found to be infeasible as such systems are commonly too domain specific and difficult to obtain. Consequently, we have decided to evaluate the stand-alone interpretation performance of *Scusi?* for different utterance types.

The evaluation test set comprised 39 utterances: 17 declarative (e.g., "the book is on the table") and 22 imperative (e.g., "open the door"). The two groups of utterances were further sub-divided into short (3 words on average) and long (9 words on average). Each utterance was designed to test different aspects of *Scusi?*'s interpretation process. For example, some utterances refer to the same concepts with alternative words (e.g., "wash" and "clean"). Utterances referring to multiple possible referents were also considered. An example of this is the phrase "a book", since the knowledge base contains several books. Table 1 illustrates sample sentences used during testing. For each utterance in the test set, *Scusi?* was allowed to generate a maximum of 300 sub-interpretations (texts, parse trees, UCGs and ICGs).

An interpretation was deemed successful if it correctly represented the user's intention within the limitations of *Scusi?*'s knowledge base. The accuracy of all interpretations was judged manually by inspection on a case-by-case basis, and the successful ICG(s) were regarded as the gold standard (e.g., the ICGs referring to the different books in the knowledge base are considered gold standards for the utterance "get a book"). The last column in Table 1 shows the number of gold standard ICGs according to *Scusi?*'s knowledge base of 144 items: 24 relations and 120 concepts. In our experiments, we recorded whether the gold standard was found, and if so, the rank of the gold ICG on the basis of its probability.

To measure the impact of ASR error on *Scusi?*'s performance, we conducted two sets of experiments. The first round of experiments involved the original

Table 1. Example sentences used during system testing

Sentence	Length	Type	#gold
"get the bag"	short	imperative	2
"the book is on the table"	short	declarative	6
"open the door to the cupboard in the lounge"	long	imperative	1
"the large bag in the bathroom is white"	long	declarative	1
"Michael's box of books is on the table in the bedroom"	long	declarative	3

Table 2. *Scusi?* ranking performance for the full ASR and perfect ASR experiments

	Top	Top 3	Top 10	Not found	Average rank
Full ASR	26	32	36	3	1.8
Perfect ASR	31	36	38	1	1.7

ASR output data as returned by Microsoft Speech SDK 5.1. All tests were then repeated assuming a perfect ASR system; i.e., only the ASR output that corresponded to the gold standard text was used.

A summary of our results is given in Table 2. For the first experiment set, the gold ICG had the top or equal top rank in 26 out of 39 utterances. In 10 of the remaining 13 cases, the gold standard was located in the top 10 interpretations. About half of these cases were due to ASR and parsing ambiguity. Only in 3 cases *Scusi?* did not find the gold standard ICG. Near perfect interpretation performance was achieved for short declarative utterances, but the performance of *Scusi?* was otherwise invariant to the length and type of examined utterances. It is worth noting that in 6 cases, the ranking of the gold interpretation increased between the ASR output and the ICG, i.e., a lower ranked text produced a higher-ranked ICG. This demonstrates the effectiveness of maintaining multiple options, instead of selecting only the top-ranked alternative.

For the set of experiments that assume a perfect ASR system, the gold ICG had the top rank in 31 cases, and *Scusi?* did not find the gold ICG for one utterance only. In the remaining test cases, the gold ICG was ranked among the top 10 best hypotheses.

In terms of computational complexity, *Scusi?* required 18 expansion cycles on average before the gold ICG was generated in the first experiment. Furthermore, the mean number of ICGs generated prior to the gold ICG was approximately 3.6. This result demonstrates the effectiveness of our search algorithm.

6 Related Work

This research builds on the work described in [4], whose main focus was the consideration of multiple alternatives during the generation of interpretations. The main focus of this paper is the association of probabilities with interpretations of spoken utterances. In particular, we considered how *Scusi?* creates and ranks candidate ICGs given a UCG.

Many researchers have investigated probabilistic approaches to the interpretation of utterances in discourse systems. Young [5] introduced a probabilistic framework for dialogue systems based on Markov Decision Processes (MDPs) and reinforcement learning. However, the system requires user simulation models for the practical application of the technique. Miller et al. [6] consider a three stage system that maps utterances into meaning representation frames, where each stage is modelled as a statistical process. A similar probabilistic approach is examined in [7] and [8], but Pfleger et al. [8] use an empirical scoring function to interpret multi-modal input, rather than a fully probabilistic model. All these systems employ a semantic grammar for parsing. In contrast, *Scusi?* uses generic, syntactic tools, and incorporates semantic and domain-related information only in the final stage of the interpretation process. Knight et al. [9] compare the performance of a grammar-based dialogue system to that of a system based on a statistical language model and a robust phrase-spotting parser. The latter performs better for relatively unconstrained utterances by users unfamiliar with the system. The probabilistic approach and intended users of our system are in line with this finding.

Concept graphs have been employed for discourse interpretation in [10,11], but *Scusi?* differs from these systems in its use of UCGs as an intermediate stage that is independent from semantic and domain knowledge. From a processing point of view, Shankaranarayanan and Cyre [11] retain the first parse tree that supports an acceptable interpretation, rather than multiple parse trees. Although Sowa and Way [10] allow for multiple interpretations, their system prunes parses that fail semantic expectations. In contrast, *Scusi?* does not apply a filtering mechanism, and allows flawed candidates to undergo a deeper examination.

The current research closely resembles that in [12,13] in its integration of context-based expectations with alternatives obtained from spoken utterances. Horvitz and Paek [12] focus on higher level informational goals than those considered by *Scusi?*, using a single output produced by a parser as linguistic evidence for their discourse system. Unlike *Scusi?*, Gorniak and Roy [13] restrict the expected input, factoring in domain knowledge at the outset of the interpretation process. This allows their system to process expected utterances efficiently, but makes it difficult to interpret unexpected utterances.

7 Conclusion

We have described *Scusi?*, the speech interpretation module used in the DORIS project. *Scusi?* is a multi-stage interpretation system that maintains multiple options at each stage of the process, and ranks all (partial) interpretations based on estimates of their posterior probability. This work examined the techniques used by *Scusi?* to postulate and assess hypotheses regarding the meaning of a spoken utterance.

Two separate sets of experiments were conducted using original ASR output data, as well as data generated by a hypothetical perfect ASR. Our empirical evaluation shows that *Scusi?* performs well for declarative and imperative utterances of varying length, but ASR performance has a significant impact. The gold ICG model was ranked in the top three best sub-interpretations in most test cases. In addition, the performance of *Scusi?* was largely invariant to the length and type of utterances considered. In the near future, we will expand our data set, focusing on longer and more complex utterances, and investigate the interactions between the components of the system in order to further improve performance.

Acknowledgments. This research was supported in part by the ARC Centre for Perceptive and Intelligent Machines in Complex Environments. The authors also thank Eugene Charniak for his modifications to his probabilistic parser.

References

1. Sowa, J.F.: Conceptual Structures: Information Processing in Mind and Machine. Systems Programming Series. Addison-Wesley, Reading, MA (1984)
2. Gildea, D., Jurafsky, D.: Automatic labeling of semantic roles. Computational Linguistics 28(3), 245–288 (2002)
3. Niemann, M., George, S., Zukerman, I.: Towards a probabilistic, multi-layered spoken language interpretation system. In: IJCAI 2005. Proceedings of the fourth workshop on knowledge and reasoning in practical dialogue systems, Edinburgh, Scotland, pp. 8–15 (2005)
4. George, S., Niemann, M., Zukerman, I., Marom, Y.: Considering multiple options when interpreting spoken utterances. In: Proceedings of the Fifth IJCAI Workshop on Knowledge and Reasoning in Practical Dialogue Systems, Hyderabad, India, pp. 7–14 (2007)
5. Young, S.J.: Probabilistic methods in spoken-dialogue systems. Philosophical transactions of the Royal Society A 358(1769), 1389–1402 (2000)
6. Miller, S., Stallard, D., Bobrow, R., Schwartz, R.: A fully statistical approach to natural language interfaces. In: ACL 1996. Proceedings of the 34th Conference of the Association for Computational Linguistics, Santa Cruz, California, pp. 55–61 (1996)
7. He, Y., Young, S.: A data-driven spoken language understanding system. In: ASRU 2003. IEEE Workshop on Automatic Speech Recognition and Understanding, St. Thomas, US Virgin Islands, pp. 583–588 (2003)
8. Pfleger, N., Engel, R., Alexandersson, J.: Robust multimodal discourse processing. In: Proceedings of the 7th Workshop on the Semantics and Pragmatics of Dialogue, Saarbrücken, Germany, pp. 107–114 (2003)
9. Knight, S., Gorrell, G., Rayner, M., Milward, D., Koeling, R., Lewin, I.: Comparing grammar-based and robust approaches to speech understanding: a case study. In: EUROSPEECH 2001. 7th European Conference on Speech Communication and Technology, Aalborg, Denmark, pp. 1779–1782 (2001)
10. Sowa, J.F., Way, E.C.: Implementing a semantic interpreter using conceptual graphs. IBM Journal of Research and Development 30(1), 57–69 (1986)
11. Shankaranarayanan, S., Cyre, W.R.: Identification of coreferences with conceptual graphs. In: Tepfenhart, W.M., Dick, J.P., Sowa, J.F. (eds.) ICCS 1994. LNCS, vol. 835, pp. 45–60. Springer, Heidelberg (1994)
12. Horvitz, E., Paek, T.: A computational architecture for conversation. In: UM 1999. Proceedings of the seventh international conference on User modeling, Banff, Canada, pp. 201–210 (1999)
13. Gorniak, P., Roy, D.: Probabilistic grounding of situated speech using plan recognition and reference resolution. In: Proceedings of the 7th international conference on Multimodal interfaces, Torento, Italy, pp. 138–143 (2005)

Temporal Extensions to Defeasible Logic

Guido Governatori[1,⋆] and Paolo Terenziani[2]

[1] School of ITEE, The University of Queensland, Brisbane, Australia
[2] Università del Piemonte Orientale, Alessandria, Italy

Abstract. In this paper, we extend Defeasible Logic (a computationally-oriented non-monotonic logic) in order to deal with temporalised rules. In particular, we extend the logic to cope with durative facts, as well as with delays between the antecedent and the consequent of rules. We showed that the extended temporalised framework is suitable to model different types of causal relations which have been identified by the specialised literature. We also prove that the computational properties of the original logic are still retained by the extended approach.

1 Introduction

Non-monotonic logic has been proposed as an appropriate representation of commonsense reasoning. The notion of defeasible conclusion, i.e., a conclusion that can be revised if more evidence is provided, is at heart of commonsense reasoning and therefore of non-monotonic logic. A plethora of non-monotonic formalisms have been investigated, and a common issue is their high computational complexity. To obviate this problem Nute [20] proposed Defeasible Logic. Defeasible Logic (DL) is a rule based non-monotonic formalism that has been designed from beginning to be computationally feasible and easily implementable. Of course there is a trade-off between the expressive power of a logic and its computational complexity. Recently, a number of studies has shown that DL seems to be appropriate to reason in several application areas, ranging from modelling of contracts [13,6], legal reasoning [7,12], modelling of agents and agent societies [11,10], and applications to the Semantic Web [2]. An important finding of these investigations is that time is essential for an accurate representation of real world scenarios. While DL proved to be suitable to cope with most of the phenomena specific to the application domains, the treatment of the temporal issues have been by large ignored, and still is an open problem. Recently a few extensions of DL with time have been proposed [12,9,8].

[12] proposes an extension where each proposition is paired with a timestamp representing the time when the proposition holds. The aim of this work [12] is to study the notion of persistence. On the other hand [9] extends DL by attaching two temporal dimensions to each rules: the first to tell when a rule is deliberated (i.e., when a rule is created), and the second for the time of validity of a rule (i.e. when a rule can be used). In both cases only instantaneous events are considered and there are no explicit temporal relationships between the antecedent and the conclusion of a rule. On the other hand such issues are important not only to model many application domains but also to

⋆ Supported by the ARC DP0558854.

represent theoretical notions such as, for instance, causal relationships. [8] introduces intervals to model deadlines, this permits to have durative events, but there is no explicit temporal relationships between the antecedent and the conclusion of a rule.

In this paper we overcome such limitations by proposing an extension of DL to cope with durative events and with delays between antecedents (causes) and conclusions (effects) of rules in the logic. The proposed formalism also allows for an intensional representation of periodicity. We will first recall the basics of DL (Section 2), then we introduce and motivate the extended temporal formalism (Section 3). Section 4 illustrates the application of the formalism to model some types of causal relations. In Section 5 we formally describe the logic and its inference mechanism. We also show that the proposed extension preserves the computational behaviour of DL.

2 Defeasible Logic

Over the years DL [20,3] proved to be a flexible non-monotonic formalism able to capture different and sometimes incompatible facets of non-monotonic reasoning [4], and efficient and powerful implementations have been proposed [18,13].

Knowledge in DL can be represented in two ways: facts and rules. *Facts* are indisputable statements, represented either in form of states of affairs (literal and modal literal) or actions that have been performed. For example, “Tweety is a penguin” is represented by $Penguin(Tweety)$. A *rule* describes the relationship between a set of literals (premises) and a literal (conclusion), and we can specify how strong the relationship is. As usual rules allow us to derive new conclusions given a set of premises. For the strength of rules we distinguish between *strict rules*, *defeasible rules* and *defeaters*.

Strict rules, defeasible rules and defeaters are represented, respectively, by expressions of the form $A_1,\dots,A_n \to B$, $A_1,\dots,A_n \Rightarrow B$ and $A_1,\dots,A_n \leadsto B$, where $A_1,\dots,A_n$ is a possibly empty set of prerequisites (causes) and B is the conclusion (effect) of the rule. We only consider rules that are essentially propositional, i.e., rules containing free variables are interpreted as the set of their ground instances.

Strict rules are rules in the classical sense: whenever the premises are indisputable then so is the conclusion. Thus they can be used for definitional clauses. An example of a strict rule is “Penguins are birds”, formally: $Penguin(X) \to Bird(X)$.

Defeasible rules are rules that can be defeated by contrary evidence. An example of such a rule is “Birds usually fly”: $Bird(X) \Rightarrow Fly(X)$. The idea is that if we know that X is a bird, then we may conclude that X flies *unless there is other evidence suggesting that she may not fly*.

Defeaters are a special kind of rules. They are used to prevent conclusions not to support them. For example: $Heavy(X) \leadsto \neg Fly(X)$. This rule states that if something is heavy then it might not fly. This rule can prevent the derivation of a “fly” conclusion. On the other hand it cannot be used to support a “not fly” conclusion.

DL is a “skeptical” non-monotonic logic, meaning that it does not support contradictory conclusions. Instead DL seeks to resolve conflicts. In cases where there is some support for concluding A but also support for concluding $\neg A$, DL does not conclude either of them (thus the name “skeptical”). If the support for A has priority over the support for $\neg A$ then A is concluded. No conclusion can be drawn from conflicting rules in DL unless these rules are prioritised. The *superiority relation* among rules is used to define priorities among rules, that is, where one rule may override the conclusion of another rule. For example, given the defeasible rules

$$r : \mathit{Bird}(X) \Rightarrow \mathit{Fly}(X) \qquad r' : \mathit{Penguin}(X) \Rightarrow \neg \mathit{Fly}(X)$$

which contradict one another, no conclusive decision can be made about whether a Tweety can fly or not. But if we introduce a superiority relation $\succ$ with $r' \succ r$, then we can indeed conclude that Tweety cannot fly since it is a penguin.

We now give a short informal presentation of how conclusions are drawn in DL. Let D be a theory in DL (i.e., a collection of facts, rules and a superiority relation). A *conclusion* of D is a tagged literal and can have one of the following four forms:

$+\Delta q$ meaning that q is definitely provable in D (i.e., using only facts and strict rules).
$-\Delta q$ meaning that we have proved that q is not definitely provable in D.
$+\partial q$ meaning that q is defeasibly provable in D.
$-\partial q$ meaning that we have proved that q is not defeasibly provable in D.

Strict derivations are obtained by forward chaining of strict rules, while a defeasible conclusion p can be derived if there is a rule whose conclusion is p, whose prerequisites (antecedent) have either already been proved or given in the case at hand (i.e., facts), and any stronger rule whose conclusion is $\neg p$ has prerequisites that fail to be derived. In other words, a conclusion p is derivable when:

- p is a fact; or
- there is an applicable strict or defeasible rule for p, and either
  - all the rules for $\neg p$ are discarded (i.e., are proved to be not applicable) or
  - every applicable rule for $\neg p$ is weaker than an applicable strict[1] or defeasible rule for p.

The formal definitions of derivations and the proof conditions or inference rules for the extension of the logic with the features we are going to describe in the next section will be presented in Section 5.

3 Modelling Temporal Rules

In the following, we first introduce the temporal dimension within the rules and we provide an informal high-level representation of the rules. We then refine it in the next section, showing how high-level rules can be actually modelled within DL. In the discussion below, we use the causal metaphor to explain the relationship between the antecedents and the consequent of the rules. Such a metaphor seems to us quite natural, especially in case rules are interpreted within a temporal framework. We will further investigate the relationships between our temporal rules and different definitions of causation in the literature in Section 4.

We represent rules of the general form:

$$a_1 : d_1, \ldots, a_n : d_n \Rightarrow^d b : d_b$$

[1] Notice that a strict rule can be defeated only when its antecedent is defeasibly provable.

Such rules have the following general intuitive interpretation:

- In the rule, all events are represented as a pair $e : d$, where e is the identifier of the event and $d \in \mathbb{N}, d \geq 1$ is the duration of the event. As special cases, instantaneous events are modelled by $d = 1$, while persistent events by $d = +\infty$.
- $a_1, \dots, a_n$ are the "causes". They can start at different points in time.
- b is the "effect".
- "d" ($d \in \mathbb{Z}$) is the *exact* delay between "causes" and "effect". "d" cannot be infinite.

The above definition represents a whole general schema of rules, depending on the choice of the event endpoints which are related by the delay relation. One may consider the following alternative interpretations (where "last" means "temporally last"):

1. d is the delay between the start of the last cause and the beginning of the effect;
2. d is the delay between the end of the last cause and the beginning of the effect (in such an interpretation the duration of the causes cannot be infinite).

Other alternatives are technically possible, but seem to us not very motivated from the applicative point of view. For instance, considering d the delay between the start (or end) of the first cause and the start of the effect seems to us quite counter-intuitive. In fact, if all $a_1, \dots a_n$ are explicitly necessary to trigger the rule, it seems quite unnatural that the rule is then triggered as soon as the first of them starts (ends) to hold.

Then to trigger a rule, different conditions have to be satisfied, depending on the chosen interpretation. For instance, let us consider the intuitive triggering conditions for case (2) above (and supposing that all durations are finite).

1. we must be able to prove each a_i for *exactly* d_i consecutive time intervals, i.e., $\forall i \exists t_0, t_1, \dots t_{d_i}, t_{d_{i+1}}$ consecutive time intervals such that we can not prove a_i at time t_0, we can prove a_i at times $t_1, \dots t_{d_i}$, and we cannot prove a_i at time $t_{d_{i+1}}$
2. Let $t_a^{\max} = \max\{t_{d_j}\} 1 \leq j \leq n$ be the last time when the latest cause can be proven.
3. b can be proven for d_b instants starting from time $t_a^{\max} + d$, i.e., it can be proven in all the time instants between $t_a^{\max} + d - 1$ and $t_a^{\max} + d + d_b + 1$.

4 Causation

Since causation "is common in scientific thinking" and "is dominant in everyday commonsense thinking" [23], it has been widely studied by many AI approaches, mainly aiming to provide computational models of causal reasoning. In particular, many approaches have focused on the relationships between time and causation, starting, e.g., from the pioneering reified logics by McDermott [19] and Allen [1], and by the approaches by Shoham [23], Konolidge [16] and Sandewall [22], that underlined the defeasible character of causal connection and\or stressed the strict relationship between causal reasoning and non-monotonic reasoning (e.g., to deal with change).

Moreover, several approaches focused specifically on the temporal relationships between causes and effects. The problem of determining such temporal constraints is still an open one: for instance, there seems to be no agreement on whether "backward" causation (i.e., causal relations in which the effects can precede the causes) is possible or not (for a collection of accounts on causation by many of the outstanding contemporary philosophers see Sosa [24]).

While domain-dependent approaches in specific application areas have devised restrictive “ad-hoc” definitions (see, e.g., [15] in the medical field), many approaches only identified very general temporal constraint (namely, the constraint that the beginning of the effect cannot precede the beginning of the causes [23,19,1]). An interesting proposal has been devised by Reiger and Grinberg [21], who distinguished between different types of causation depending on the different temporal relations they impose between causes and effects. In particular, in a “One-shot” causal relation the presence of the cause is required only momentarily in order to allow the effect to begin (e.g., to kick a ball causes the ball to move), while in a “Continuous” causal relation the presence of the cause is required in order to begin and sustain the effect (e.g., to run causes the production of lactic acid). Other researchers have followed such a line of research. For instance, within the CYC ontology, Guha and Lenat [14] have identified also “Mutually sustaining” causal relations (e.g., “poverty and illiteracy were mutually sustaining problems plaguing America all during the 1970s”), in which each bit of the cause causes a slightly later bit of the effect, and viceversa (i.e., causes and effects are co-temporal). Analogously, Terenziani and Torasso [25] have introduced also “Culminated Event Causation”, to model causal relationships where the cause is an accomplishment, whose culmination must be reached in order to enforce the effect to start (e.g., switching the light on is an activity with a goal, and its effect –having the light on– is reached with the culmination of such an activity).

The definition of rules we have proposed in Section 3 can model causation, by interpreting the antecedents of the rules as causes and the consequents as effects. Of course, the defeasible non-monotonic character of causation is naturally coped with by the underlying DL. Moreover, the definition of rules we have provided is very general, so that all the above different types of temporal constraints between causes and effects can be captured (as special cases of the general rule) by suitably restricting the values of delay and durations in the rule. For the sake of simplicity, we exemplify such a claim considering simple rules with just one antecedent (cause), i.e., rules of the form $a : t_a \Rightarrow^d b : t_b$ and considering the interpretation (2) for delays (the case of interpretation (1) is analogous):

– We can model “backward” causation, by considering a negative delay between causes and effects, so that $0 > t_a + d$;
– “one-shot” causation can be modelled with $0 \leq t_a + d$ and $0 < t_b + d$;
– “continuous” causation can be modelled with $0 \leq t_a + d$ and $0 \geq t_b + d$;
– “mutually sustaining” causation can be modelled with $0 = t_a + d$, and $0 = t_b + d$;
– “culminated event causation” can be modelled simply by imposing $0 \leq d$.

5 Temporalised Defeasible Logic

As usual with non-monotonic reasoning, we have to specify 1) how to represent a knowledge base and 2) the inference mechanism used to reason with the knowledge base. The language of temporal Defeasible Logic comprises a (numerable) set of atomic propositions $Prop = \{p, q, \dots\}$, a discrete totally ordered set of instants of time $\mathscr{T} = \{t_1, t_2, \dots\}$ and the negation sign $\neg$.

We supplement the usual definition of literal (an atomic proposition in *Prop* or the negation of it), with two notions of temporalised literals, one notion to indicate when a literal holds (i.e., $p@t$, where $t \in \mathscr{T}$) and the duration of an event (i.e., $p : d$, $d \in \mathbb{N}$). Intuitively $p@t$ means that p holds at time t, and $p : d$ means that p holds for d time instants. Given a (temporalised) literal p, $\sim p$ denotes the complement of p, that is $\sim p = \neg q$ if $p = q$, $q \in Prop$, and $\sim p = q$ if $p = \neg q$ for some $q \in Prop$.

Formally a *rule* r consists of its *antecedent* (or *body*) $A(r)$ ($A(r)$ may be omitted if it is the empty set) which is a finite set of temporal literals of the form $p : d$, an arrow, and its *consequent* (or *head*) $C(r)$ which is a temporal literal of the form $p : d$. The arrow of a rule r is labelled by a natural number d, the delay of r; we will use $\delta(r)$ to denote the delay of r. Given a set R of rules, we denote the set of strict rules in R by R_s, the set of strict and defeasible rules in R by R_{sd}, the set of defeasible rules in R by R_d, and the set of defeaters in R by R_{dft}. $R[q]$ denotes the set of rules in R with consequent $q : d$.

A (temporalised) defeasible theory is a structure $(\mathscr{T}, t_0, F, R, \succ)$ where: $\mathscr{T} = \{t_1, t_2, \dots\}$ is a discrete totally ordered set of instants of time; $t_0 \in \mathscr{T}$ is the reference point (or origin) of $\mathscr{T}$; F is a finite set of facts, i.e., of temporalised literals of form $p@t$; R is a finite set of rules; $\succ$, the superiority relation, is a binary relation over R such that the transitive closure of it is acyclic.

A *conclusion* of D is a tagged literal and can have one of the following four forms:

$+\Delta q@t$ meaning that q is definitely provable, at time t, in D (i.e., using only facts and strict rules).
$-\Delta q@t$ meaning that we have proved that q is not definitely provable, at time t, in D.
$+\partial q@t$ meaning that q is defeasibly provable, at time t, in D.
$-\partial q@t$ meaning that we have proved that q is not defeasibly provable, at time t, in D.

The proof tags tell us whether it is possible to prove a conclusion at a specific time and the strength or confidence of the conclusion. Provability is based on the concept of a *derivation* (or proof) in D. A derivation is a finite sequence $P = (P(1), \dots, P(n))$ of tagged literals satisfying the proof conditions (which correspond to inference rules for each of the kinds of conclusion). $P(1..n)$ denotes the initial part of the sequence P of length n. Before introducing the proof conditions for the proof tags relevant to this paper we provide some auxiliary notions. From now on we will use # as a variable ranging over $\{\Delta, \partial\}$. All the remaining definition refer to a derivation P.

Definition 1. *A temporalised literal $p : d$ is* #-active *at t iff 1) $-\#p@t-1 \in P(1..n)$, 2) $\forall t', t-1 < t' \leq t+d$, $+\#p@t' \in P(1..n)$, and 3) $-\#p@t+d+1 \in P(1..n)$. If $d = +\infty$ then the last condition does not apply.*

Intuitively that a literal $a : d$ is #-active at t means that the event a started at t and lasted for d instants. Based on the notion of #-active time we can define the activation time of a rule. The idea is that at the activation time a rule becomes active and trigger its own conclusion.

Definition 2. *Let r be a rule. A set $AT(r) \subset \mathscr{T}$ is an* #-activation set of *r iff*

1. *there is a surjective mapping ρ from $A(r)$ to $AT(r)$, and*
2. *$\forall a_i : d_i \in A(r)$, $a_i : d_i$ is #-active at $\rho(a_i : d_i)$.*

t is a #-activation time for *r iff there is a #-activation set $AT(R)$ s.t. $t = \max(AT(r))$.*

Since it is possible that a temporal literal $a : d$ is active at different times, there are many activation times for a rule. The activation time for a rule is used to determine when the rule fires and when the effect of the rule (its consequent) will be produced. The formal relationship between the activation time, the delay and the start of the effect is given in the following definition.

Definition 3. *A rule r is* #-applicable *at t iff $t - \delta(r)$ is a #-activation time for r.*

Finally we give a definition of the potential duration of the effect of a rule.

Definition 4. *A rule $r \in R_{sd}[p : d_r]$ is (t,t')*-effective *iff r is ∂-applicable at t, $t' - t < d_r$.*

Let us concentrate on the conditions that prevent a rule to be triggered. Again we start from temporal literals.

Definition 5. *A temporal literal $a : d$ is* #-passive *at time t iff $\exists t', t \le t' < t + d + 1$ such that $-\#a@t$.*

This means that a temporal literal is passive at time t if there is an instant in the interval $t, t + d$ where the literal does not hold.

Definition 6. *A rule r is* #-discarded *at time t iff $\exists a_i : d_i \in A(r)$ such that $\forall t' < t - \delta(r)$, $a_i : d_i$ is #-passive at t.*

We define now a condition to determine whether an applicable rule does not produce its effect since the derivation of the effect is blocked by a stronger rules preventing the effect.

Definition 7. *A rule $r \in R_{sd}[p : d_r]$* defeats *a rule $s \in R[\sim p : d_s]$ at time t iff 1) s is ∂-applicable at t, and 2) r is (t',t)-effective, and 3) $r \succ s$.*

The effect of a rule, i.e., its consequent, has a duration, so the effect can persist. The next definition captures the idea of persistence.

Definition 8. *A rule $r \in R_{sd}[p : d_r]$* persists *at time t iff*

- *r is ∂-applicable at t', $t - d_r < t' < t$ and $+\partial p@t' \in P(1..n)$ and*
- *$\forall t'', t' < t'' \le t$, $\forall s \in R[\sim p]$ either*
  - *s is ∂-discarded at t'' or*
  - *if s is (t''',t'')-effective, then $\exists v \in R_{sd}[p]$ such that v defeats s at t'''.*

We are now ready to give the proof conditions for $+\Delta$, $-\Delta$, $+\partial$ and $-\partial$.

Proof condition for $+\Delta$
If $+\Delta p@t = P(n+1)$ then
1) $p@t \in F$ or
2) $\exists r \in R_s[p]$, r is Δ-applicable at t, or
3) $\exists r \in R_s[p]$, r is Δ-applicable at t' and $t - \delta(r) < t'$.

Proof condition for $-\Delta$
If $-\Delta p@t = P(n+1)$ then
1) $p@t \notin F$ and
2) $\forall r \in R_s[p]$, r is Δ-discarded at t and
3) $\forall r \in R_s[p]$, if r is Δ-applicable at t', then $t' + \delta(r) < t$.

The conditions above are essentially the normal conditions for definite proofs in DL. $+\Delta$ describes monotonic derivations using forward chaining. The only things to notice are that: (i) each literal $a_i : d_i$ in the antecedent must be definitely proved for exactly d_i

times; (ii) clause (3) is a persistence clause and allows us to derive p at time t if we have previously proved p at time t' and the distance between t and t' is less than the duration of a strict rule for p applicable at t'.

To prove that a definite conclusion is not possible we have to show that all attempts to give a definite proof of the conclusion fail. In particular we have to check that all strict rules for $\sim p$ will not produce an effect at t.

Proof condition for $+\partial$
If $+\partial p@t = P(n+1)$ then
1) $+\Delta p@t \in P(1..n)$ or
2).1) $-\Delta{\sim}p@t \in P(1..n)$ and
  .2) $\exists r \in R_{sd}[p]$ such that r persists at t or
    r is ∂-applicable at t, and
  .3) $\forall s \in R[\sim p]$ either
    .1) s is ∂-discarded at t or
    .2) if s is (t',t)-effective,
      then $\exists v \in R_{sd}[p]$ v defeats s at t'.

Proof condition for $-\partial$
If $-\partial p@t = P(n+1)$ then
1) $-\Delta p@t \in P(1..n)$ and
2).1) $+\Delta{\sim}p@t \notin P(1..n)$ or
  .2) $\forall r \in R_{sd}[p]$ r does not persist at t and
    is ∂-discarded at t, or
  .3) $\exists s \in R[\sim p]$ such that
    .1) s is (t',t)-effective, and
    .2) $\forall v \in R_{sd}[p]$, v does not defeat s at t'.

Clauses (1) and (2.1) of $+\partial$ are the usual clauses of standard DL only relativised to a time instant. In particular clause (1) allows us to weaken a definite conclusion to a defeasible conclusions. Clause (2.1) ensures consistency of the logic, i.e., a stronger proof for the opposite of what we want to prove is not possible. Clause (2.2) requires that at least one rule for p can indeed produce p at time t. Thus we have to subtract the delay from t to see whether the last cause of the rule started at $t-d$. Clause (3.1) is intended to check that each rule for the negation of the conclusion we want to prove is either discarded at t. i.e., one of the causes of this rule has been proved not to hold for the right duration. Otherwise, an applicable rule for $\sim p$ producing the effect at t must be blocked by a stronger rule for p. The stronger rule should block the beginning of the effect $\sim p$.

Let us consider a theory where $F = \{a@0, b@5, c@5\}$ and with the rules

$$(r_1)\ a:1 \Rightarrow^{10} p:10, \qquad (r_2)\ b:1 \Rightarrow^{7} \neg p:5, \qquad (r_3)\ c:1 \Rightarrow^{8} p:5$$

where $r_3 \succ r_2$. Based on this theory we can derive $+\partial p@10$. r_1 is ∂-applicable and r_2 is ∂-discarded. At time 11 we can derive p since r_1 persists here. At time 12 r_2 is no longer discarded, so r_2 interrupts the effect of r_1. Since we have no means to determine which reason prevails we derive both $-\partial p@12$ and $-\partial\neg p@12$. At 13 rule r_3 becomes active and defeats r_2, thus from 13 to 18 we can derive p. However, r_3 terminates its effect at 18, and r_1, whose duration would ensure that p persists till 20, cannot be used to derive p since the effect of r_1 was terminated by r_2 at time 12.

To prove that a defeasible conclusion is not possible we have to show that all attempts to give a defeasible proof of the conclusion fail. In particular we have to check that all rules for $\sim p$ will not produce an effect at t, and if they were to produce it there are some active rules for the negation not weaker than rules for the conclusion and the rules are effective at the appropriate instants.

Maher [17] proved that for standard DL the extension of a defeasible theory can be computed in time linear to the size of the theory (where the size of the theory is in function of the number of symbols and rules in it). Here we can preserve the complexity results for theories without backward causation.

Theorem 1. *Let D be a temporalised defeasible theory without backward causation. Then the extension of D from t_0 and $t \in \mathscr{T}$ (i.e., the set of all consequences based on the signature of D derivable from t_0 and t) can be computed in time linear to the size or theory. i.e.. $O(|Prop| \times |R| \times t)$.*

Proof (sketch). The idea is to consider snapshots of the theory at every single instant t' from t_0 and t to derive the conclusions that hold at time t'. The rules are linearly reduced to the language of [12]. In particular temporal literals in the antecedent are understood as a set of temporal literals with timestamps (e.g., for t' a durative literal such as $a : 2$, will be expanded to $\{a@t' - 1, a@t'\}$) and literal with infinite duration are dealt with persistent rules. After these steps we can apply the technique of [11] to compute in linear time the extension of the resulting equivalent theory.

Notice that, although the rules contain temporal constraints that are related to STP (Simple Temporal Problem [5]), in this paper, no "classical" form of temporal reasoning (in the sense of temporal constraint propagation [5]) is required. Let us consider the following simple set of rules, involving delays between instantaneous events:

$$\{a : 1 \Rightarrow^{10} b : 1,\ b : 1 \Rightarrow^{10} c : 1,\ a : 1 \Rightarrow^{10} c : 1\}$$

In a "classical" temporal constraint environment, such as in the STP framework, the rules in the set would be interpreted as a *conjunction*, so that constraint propagation could be used in order to infer from the first two dependencies that (in case the interpretation (2) is chosen for the delay) the delay between a and c is 21, so that the whole set of constraints is not consistent. However, in the present framework, the above set is simply interpreted as a set of *alternative rules*. In such a context, one can consistently say that c can be caused by a through the first two rules (where the delay between a and c is 21), or directly through the third rule (where the delay is 10).

6 Conclusion and Future Extensions

In this paper, we have proposed a temporal extension of DL in order to deal with temporalised durative facts, and with (possible) delays between antecedents and consequences of the rules. The extension we propose

- increases the expressiveness of the logic, as needed to deal with temporal phenomena in several application domains, and, in particular, with different types of causal relationships;
- maintains the computational properties of the original logic. Indeed, the temporal extension has been carefully shaped in order to maintain (in case backward causation rules are not used) the linear complexity of the original logic.

As future work, we aim at analysing the complexity of the extended logic, in case also backward causation rules (i.e., rules in which the consequent can start before the antecedent) are taken into account. The addition of backward rules makes complexity significantly harder, since it requires to navigate the timeline also backward. Our conjecture is that, in such a case, the complexity becomes quadratic on the overall span of time determined by the period of repetition of all the cycles of the rules in the theory.

References

1. Allen, J.F: Towards a general theory of action and time. Artificial Intelligence 23, 123–154 (1984)
2. Antoniou, G.: Nonmonotonic rule system on top of ontology layer. In: Bergmann, R. (ed.) Experience Management. LNCS (LNAI), vol. 2432, pp. 394–398. Springer, Heidelberg (2002)
3. Antoniou, G., Billington, D., Governatori, G., Maher, M.J.: Representation results for defeasible logic. ACM Transactions on Computational Logic 2(2), 255–287 (2001)
4. Antoniou, G., Billington, D., Governatori, G., Maher, M.J., Rock, A.: A family of defeasible reasoning logics and its implementation. In: Proc. ECAI 2000, pp. 459–463 (2000)
5. Dechter, R., Meiri, I., Pearl, J.: Temporal constraint networks. Artificial Intelligence 49, 61–95 (1991)
6. Governatori, G.: Representing business contracts in RuleML. International Journal of Cooperative Information Systems 14(2-3), 181–216 (2005)
7. Governatori, G., Dumas, M., ter Hofstede, A.H.M., Oaks, P.: A formal approach to protocols and strategies for (legal) negotiation. In: Proc. ICAIL 2001, pp. 168–177 (2001)
8. Governatori, G., Hulstijn, J., Riveret, R., Rotolo, A.: Characterising deadlines in temporal modal defeasible logic. In: Orgun, M.A., Thorton, J. (eds.) Australian AI 2007. LNCS(LNAI), vol. 4830, pp. 486–496. Springer, Heidelberg (2007)
9. Governatori, G., Padmanabhan, V., Antonino, R.: Rule-based agents in temporalised defeasible logic. In: Yang, Q., Webb, G. (eds.) PRICAI 2006. LNCS (LNAI), vol. 4099, pp. 31–40. Springer, Heidelberg (2006)
10. Governatori, G., Rotolo, A.: Defeasible logic: Agency, intention and obligation. In: Lomuscio, A.R., Nute, D. (eds.) DEON 2004. LNCS (LNAI), vol. 3065, pp. 114–128. Springer, Heidelberg (2004)
11. Governatori, G., Rotolo, A., Padmanabhan, V.: The cost of social agents. In: Proc. AAMAS 2006, pp. 513–520 (2006)
12. Governatori, G., Rotolo, A., Sartor, G.: Temporalised normative positions in defeasible logic. In: Proc ICAIL 2005, pp. 25–34 (2005)
13. Grosof, B.N.: Representing e-commerce rules via situated courteous logic programs in RuleML. Electronic Commerce Research and Applications 3(1), 2–20 (2004)
14. Guha, R.V., Lenat, D.B.: Building Large Knowledge Based Systems. Addison-Wesley, Reading (1991)
15. Hamlet, I., Hunter, J.: Representation of time in medical expert systems. LNMI 33, 112–119 (1987)
16. Konolidge, K.: Using default and causal reasoning in diagnosis. Annals of Mathematics and Artificial Intelligence 11, 97–135 (1994)
17. Maher, M.J.: Propositional defeasible logic has linear complexity. Theory and Practice of Logic Programming 1(6), 691–711 (2001)
18. Maher, M.J., Rock, A., Antoniou, G., Billington, D., Miller, T.: Efficient defeasible reasoning systems. International Journal of Artificial Intelligence Tools 10(4), 483–501 (2001)
19. McDermott, D.: A temporal logic for reasoning about processes and plans. Cognitive Science 6, 101–105 (1982)
20. Nute, D.: Defeasible logic. In: Handbook of Logic in Artificial Intelligence and Logic Programming, vol. 3, pp. 353–395. Oxford University Press, Oxford (1994)
21. Rieger, C., Grinberg, M.: The declarative representation and simulation of causality in physical mechanisms. In: Proc. IJCAI 1977, pp. 250–256 (1977)
22. Sandewall, E.: Features and Fluents. Oxford Scientific Publications, Oxford (1994)
23. Shoham, Y.: Reasoning About Change. MIT Press, Cambridge (1987)
24. Sosa, E.: Causation and Conditionals. Oxford university Press, Oxford (1975)
25. Terenziani, P., Torasso, P.: Time, action-types and causation: an integrated analysis. Computaional Intelligence 11(3), 529–552 (1995)

Characterising Deadlines in Temporal Modal Defeasible Logic

Guido Governatori[1], Joris Hulstijn[2], Régis Riveret[3], and Antonino Rotolo[3]

[1] School of ITEE, The University of Queensland, Australia
guido@itee.uq.edu.au
[2] Vrije Universiteit, Amsterdam, The Netherlands
jhulstijn@feweb.vu.nl
[3] CIRSFID, University of Bologna, Italy
antonino.rotolo@unibo.it, regis.riveret@unibo.it

Abstract. We provide a conceptual analysis of several kinds of deadlines, represented in Temporal Modal Defeasible Logic. The paper presents a typology of deadlines, based on the following parameters: deontic operator, maintenance or achievement, presence or absence of sanctions, and persistence after the deadline. The deadline types are illustrated by a set of examples.

1 Introduction

Many normative rules in applications of artificial intelligence are concerned with deadlines. Bills must be paid before the end of the month; books need to be returned before a due date; products must be delivered within ten working days. In general we can say that a deadline combines a specification of a time point or condition δ, with the obligation to achieve some state of affairs φ *before* condition δ arrives.

Because deadlines combine time and obligations, they are naturally studied by a combination of temporal logic with deontic logic. Various papers use a combination of branching time temporal logic (CTL) with Standard Deontic Logic [5,4,3]. In such logics, the meaning of a deadline clause, i.e., that φ should occur before δ, can be expressed as an obligation which involves the 'until'-operator: $\mathrm{OBL}(\varphi \mathrm{U} \delta)$. This means that situations in which δ becomes true while φ has not been achieved, are considered to be a violation according to some normative system.

However, such a characterisation of deadlines does not indicate what will happen *after* the deadline. Is it required to still complete φ even after the deadline, as for late payments? Or is it impossible or unnecessary to complete φ after the deadline, as for the late delivery of a wedding cake? Is an explicit sanction enforced after the deadline, or is the sanction simply that achieving one's goal has become impossible?

Since deadlines can have different functions, it is likely that several notions of deadline can be distinguished. In this paper we therefore want to provide a conceptual analysis of various kinds of deadlines, and to provide a formal characterisation. Such a characterisation should make it easier to have intelligent systems reason about systems of deadline clauses, to find out for example whether they are mutually consistent, or how a course of action can be planned that meets all deadline clauses; compare the motivation given by [5].

To express the characterisation, we choose a Temporal Modal Defeasible Logic which combines the deontic modalities of obligation and permission with temporal intervals. Many normative rules allow for exceptions. Without defeasibility, it would be impossible to distinguish exceptions from violations. Moreover, a defeasible logic allows for a modular representation of deadline clauses, in which a clause may for example override earlier clauses, or override clauses issued by a lesser authority.

Usually, logic papers focus on the logical properties of a newly developed logic. In this paper, however, we would like to stress the conceptual issues of modelling temporal normative rules. It is necessary to understand deadlines, for example, for the implementation of procurement procedures, or for the management of automated service level agreements [10]. This research is part of a larger research effort to investigate the use of defeasible logic for reasoning with legal information [9,8].

The remainder of the paper is structured as follows. In Section 2 we define the Temporal Modal Defeasible Logic to be used in our characterisation. In Section 3 we provide a characterisation of different types of deadlines, and provide template formulas in the logic, to represent these types.

2 Temporal Modal Defeasible Logic

The logical framework of this paper is an interval based variant of Temporal Modal Defeasible Logic (TMDL), which is an extension of Defeasible Logic [2] to capture the deontic modalities obligation and permission and some aspects of time. Other variants of TMDL have proved useful in modelling normative reasoning [9,8] and cognitive agents [7].

We assume a linear discrete bounded set $\mathscr{T}$ of points of time termed ‘instants’ and over it the temporal order relation $>\subseteq \mathscr{T}\times\mathscr{T}$. We define intervals $[t_1,t_2]$ as sets of instants between the instants t_1 and t_2. Our framework is meant to incorporate temporal parameters into a non-monotonic model for deontic reasoning. It is not our intention to provide a general approach to temporal reasoning, nor an extensive analysis of operations and relations over time intervals, as e.g. [1].

A TMDL theory consists of a linear discrete bounded set of instants, a set of *facts* or indisputable statements, a set of *rules*, and a *superiority relation* $\succ$ among rules specifying when a rule may override the conclusion of another rule. A *strict rule* is an expression of the form $\varphi_1,\ldots,\varphi_n \rightarrow \psi$ such that whenever the premises are indisputable so is the conclusion. A *defeasible rule* is an expression of the form $\varphi_1,\ldots,\varphi_n \Rightarrow \psi$ whose conclusion can be defeated by contrary evidence. An expression $\varphi_1,\ldots,\varphi_n \rightsquigarrow \psi$ is a *defeater* used to defeat some defeasible rules by producing evidence to the contrary.

Definition 1 (Language). Let $\mathscr{T}$ be a linear discrete bounded ordered set of instants of time, in which the minimal unit is u and the lower and higher boundaries of $\mathscr{T}$ are denoted respectively by *min* and *max*. Let Prop be a set of propositional atoms, let Mod be a set of modal operators $\{\mathrm{OBL},\mathrm{PERM}\}$, and let Lab be a set of labels of rules. The sets below are defined as the smallest sets closed under the following rules:

Literals $\mathrm{Lit} = \mathrm{Prop}\cup\{\neg p | p\in \mathrm{Prop}\}$
Modal Literals $\mathrm{ModLit} = \{X\,l\,, \neg X\,l | X\in \mathrm{Mod}, l\in \mathrm{Lit}\}$

Intervals $\text{Inter} = \{T = [t_1,t_2] | [t_1,t_2] = \{x | t_1 \leq x \leq t_2\}, t_1,t_2 \in \mathscr{T}\}$
Temporal Literals $\text{TLit} = \{l{:}T | l \in \text{Lit},\ T \in \text{Inter}\}$
Temporal Modal Literals $\text{TModLit} = \{\varphi{:}T | \varphi \in \text{ModLit},\ T \in \text{Inter}\}$
Rules $\text{Rule}_s = \{(r \quad \varphi_1,\ldots,\varphi_n \rightarrow \psi) | r \in \text{Lab},\ \varphi_1,\ldots,\varphi_n,\psi \in \text{TLit} \cup \text{TModLit}\}$
$\text{Rule}_d = \{(r \quad \varphi_1,\ldots,\varphi_n \Rightarrow \psi) | r \in \text{Lab},\ \varphi_1,\ldots,\varphi_n,\psi \in \text{TLit} \cup \text{TModLit}\}$
$\text{Rule}_{dft} = \{(r \quad \varphi_1,\ldots,\varphi_n \leadsto \psi) | r \in \text{Lab},\ \varphi_1,\ldots,\varphi_n,\psi \in \text{TLit} \cup \text{TModLit}\}$
$\text{Rule} = \{\text{Rule}_s \cup \text{Rule}_d \cup \text{Rule}_{dft}\}$
Superiority Relations $\text{Sup} = \{s \succ r | s,r \in \text{Lab}\}$

We shall use some abbreviations: $A(r)$ denotes the set $\{\varphi_1,\ldots,\varphi_n\}$ of *antecedents* of the rule r, and $C(r)$ to denote the *consequent* ψ of the rule r. We also use $\text{Rule}[\varphi]$ for those rules in Rule whose consequent is φ. If γ is a literal, $\sim\gamma$ denotes the complementary literal: if q is the positive literal p then $\sim q$ is $\neg p$; and if q is $\neg p$, then $\sim q$ is p.

The resulting formalism permits to have expressions such as $\psi{:}[0,20]$ and $\text{OBL}\varphi{:}[1,10]$: the former says that ψ holds from 0 to 20, the latter that the obligation that φ applies from 1 to 10. Note that we allow for 'punctual intervals', i.e. intervals of the form $[t,t]$. As a notational convention, we simply write $\varphi{:}t$ instead of $\varphi{:}[t,t]$.

Definition 2 (TMDL Theory). A TMDL theory is a structure $D = (\mathscr{T}, F, R, \succ)$ where $\mathscr{T}$ a linear discrete bounded totally ordered set of instants of time, $F \subseteq \text{TLit} \cup \text{TModLit}$ is a finite set of facts, $R \subseteq \text{Rule}$ is a finite set of rules such that each rule has a unique label, $\succ \subseteq \text{Sup}$ is a set of acyclic superiority relations.

A conclusion of a theory D is a tagged literal φ having one of the following forms:

– $+\Delta\varphi$ and $-\Delta\varphi$, meaning respectively that φ *is*, or is *not* definitely provable in D,
– $+\partial\varphi$ and $-\partial\varphi$, meaning respectively that φ *is*, or is *not* defeasibly provable in D.

Provability is based on the concept of a derivation (or proof) in D. A derivation is a finite sequence $P = (P(1),\ldots,P(n))$ of tagged literals. Each tagged literal satisfies some proof conditions, which correspond to inference rules for the four kinds of conclusions we have mentioned above. Before moving to the conditions governing derivability of conclusions, we need to introduce some preliminary notions.

Definition 3 (Rule applicability). Let $\# \in \{\Delta, \partial\}$. If rule r is #-applicable then $\forall\varphi \in A(r)$, $+\#\varphi \in P(1..i)$. If r is #-discarded then $\exists\varphi \in A(r)$ such that $-\#\varphi \in P(1..i)$.

Definition 4 (Modality relations). For any $\varphi \in \text{ModLit}$, let the set convert($\varphi$) be:
$\text{convert}(\text{OBL}\varphi) = \{\text{PERM}\varphi, \neg\text{OBL}{\sim}\varphi\}$, $\text{convert}(\neg\text{OBL}{\sim}\varphi) = \{\text{PERM}\varphi\}$,
$\text{convert}(\text{PERM}\varphi) = \{\neg\text{OBL}{\sim}\varphi\}$, $\text{convert}(\neg\text{PERM}{\sim}\varphi) = \{\text{OBL}\varphi, \text{PERM}\varphi\}$.

Definition 5 (Intervals: basic notions). Let 'start()' and 'end()' be the functions that return respectively the lower and upper bounds of an interval. For any $T_1, T_2, T_3 \in \text{Inter}$,

– $T_1 \sqsubseteq T_2$ iff $\text{start}(T_2) \leq \text{start}(T_1)$ and $\text{end}(T_1) \leq \text{end}(T_2)$;
– $\text{over}(T_1,T_2)$ iff $\text{start}(T_1) \leq \text{end}(T_2)$ and $\text{start}(T_2) \leq \text{end}(T_1)$;
– $T_1 \sqcup T_2 = T_3$ iff $\text{end}(T_1) + u = \text{start}(T_2)$, $\text{start}(T_3) = \text{start}(T_1)$ and $\text{end}(T_3) = \text{end}(T_2)$.

We provide below the proof conditions to determine whether a temporal (modal) literal $\gamma{:}T$ is a definite conclusion of a theory D.

If $P(i+1) = +\Delta\gamma{:}T$ then
(1) $\gamma{:}T' \in F$ s.t. $T \sqsubseteq T'$, or
(2) $+\Delta\beta{:}T \in P(1..i)$ s.t. $\gamma \in \text{convert}(\beta)$, or
(3) $\exists r \in R_s[\gamma{:}T']$,
  s.t. $T \sqsubseteq T'$, r is Δ-applicable, or
(4) $+\Delta\gamma{:}T' \in P(1..i)$ and $+\Delta\gamma{:}T'' \in P(1..i)$,
  s.t. $T' \sqcup T'' = T$.

If $P(i+1) = -\Delta\gamma{:}T$ then
(1) $\gamma{:}T' \notin F$ s.t. $T \sqsubseteq T'$, and
(2) $-\Delta\beta{:}T \in P(1..i)$ s.t. $\gamma \in \text{convert}(\beta)$, and
(3) $\forall r \in R_s[\gamma{:}T']$,
  s.t. $T \sqsubseteq T'$, r is Δ-discarded, and
(4) $-\Delta\gamma{:}T' \in P(1..i)$ or $-\Delta\gamma{:}T'' \in P(1..i)$,
  s.t. $T' \sqcup T'' = T$.

Defeasible provability deals with conflicts. This requires to state when two literals are in conflict with each other:

Definition 6 (Complementary modal literals). Let $l \in$ Lit and $X \in$ Mod. The set $\mathscr{C}(Xl)$ of complements of Xl is defined as follows:
$\mathscr{C}(\text{OBL}\,l) = \{\neg\text{OBL}\,l, \text{OBL}{\sim}l, \neg\text{PERM}l, \text{PERM}{\sim}l\}$ $\mathscr{C}(\neg\text{OBL}l) = \{\text{OBL}l, \neg\text{PERM}l\}$
$\mathscr{C}(\text{PERM}l) = \{\text{OBL}{\sim}l, \neg\text{PERM}l\}$ $\mathscr{C}(\neg\text{PERM}l) = \{\neg\text{OBL}{\sim}l, \text{PERM}l, \neg\text{PERM}{\sim}l\}$

As usual with defeasible logic, the derivation of a conclusion follows a three phase protocol. First we provide an argument in favour of the conclusion we want to prove. Second, we have to consider all possible counterarguments. Third, we have to rebut each of those counterarguments, i.e., find counterarguments against them.

In TMDL we must consider the times and intervals associated with the conclusion and the premises. So to prove a conclusion at time t we must find a rule of which the interval associated with the conclusion covers t. For counterarguments we look for rules producing the complement of the conclusion to be proved, and whose interval overlaps with the interval of the conclusion. The proof conditions are as follows.

If $P(i+1) = +\partial\gamma{:}T$ then
(1) $+\Delta\gamma{:}T \in P(1..i)$, or
(2) $-\Delta\beta{:}T' \in P(1..i)$,
  s.t. $\beta \in \mathscr{C}(\gamma)$, over$(T', T)$, and
 (2.1) $+\partial\beta{:}T' \in P(1..i)$, $\gamma \in \text{convert}(\beta)$, or
 (2.2) $\exists r \in R_{sd}[\gamma{:}T']$,
   s.t. $T \sqsubseteq T'$, r is ∂-applicable, and
 (2.2.1) $\forall s \in R[\alpha{:}T']$,
   s.t. $\alpha \in \mathscr{C}(\gamma)$, over$(T', T)$
  (2.2.1.1) s is ∂-discarded, or
  (2.2.1.2) $\exists w \in R[\gamma{:}T'']$, s.t. $T \sqsubseteq T''$,
    w is ∂-applicable, and $w \succ s$, or
(3) $+\partial\gamma{:}T' \in P(1..i)$ and $+\partial\gamma{:}T'' \in P(1..i)$,
  s.t. $T' \sqcup T'' = T$.

If $P(i+1) = -\partial\gamma{:}T$ then
(1) $-\Delta\gamma{:}T \in P(1..i)$, and
(2) $+\Delta\beta{:}T' \in P(1..i)$,
  s.t. $\beta \in \mathscr{C}(\gamma)$, over$(T', T)$, or
 (2.1) $-\partial\beta{:}T' \in P(1..i)$, $\gamma \in \text{convert}(\beta)$, and
 (2.2) $\forall r \in R_{sd}[\gamma{:}T']$,
   s.t. $T \sqsubseteq T'$, r is ∂-discarded, or
 (2.2.1) $\exists s \in R[\alpha{:}T']$,
   s.t. $\alpha \in \mathscr{C}(\gamma)$, over$(T', T)$
  (2.2.1.1) s is ∂-applicable, and
  (2.2.1.2) $\forall w \in R[\gamma{:}T'']$, s.t. $T \sqsubseteq T''$,
    w is ∂-discarded, or $w \not\succ s$, and
(3) $-\partial\gamma{:}T' \in P(1..i)$ or $-\partial\gamma{:}T'' \in P(1..i)$,
  s.t.$T' \sqcup T'' = T$.

Let us explain the proof condition of $+\partial\gamma{:}T$, the defeasible provability of $\gamma{:}T$. There are two cases: (i) show that $\gamma{:}T$ is already definitely provable (see (1)), or (ii) use the defeasible part of D. To prove $\gamma{:}T$ defeasibly we must show that its complements β are not definitely provable (see (2)). In clause (2.1), we show the provability of $\beta{:}T'$ such that β can be converted into γ (see Definition 4). In clause (2.2), we require that there is a strict or defeasible rule $r \in R$ applicable, with $\gamma{:}T'$ as its head such that interval T' includes T. Now we consider possible attacks: reasoning chains in support of a complement α of γ: any rule $s \in R$ with a complement $\alpha{:}T'$ as its head and over(T',T). These attacking rules must be discarded (2.2.1.1), or must be counterattacked by a stronger rule w (2.2.1.2). Finally, in clause (3) we deal with the case where γ is defeasible provable on T' and T'', which form T.

The proof conditions for $-\partial\gamma{:}T$, that $\gamma{:}T$ is *not* defeasibly provable, are structured in a similar way. Positive derivations are replaced by negative ones and vice versa.

To illustrate the derivations, let us consider the following example:

r_1 $b{:}5 \Rightarrow \mathrm{OBL}a{:}[10,15]$ $\quad$ r_2 $c{:}6 \Rightarrow \neg\mathrm{OBL}a{:}[6,9]$
r_3 $d{:}[5,10] \Rightarrow \neg\mathrm{OBL}a{:}[12,20]$ $\quad$ r_4 $e{:}[7,8] \Rightarrow \neg\mathrm{OBL}a{:}[10,15]$

Given facts $\{b{:}[1,5], c{:}6, d{:}5,10, e{:}7\}$, we want to know if there is an obligation OBLa at time 10, i.e., $+\partial\mathrm{OBL}a{:}10$. Since b holds from 1 to 5, rule r_1 is applicable, and we have an argument to support the conclusion. Then we have to consider the putative arguments against the obligation at the chosen time. The range of effectiveness of r_2 is $[6,9]$, so that does not overlap with the conclusion of r_1, namely $[10,15]$. The range of effectiveness of r_3 is $[12,20]$ which overlaps with the conclusion of r_1, but it does not cover the intended conclusion (10). So, we can discard this rule as well. Finally, the range of r_4 overlaps with the conclusion of r_1, and covers the time for the conclusion we want to derive. However, in this case, we can rebut the argument since its premise is that e holds from 7 to 8, but we are given that e holds only at 7, so we cannot use this rule. Since there are no more arguments against OBLa:10 we can conclude it.

3 Analysing Deadlines

In this section we provide a number of parameters to analyse deadline situations. The hypothesis is that different types of deadlines can be characterised by instantiations of these parameters. Each parameter corresponds to one or more template rules. By making combinations of different template rules, we can generate every possible type of deadline clause.

(1) Customers must pay within 30 days, after receiving the invoice.

Example (1) can be represented as follows. The deadline refers to an obligation triggered by receipt of the invoice (rule inv_{init}). After that the customer is obliged to pay. The obligation terminates when it is complied with (rule inv_{term}). The termination is modelled by a defeater rule ($\leadsto$). Note that the obligation itself may even persist after the deadline. Generally, a deadline signals that a violation of the obligation has occurred (rule inv_{viol}). This may or may not trigger an explicit sanction (see below).

inv_{init} $\mathrm{get\_invoice}{:}t_1 \Rightarrow \mathrm{OBLpay}{:}[t_1, max]$
inv_{term} $\mathrm{OBLpay}{:}t_2, \mathrm{pay}{:}t_2 \leadsto \neg\mathrm{OBLpay}{:}t_2+1$
inv_{viol} $\mathrm{get\_invoice}{:}t_1, \mathrm{OBLpay}{:}t_1+31 \Rightarrow \mathrm{viol(inv)}{:}t_1+31$

Suppose that the set of facts is $\{\mathrm{get\_invoice}{:}0, \mathrm{pay}{:}20\}$. We can derive $+\partial\mathrm{get\_invoice}{:}0$, which makes rule inv_{init} applicable, leading to $+\partial\mathrm{OBLpay}{:}[0, max]$, that is, an obligation to pay applies indefinitely. Rule inv_{term} terminates the obligation at 21. Therefore rule inv_{viol} is not applicable, and we cannot derive a violation: $-\partial\text{viol(inv)}{:}30$.

This example can be turned into a general template for the representation of deadlines, where $\chi_1,..\chi_n$ are one or more contextual constraints, which trigger the existence of the obligation and fix time variables like t, φ is the condition to be achieved, and δ is the deadline condition. In the example above δ only specifies the 30 days period, but in general δ could be any action or event, such as 'having dinner', in 'I must finish work before dinner', when you do not exactly know when dinner will be. The violation fact $\text{viol}(r)$ is a specific literal, indexed by the name of the group of rules.

$$\begin{array}{lll} r_{init} & \chi_1{:}t_1,...,\chi{:}t_n \Rightarrow \text{OBL}\varphi{:}[t,max] & \text{(initialise)} \\ r_{term} & \text{OBL}\varphi{:}t,\varphi{:}t \rightsquigarrow \neg\text{OBL}\varphi{:}t+1 & \text{(terminate)} \\ r_{viol} & \delta{:}t_\delta,\text{OBL}\varphi{:}t_\delta \Rightarrow \text{viol}(r){:}t_\delta & \text{(violation)} \end{array}$$

Achievement or Maintenance? Based on similar ideas about intentions and goals, we can distinguish *achievement obligations*, like example (1), from so called *maintenance obligations*, like example (2). For an achievement obligation, the condition φ must occur at least once before the deadline. For maintenance obligations, condition φ must obtain during all instants before the deadline. Here, the deadline only signals that the obligation is terminated. A violation occurs, when the obliged state does not obtain at some point before the deadline. Note that prohibitions, i.e. obligations to avoid some 'bad state', form a large class of maintenance obligations.

(2) Customers must keep a positive balance, for 30 days after opening an bank account.

$$\begin{array}{ll} \text{pos}_{init} & \text{open\_account}{:}t_1 \Rightarrow \text{OBLpositive}{:}[t_1,t_1+30] \\ \text{pos}_{viol} & \text{OBLpositive}{:}t_2,\neg\text{positive}{:}t_2 \Rightarrow \text{viol(pos)}{:}t_2 \end{array}$$

The generic formalisation for maintenance obligations consists of the following template formulas. Note that no termination rule is needed.

$$\begin{array}{lll} r_{init\_main} & \chi{:}t,\delta{:}t_\delta \Rightarrow \text{OBL}\varphi{:}[t,t_\delta] & \text{(initialise maintenance)} \\ r_{viol\_main} & \text{OBL}\varphi{:}t,\neg\varphi{:}t \Rightarrow \text{viol}(r){:}t & \text{(violation of maintenance)} \end{array}$$

Persistence after the deadline. By definition, maintenance obligations do not persist after the deadline. But achievement obligations often do persist after the deadline, until they are achieved. However, this is not the case for all achievement obligations.

(3) A wedding cake must be delivered, before the wedding party.

In example (3), the obligation to deliver the cake does not persist after the deadline, since the wedding guests will have no use for it. In addition, the couple who ordered the cake, are entitled not to pay for the cake, or even claim damages after the deadline has passed without delivery. This can be seen as a kind of sanction.

In general, the difference between persistent and non-persistent achievement obligations will depend on conventions about the underlying goal of the deadline. Non-persistent obligations have a termination rule, similar to maintenance obligations.

wed_{init} $\text{order}{:}t_1, \text{wedding}{:}t_2 \Rightarrow \text{OBLcake}{:}[t_1, t_2]$
wed_{term} $\text{OBLcake}{:}t_3, \text{cake}{:}t_3 \rightsquigarrow \neg\text{OBLcake}{:}t_3 + 1$
wed_{viol} $\text{wedding}{:}t_2, \text{OBLcake}{:}t_2 \Rightarrow \text{viol}(wed){:}t_2$

The rules for non-persistent deadlines can be generalised as follows [1].

$r_{init\_non\_pers}$ $\chi{:}t, \delta{:}t_\delta \Rightarrow \text{OBL}\varphi{:}[t, t_\delta]$ (initialise non-persistence)

Explicit or implicit Sanctions. Many normative rules are associated with an explicit sanction, such as payment of a fine. But this is not always the case. Often there is only an *implicit sanction*, of not being able to achieve the goal that underlies the deadline. For a sanction to be effective, it needs to be strongly disliked by the person sanctioned.

(4) Customers must pay within 30 days, after receiving the invoice. Otherwise, an additional fine must be paid.
(5) Customers must keep a positive balance for 30 days after opening an account, otherwise their account is blocked.
(6) A wedding cake must be delivered, before the wedding party. If not, the contract is breached and (for example) the customer may not have to pay.

An explicit sanction is often implemented through a separate obligation, which is triggered by a detected violation. In this setting, legislators may need further deadlines to enforce the sanctions, leading to a 'cascade' of obligations. The obligation to pay a fine in (4) is an example of that. In other cases, like in example (5), the sanction is applied automatically. The wedding cake example (6) is a case of mutual obligations. A late delivery is a breach of contract. So the sanction is typically that the customer does not have to pay for the cake. An obligation to pay for the cake may be derived from another rule r or, alternatively, it is given as a fact. Clearly, in the former case we require that $\text{cake}_{sanc} \succ r$. Continuing the previous representations of example (1), (2) and (3), we add the following rules to account for the sanctions in (4), (5) and (6).

inv_{sanc} $\text{viol}(\text{inv}){:}t \Rightarrow \text{OBLpay\_fine}{:}[t, max]$
pos_{sanc} $\text{viol}(\text{pos}){:}t \Rightarrow \text{account\_blocked}{:}[t, max]$
wed_{sanc} $\text{viol}(\text{wed}){:}t, \text{OBLpay}{:}[t, max] \Rightarrow \neg\text{OBLpay}{:}[t+1, max]$

In example (4), suppose that the set of facts is $\{\text{get\_invoice}{:}0\}$. We have $+\partial\text{OBLpay}{:}31$. The deadline is not respected, and so we derive $+\partial\text{OBLpay\_fine}{:}[31, max]$. Regarding example (5), suppose that the facts are $\{\text{open\_account}{:}0, \neg\text{positive}{:}[20, 30]\}$. We derive $+\partial\text{OBLpositive}{:}[0, 30]$. The balance rule is violated, so $+\partial\text{account\_blocked}{:}[20, max]$. In example (6), suppose we have $\{\text{order}{:}0, \text{OBLpay}{:}[0, max], \text{wedding}{:}12\}$. Because the termination rule does not apply, we can derive $+\partial\text{OBLcake}{:}12$. By wed_{viol} we derive $+\partial\text{viol}(\text{wed}){:}12$, and therefore $+\partial\neg\text{OBLpay}{:}[13, max]$.

Because obligational sanctions are most common, rule inv_{sanc} is generalised into a generic sanction rule, where σ is strongly disliked by the agent being sanctioned.

r_{sanc} $\text{viol}(r){:}t \Rightarrow \text{OBL}\sigma{:}[t, max]$ (sanction)

[1] Only r_{init} has changed. Alternatively, we could leave r_{init} unchanged, and alter r_{term}.

Notice that sanctions are not always needed. In example (7) we see a non-persistent deadline, with no explicit sanction, apart from the impossibility to achieve one's goals.

(7) Dinner guests must arrive before dinner.

Obligation or Permission? Deadlines are not only connected with obligations. In example (8), there is a permission – or entitlement – which expires at the deadline.

(8) Within 30 days after delivery, customers may return a product to the seller.

Using the principle that an entitlement for one, is an obligation for the other to respect that entitlement, we could also say that after 30 days the seller is no longer obliged to take the product back. Permissions are similar to maintenance obligations, and can be modelled as follows:

$$\text{ret}_{init}\ \text{delivery}{:}t \Rightarrow \text{PERMreturn\_product}{:}[t, t+30]$$

The right to return a product is an example of an achievement permission. Example (9), roughly the opposite of example (2), illustrates a maintenance permission: the right to remain in a certain state for a certain period.

(9) Customers may have a negative balance, for at most 30 consecutive days.

Here we assume to fix the time when the account gets a negative balance.

$$\begin{array}{ll}
\text{bal}_{init} & \text{positive}{:}t_1-1, \neg\text{positive}{:}t_1 \Rightarrow \text{PERM}\neg\text{positive}{:}[t_1, t_1+30] \\
\text{bal}_{term} & \neg\text{positive}{:}[t_1, t_1+30], \text{PERM}\neg\text{positive}{:}[t_1, t_1+30] \Rightarrow \\
 & \neg\text{PERM}\neg\text{positive}{:}[t_1+31, max] \\
\text{bal}_{viol} & \neg\text{PERM}\neg\text{positive}{:}t_2, \neg\text{positive}{:}t_2 \Rightarrow \text{viol(bal)}{:}t_2
\end{array}$$

Although one may observe that (positive) achievement permissions generally do not have violations or explicit sanctions, violations or sanctions can in fact be connected to maintenance permissions. After the permission ends, usually an original obligation will be restored. Remember that $\neg\text{PERM}\neg\varphi \equiv \text{OBL}\varphi$. So, depending on wether you want to model this as strong or weak permission [9], a negative balance after the permission has run out, does constitute a violation. In some cases, this may also lead to an explicit sanction, as illustrated by (10).

(10) Customers may have a negative balance, for at most 30 consecutive days. After that period, an interest rate of 10% must be paid.

$$\text{bal}_{sanc}\ \text{viol(bal)}{:}t \Rightarrow \text{OBLpay\_interest}{:}t$$

This discussion leads to the following general template rules, for achievement and maintenance permission, respectively. The sanction rule is again r_{sanc}.

$$\begin{array}{lll}
r_{init\_perm} & \chi{:}t, \delta{:}t_\delta \Rightarrow \text{PERM}\varphi{:}[t, t_\delta] & \text{(init. achiev. perm.)} \\
r_{init\_perm\_main} & \chi{:}t, \delta{:}t_\delta \Rightarrow \text{PERM}\varphi{:}[t, t_\delta] & \text{(init. maint. perm.)} \\
r_{term\_perm\_main} & \chi{:}t, \delta{:}t_\delta, \text{PERM}\varphi{:}[t, t_\delta] \Rightarrow \neg\text{PERM}\varphi{:}[t_\delta+1, max] & \text{(term. maint. perm.)} \\
r_{viol\_perm\_main} & \neg\text{PERM}\varphi{:}t', \varphi{:}t' \Rightarrow \text{viol}(r){:}t' & \text{(viol. maint. perm.)}
\end{array}$$

For achievement permissions or entitlements it generally does not make sense to have an explicit sanction. There is only the implicit sanction, namely that the agent is not able to achieve the underlying goals, made possible by the permission. This is a very common situation for permissions, but also for opportunities. In example (11), there is no deontic operator at all. Here, it is the practical opportunity to travel, which expires.

(11) A traveller wants to travel to Rome. The last train to Rome departs at 22:05.

So whence the popular idea that you *must* return the product within 30 days, or that you *must* catch the train? Well, given the fact that you are not satisfied with the product and that you intend to return the product, or given the fact that you intend to travel, the only way to achieve these intentions is before the deadline. The necessity is not deontic, but is based on practical reasoning. We observe that many deadlines have this hybrid nature of deontic and practical reasoning.

This completes our list of parameters. We can now construct different types of deadlines by varying the parameters discussed above. The result is shown in Table 1. For each of the types, we have listed the corresponding examples.

From the table it is clear that some combinations of parameters do not make sense. First, (positive) achievement permissions do not have sanctions, except for the implicit sanction of not being able to achieve one's goal. That excludes type 5, 6 from the table. Moreover, the deadline of a permission or a maintenance obligation, is the moment from which the permission or obligation no longer persist. So by definition, type 5 (again), 7, 9, 11, 13 and 15 are ruled out as well. The remaining 9 deadline types have been

Table 1. Deadline typology

	achieve	operator	sanction	persist	examples	rule templates
1.	Ach.	OBL	Y	Y	payment, fine (4)	$r_{init}, r_{term}, r_{viol}, r_{sanc}$
2.	Ach.	OBL	Y	N	wedding cake (3), (6)	$r_{init\_non\_pers}, r_{term}, r_{viol}, r_{sanc}$
3.	Ach.	OBL	N	Y	payment, no fine (1)	$r_{init}, r_{term}, r_{viol}$
4.	Ach.	OBL	N	N	dinner guests (7)	$r_{init\_non\_pers}, r_{term}, r_{viol}$
5.	Ach.	PERM	Y	Y	no sense	
6.	Ach.	PERM	Y	N	no sense	
7.	Ach.	PERM	N	Y	no sense	
8.	Ach.	PERM	N	N	return product (8)	r_{init_perm}
9.	Maint.	OBL	Y	Y	no sense	
10.	Maint.	OBL	Y	N	account, blocked (5)	$r_{init\_main}, r_{viol\_main}, r_{sanc}$
11.	Maint.	OBL	N	Y	no sense	
12.	Maint.	OBL	N	N	account (2)	$r_{init\_main}, r_{viol\_main}$
13.	Maint.	PERM	Y	Y	no sense	
14.	Maint.	PERM	Y	N	negative, interest (10)	$r_{init\_main\_perm}, r_{term\_main\_perm}, r_{viol\_main\_perm}, r_{sanc}$
15.	Maint.	PERM	N	Y	no sense	
16.	Maint.	PERM	N	N	negative (9)	$r_{init\_main\_perm}, r_{term\_main\_perm}, r_{viol\_main\_perm}$

illustrated by examples. By a combination of the template rules for each of these types, we can therefore represent each possible deadline type in a uniform way.

4 Conclusions

In this paper we have given an analysis of deadlines, expressed in Temporal Modal Defeasible Logic. We have chosen a defeasible logic, because it can distinguish exceptions from violations, and because it allows us to have rules override the effect of other rules. This makes it relatively straightforward to model the initiation, termination and violation rules that generally specify a deadline. We have chosen to use intervals, because they provide a natural representation of deadlines, in particular of the distinction between achievement and maintenance obligations.

Although defeasible logic has been already extended to represent contracts [6], this is the first attempt to integrate time. In artificial intelligence, Allen's [1] interval algebra is widely applied. Allen provides a model of the various ways in which temporal intervals may interact and overlap. Note, however, that the satisfiability problem in Allen's algebra is NP-complete [11], which motivated the study of complexity in subalgebras. Although our logic is less expressive and does not cover Allen's algebra operations, its computational complexity is a matter of future research.

Previous work on deadlines typically uses a combination of deontic logic and branching time temporal logic [5,4,3]. However, existing work has typically not addressed what happens *after* the deadline. There are many kinds of deadlines, with different functions. In this paper we have presented a typology of deadlines, using the following parameters: distinction between achievement and maintenance obligations, persistence of the obligation after the deadline, presence of an explicit sanction and the type of operator, obligation or permission. By combining template rules for each of these parameters, we can give a formal characterisation of each of these deadline types. Currently, the typology is validated against a case study of a rental agreement.

References

1. Allen, J.F.: Towards a general theory of action and time. Art. Intelligence 23, 123–154 (1984)
2. Antoniou, G., Billington, D., Governatori, G., Maher, M.J.: Representation results for defeasible logic. ACM Transactions on Computational Logic 2(2), 255–287 (2001)
3. Broersen, J., Dignum, F., Dignum, V., Meyer, J.-J.: Designing a deontic logic of deadlines. In: Lomuscio, A.R., Nute, D. (eds.) DEON 2004. LNCS (LNAI), vol. 3065, pp. 43–56. Springer, Heidelberg (2004)
4. Dignum, F., Kuiper, R.: Specifying deadlines with obligations and dense time. International Journal of Electronic Commerce 3(2), 67–85 (1998)
5. Dignum, F., Weigand, H., Verharen, E.: Meeting the deadline: on the formal specification of temporal deontic constraints. In: Michalewicz, M., Raś, Z.W. (eds.) ISMIS 1996. LNCS, vol. 1079, pp. 243–252. Springer, Heidelberg (1996)
6. Governatori, G.: Representing business contracts in RuleML. International Journal of Cooperative Information Systems 14(2-3), 181–216 (2005)
7. Governatori, G., Padmanabhan, V., Rotolo, A.: Rule-based agents in temporalised defeasible logic. In: Yang, Q., Webb, G. (eds.) PRICAI 2006. LNCS (LNAI), vol. 4099, pp. 31–40. Springer, Heidelberg (2006)

8. Governatori, G., Palmirani, M., Riveret, R., Rotolo, A., Sartor, G.: Norm modifications in defeasible logic. In: Proceedings of JURIX 2005, pp. 13–22 (2005)
9. Governatori, G., Rotolo, A., Sartor, G.: Temporalised normative positions in defeasible logic. In: Proceedings of ICAIL 2005, pp. 25–34 (2005)
10. Muller, N.J.: Managing service level agreements. International Journal of Network Management 9(3), 155–166 (1999)
11. Vilain, M., Kautz, H., van Beek, P.: Constraint propagation algorithms for temporal reasoning. In: Readings in qualitative reasoning about physical systems, Morgan Kaufmann, San Francisco (1989)

Dialogue Games in Defeasible Logic

S. Thakur[1], G. Governatori[1], V. Padmanabhan[2], and J. Eriksson Lundström[3]

[1] School of Information Technology and Electrical Engineering
The University of Queensland, Brisbane, Queensland, 4072, Australia
[2] Dept. of Computer & Information Sciences, University of Hyderabad, India
[3] Uppsala University, Sweden
{subhasis, guido}@itee.uq.edu.au, vineetcs@uohyd.ernet.in,
jenny.eriksson@dis.uu.se

Abstract. In this paper we show how to capture dialogue games in Defeasible Logic. We argue that Defeasible Logic is a natural candidate and general representation formalism to capture dialogue games even with requirements more complex than existing formalisms for this kind of games. We parse the dialogue into defeasible rules with time of the dialogue as time of the rule. As the dialogue evolves we allow an agent to upgrade the strength of unchallenged rules. The proof procedures of [1] are used to determine the winner of a dialogue game.

1 Introduction

Agents interact with other agents. The nature of the interactions between two agents can be of various kinds. Here we consider two types of interaction: cooperative and adversarial. In a cooperative situation the agents exchange information with the aim of reaching a common goal, while in an adversarial scenario the goals of the parties are conflicting. However, this does not imply a clear-cut dissimilarity between the two types of interaction. Conflicting sub-goals often are found amongst agents in a cooperative setting, while all the same in an adversarial discussion one agent may partially accept a proposal of her adversary as it provides for a stronger justification of her case.

These kinds of interactions are part of the broader field of argumentation, and formal argumentation is the branch using logic and (formal methods in general) to model it. Over the past few years a line of research emerged for the representation of these type of arguments: dialogue games. Dialogue games are proven extremely useful for modelling some forms of legal reasoning. In this paper we focus on one form of dialogue games, the adversarial, where the two parties debate over one topic.

Most formal models of dialogues provide computational and procedural representations of some real-life domain (e.g., legal reasoning). Dialogue games are by their own nature defeasible, it means that arguments put forward by one of the agents in support of a conclusion can be defeated by contrary evidence put forward by the other agent. Accordingly, standard model-theoretic semantics is not appropriate for this kind of reasoning. Dung [4] proposed argumentation semantics to obviate this issue. The main idea of argumentation semantics is that the main objects we evaluate are “arguments”[1]. Various relationships (e.g. attack, rebut and defeat) between arguments are defined by the semantics, and the relationships are extended to sets of arguments. The key notion for a set of arguments is the notion of *support*, that is whether a set of arguments is self-consistent and provides the base to derive a conclusion. In other words, if it is possible to prove the conclusion from the rules, facts and assumptions in the set of supporting arguments. A conclusion is justified, and thus provable, if there is a set of supporting arguments and all counterarguments are deficient when we consider the arguments in the set of supporting arguments. Various argumentation semantics have been proposed to capture different relationships between supporting and opposing set of arguments. However, in general some forms of argumentation semantics are able to characterise dialogue games [18].

Defeasible Logic [13,1] is an efficient non-monotonic formalism that encompasses many logics proposed for legal reasoning. Defeasible Logic can be characterised in terms of argumentation semantics [5], thus the correspondence between Defeasible Logic on one side and dialogue games on the other follows implicitly from their common semantics. The aim of this paper is to propose a direct mapping between dialogue games and Defeasible Logic, and to show that Defeasible Logic offers a general, powerful and computationally efficient framework to model and to extend dialogue games.

The paper is organised as follows: in Section 2 we outline the basic ideas of dialogue games, then Section 3 provides an informal introduction to defeasible logic, and in Section 4 we show how to adapt Defeasible Logic to model dialogue games. We conclude the paper in Sections 5 and 6 with a discussion of related work and possible research extensions.

2 On Dialogue Games

We consider dialogue games as a game where we have two players called the *Proponent* and the *Opponent*. Each player is equipped with a set of arguments, a subset of which the players *move*, i.e., take turns in putting forward. The aim of the game is to justify a particular conclusion while adhering to the particular protocol scheme governing the game. A basic protocol for the admissible moves by the players be, for the proponent, that the current move attacks the previous move of the opponent, and that the main claim (the content of the dispute) follows from the arguments assessed as currently valid. For the opponent we have that the arguments of the move attack the previous move, and the main claim is not derivable. Even though more complex winning conditions are possible, by a basic protocol a player wins the dialogue game when the other party is out of admissible moves.

3 Basic Defeasible Logic

Over the years Defeasible Logic [13,1] proved to be a simple, flexible, rule based non-monotonic formalism able to capture different and sometimes incompatible facets of

[1] In the abstract formulation of the argumentation semantics ‘arguments’ is left unspecified, however, in the majority of concrete instances of the argumentation framework, arguments are defined as a chain of reasoning based on facts or assumptions and rules captured in some formal language or logic.

non-monotonic reasoning [2], and efficient and powerful implementations have been proposed [12,8].

Knowledge in Defeasible Logic can be represented in two ways: facts and rules.

Facts are indisputable statements represented by ground literals. For example, "Tweety is a penguin" is represented by $\mathit{Penguin}(\mathit{Tweety})$.

A *rule*, on the other hand, describes the relationship between a set of literals (premises) and a literal (conclusion), and we can specify how strong the relationship is. As usual rules allow us to derive new conclusions given a set of premises. We distinguish between *strict rules*, *defeasible rules* and *defeaters* represented, respectively, by expressions of the form $A_1, \ldots, A_n \to B$, $A_1, \ldots, A_n \Rightarrow B$ and $A_1, \ldots, A_n \leadsto B$, where $A_1, \ldots, A_n$ is a possibly empty set of prerequisites (causes) and B is the conclusion (effect) of the rule. We only consider rules that are essentially propositional, this means that rules containing free variables are to be interpreted as rule schemas and the correspond to the set of their ground instances.

Strict rules are rules in the classical sense: whenever the premises are indisputable then so is the conclusion. Thus they can be used for definitional clauses. An example of a strict rule is "Penguins are birds", formally: $\mathit{Penguin}(X) \to \mathit{Bird}(X)$.

Defeasible rules are rules that can be defeated by contrary evidence. An example of such a rule is "Birds usually fly": $\mathit{Bird}(X) \Rightarrow \mathit{Fly}(X)$. The idea is that if we know that X is a bird, then we may conclude that X can fly *unless there is other evidence suggesting that she may not fly*.

Defeaters are special kind of rules. They are used to prevent conclusions, not to support them. For example: $\mathit{Heavy}(X) \leadsto \neg \mathit{Fly}(X)$. This rule states that if something is heavy then it might not fly. This rule can prevent the derivation of a "fly" conclusion. On the other hand it cannot be used to support a "not fly" conclusion.

Defeasible logic (DL) is a "skeptical" non-monotonic logic, meaning that it does not support contradictory conclusions. Instead DL seeks to resolve conflicts. In cases where there is some support for concluding A but also support for concluding $\neg A$, DL does not conclude either of them (thus the name "skeptical"). If the support for A has priority over the support for $\neg A$ then A is concluded. No conclusion can be drawn from conflicting rules in DL unless these rules are prioritised. The *superiority relation* among rules is used to define priorities among rules, that is, where one rule may override the conclusion of another rule. For example, given the defeasible rules

$$r : \mathit{Bird}(X) \Rightarrow \mathit{Fly}(X) \qquad r' : \mathit{Penguin}(X) \Rightarrow \neg \mathit{Fly}(X)$$

which contradict one another, no conclusive decision can be made about whether a Tweety can fly or not. But if we introduce a superiority relation $\succ$ with $r' \succ r$, then we can indeed conclude that Tweety cannot fly since it is a penguin.

We now give a short informal presentation of how conclusions are drawn in DL. A D be a theory in DL (i.e., a collection of facts, rules and a superiority relation). A *conclusion* of D is a tagged literal and can have one of the following four forms:

$+\Delta q$ meaning that q is definitely provable in D (i.e., using only facts and strict rules).
$-\Delta q$ meaning that we have proved that q is not definitely provable in D.
$+\partial q$ meaning that q is defeasibly provable in D.
$-\partial q$ meaning that we have proved that q is not defeasibly provable in D.

Strict derivations are obtained by forward chaining of strict rules, while a defeasible conclusion p can be derived if there is a rule whose conclusion is p, whose prerequisites (antecedent) have either already been proven or given in the case at hand (*i.e.*, facts), and any stronger rule whose conclusion is $\neg p$ has prerequisites that fail to be derived. In other words, a conclusion p is derivable when:

- p is a fact; or
- there is an applicable strict or defeasible rule for p, and either
  - all the rules for $\neg p$ are discarded (i.e., are proved to be not applicable) or
  - every applicable rule for $\neg p$ is weaker than an applicable strict[2] or defeasible rule for p.

Formally a Defeasible Logic theory (as formalised in [3]) is a structure $D = (F,R,>)$ where F is a finite set of factual premises, R a finite set of rules, and $>$ a superiority relation on R. Given a set R of rules, we denote the set of all strict rules in R by R_s, the set of strict and defeasible rules in R by R_{sd}, the set of defeasible rules in R by R_d, and the set of defeaters in R by R_{dft}. $R[q]$ denotes the set of rules in R with consequent q. In the following $\sim p$ denotes the complement of p, that is, $\sim p$ is $\neg q$ if $p = q$, and $\sim p$ is q if p is $\neg q$. For a rule r we will use $A(r)$ to indicate the body or antecedent of the rule and $C(r)$ for the head or consequent of the rule. A rule r consists of its antecedent $A(r)$ (written on the left; $A(r)$ may be omitted if it is the empty set) which is a finite set of literals, an arrow, and its consequent $C(r)$ which is a literal. In writing rules we omit set notation for antecedents.

Provability is based on the concept of a derivation (or proof) in D. A derivation is a finite sequence $P = (P(1),\dots,P(n))$ of tagged literals. Each tagged literal satisfies some proof conditions. A proof condition corresponds to the inference rules corresponding to one of the four kinds of conclusions we have mentioned above. $P(1..i)$ denotes the initial part of the sequence P of length i. Here we state the conditions for strictly and defeasibly derivable conclusions (see [1] for the full presentation of the logic):

If $P(i+1) = +\Delta q$ then
  (1) $q \in F$, or
  (2) $r \in R_s[q]$, $\forall a \in A(r) : +\Delta a \in P(1..i)$.

If $P(i+1) = +\partial q$ then
  (1) $+\Delta q \in P(1..i)$, or
  (2) (2.1) $\exists r \in R_{sd}[q] \forall a \in A(r) : +\partial a \in P(1..i)$ and
    (2.2) $-\Delta{\sim}q \in P(1..i)$ and
    (2.3) $\forall s \in R[{\sim}q]$ either
      (2.3.1) $\exists a \in A(s) : -\partial a \in P(1..i)$ or
      (2.3.2) $\exists t \in R_{sd}[q]\ \forall a \in A(t) : +\partial a \in P(1..i)$ and $t > s$.

4 Modelling Dialogue Games in Defeasible Logic

During the dialogue game the agents take turns in presenting their arguments (rules). The literal the agents are trying to prove or disprove will be called the critical literal.

[2] Note that a strict rule can be defeated only when its antecedent is defeasibly provable.

Each agent has knowledge that initially is known only to the agent (private knowledge). Initially all the arguments are private. In addition, both agents have access to a set of common knowledge. By putting forward arguments from the private knowledge of an agent, these arguments becomes part of the set of common knowledge. We assume that all arguments (rules) are defeasible, acknowledging an agent's right to put forward interpretations of assumptions, fact and evidence in the way most favourable for his case (cf. [9]). The set of common arguments is continuously updated at each step and defeated defeasible rules are removed at each step. At any time the set of common arguments contains defeasible rules only from the current step t_i and adjacent previous step t_{i-1} and facts, strict rules from previous steps. The theory of common set of arguments is $T_{common} = (F, R, >)$, where F is the set of facts, R is the set of rules, $>$ is superiority relationship among the rules. At each step the proof procedure is applied on the critical literal. The nature of the game determines the burden as well as the winner of the game. A party wins the game if the proof is $+\Delta A$ (A is the critical literal) at any stage of the game. If a party at any stage of the game proves $+\partial A$ (A is the critical literal) the other party has the burden to produce proof of $-\partial A$, or $+\Delta\neg A$ or $+\partial\neg A$. Our notion of burden is restricted only to the critical literal unlike the notion of burden in [15].

The condition that a agent cannot repeat its rule is unnecessary in our model. If a rule r has been put forward and successfully defeated any counterarguments supporting the opposite conclusion $\neg C(s)$ the rule r is added to the common set of knowledge. In accordance to our protocol the rule r will effectively prevent the opponent from putting forward the defeated arguments (rules) for $\neg C(r)$ into the dialogue as we require all admissible arguments to be defeating opposing arguments presented at a previous step. In addition this criterion guarantees that the dialogue game terminates, since we assume the private and public set of arguments are finite.

4.1 A Protocol

Mainly we adhere to the protocol of a dialogue game captured in [15,16,14,17]. Thus, the rules for our dialogue games are as follows:

1. The parties cannot present arguments in parallel. Thus, the parties take turn in presenting their arguments.
2. As we allow for each agent to put forward as argument the interpretations of rules and evidence in the way most favourable for his case, all arguments presented by an agent are initially treated as defeasible rules. If in next step the other party fails to provide valid counterarguments, these defeasible rules be upgraded to strict rules.
3. The arguments in support of a critical literal $\sim A$ presented by a party 1 at any step must attack (at least) the conclusion in support of the critical literal A put forward by the other party 2 in the previous step. Moreover, in order to prevent a strengthening of the defeasible rules in support of A in the set of common arguments and to remove from the set of common arguments all defeasible rules which has as conclusion A, party 1 must present at least one new argument with its own critical literal $\sim A$ as its conclusion.
4. An agent cannot attack its own arguments. In our dialogue game framework it is not admissible for an agent to contradict itself by putting forward rules with a conclusion that contradicts a rule previously presented by the agent.

5. A particular dialogue game is won by an agent when the other party at his turn cannot make an admissible move.
6. A argument r is stronger than an argument s, conflicting with r and played in the previous time-step, if r is not attacked in successive steps.

4.2 Strengthening of Rules

Dialogues are parsed into defeasible theories. The time of a dialogue is translated as the time of a rule. All the rules presented at the current step t_i and at the adjacent previous step t_{i-1} are defeasible rules. Here we consider the time as a set of finite numbers and each number is one unit more or less than its previous or next number. If not immediately rebutted by the other party, we allow for the rule strength of a rule to be strengthened from defeasible into strict. A rule is represented as $R^t_x | x \in (d, s, sd)$ where t is the time (or the move when the rule has been played), d means the rule is defeasible, s means the rule is strict, sd means the rule is either strict or defeasible. We write $a@t$ to denote the literal a being put forward or upgraded at time t. The condition for upgrading a defeasible rule to a strict rule is described below.

If p is the conclusion of a defeasible rule of the adjacent previous step t_{i-1}, we upgrade the strength of the rule to strict in next step t_{i+1} if

$$\exists r \in R^{t'}_d[p], t' < t, \forall t'' : t' < t'' < t \; R^{t''}_{sd}[\sim p] = \emptyset \text{ and } \forall a \in A(r) : +\Delta a@t$$

We stipulate that strict rules are stronger than defeasible rules.

$$\forall t, \forall r \in R^t_s[q] \text{ and } \forall s \in R^t_d[\sim q], \; r >^t s$$

At each step of the game, if an argument (rule) has precedence over any contradictory defeasible rule of the previous steps, an agent is allowed to put forward that argument. The strength is determined either by previously known superiority relationships or the validity of the rule. We assume that if at time t_2 we have a *valid* rule $R^{t_2}_2$ which contradicts a defeasible rule $R^{t_1}_1$ of time t_1 and $t_2 > t_1$, the strength of $R^{t_2}_2$ is greater than $R^{t_1}_1$. We will use defeasible logic to determine strength of a new rule.

$$w >^t s \text{ iff } w > s \in (>) \text{ or } w \in R^{t'}[p], \; s \in R^t[\sim p] \text{ where } t' < t$$

4.3 Transition Rules

The sets of common arguments construct the theories $T_1, T_2, T_3, \ldots$. Here the subscripts indicate the time when1 the sets of common arguments are constructed. At time 1, the game begins and arguments in support of a critical literal A are put forward by the proponent, then T_1 contains only defeasible rules. At time 2, the opponent proposes new defeasible rules which by the conditions presented in the Section 4.2 are stronger than some rules in theory T_1. The set of common arguments of the first two theories T_1 and T_2 consists only of defeasible rules. (Theory T_2 consists of defeasible rules from both time 1 and time 2.) Let the first theory $T_1 = (\{\}, R^1_d, >)$ be created from arguments (ARG_1) of the proponent, and the second theory $T_2 = (\{\}, R^2_d, >)$ be created through modifications of T_1 by arguments(ARG_2) from the opponent. Now the transition rules from first theory to second theory is

1. If $r \in R_d^1$ and $\forall s \in ARG_2$, $\neg C(s) \neq C(r) \wedge \neg C(s) \notin A(r)$, then $r \in R_d^2$.
2. All rules of ARG_2 are added to T_2 as defeasible rule. Here we assume that ARG_2 is valid and that a valid argument, by the above defined precedence relations, is stronger than any contradictory argument of the previous step.

At time n, $n > 2$ theory T_n is created through modification of T_{n-1} by arguments (ARG_n) of the player who has to play at that step. The rules for transition from T_{n-1} to T_n are

1. If $r \in R_s^{n-1}$ then $r \in R_s^n$.
2. If $r \in R_d^{n-2}$ and $\forall s \in ARG_{n-1}, \neg C(s) \neq C(r) \wedge \neg C(s) \notin A(r)$, then $r \in R_s^n$; otherwise $r \notin R^n$. Here we should note that the player will not oppose its previous argument. Thus, all unchallenged rules of time $n-2$ are upgraded as strict rules at time n.
3. If $r \in R_d^{n-1}$ and $\forall s \in ARG_{n-1}, \neg C(s) \neq C(r) \wedge \neg C(s) \notin A(r)$, then $r \in R_d^n$. All unchallenged defeasible rules of time $n-1$ are added to T_n as defeasible rules at time n.
4. All rules of ARG_{n-1} are added to T_n as defeasible rules. Here we assume that ARG_{n-1} is valid and that a valid argument, by the above defined precedence relations, is stronger than any contradictory argument of previous step.

It is to be noted that in the first two theories only proof of $+\partial A$ or $-\Delta A$ of a literal A could result. Thus, this framework needs at least three steps in order to determine a winner of a particular game. When the *first* party cannot produce defeasible rules that will defeat contradictory defeasible rules presented by the *second* party at the previous step, the undefeated defeasible rules of the *second* party will be strengthened into strict rules. This allows the *second* party to support its argument and hence prove the critical literal definitely. Let theory T_i be created through modification of theory T_{i-1} by argument (ARG_i) of player 1 and the critical literal A is defeasibly proven $+\partial A$ in this theory. Let at time $i+1$, player 2 cannot produce any arguments defeating the arguments of player 1, then player 1 wins at time $i+2$ as the proof as $+\partial A$ by the strengthening of the rules from defeasible into strict will result in the proof $+\Delta A$ at time $i+2$.

4.4 An Example

Now we will illustrate the model of the dialogue game defeasible logic we have proposed with the help of an example. Consider an argumentation game with two players, Alice and Bob. Agent Alice is trying to prove A and agent Bob is trying to prove $\neg A$. At each step of the dialogue game they maintain a current set of rules CR_t, where t is the time. A rule in CR_t, will be represented as $R_i'@t$ and the corresponding rule present in agent's private knowledge will be represented as R_i. Let at time t_1 the game starts and Alice makes the first move as

$$R_1 : \emptyset \Rightarrow B, \quad R_2 : B \Rightarrow A.$$

This will generate two defeasible rule as

$$R_1'(\emptyset \Rightarrow B)@t_1, \quad R_2'(B \Rightarrow A)@t_1$$

and these rules will be inserted into CR_{t_1}. Thus at time t_1 we have a proof of $+\partial A@t_1$. But at time t_2, Bob presents new evidence in order to disprove A. At time t_2 ($t_2 > t_1$), Bob presents the following argument,

$$R_3 : \emptyset \Rightarrow D, \quad R_4 : D \Rightarrow \neg A.$$

This will generate two new defeasible rules

$$R'_3(\emptyset \Rightarrow D)@t_2, \quad R'_4(D \Rightarrow \neg A)@t_2.$$

Now, Bob only attacks R'_2 presented by Alice at previous step by R'_4. Note that as $t_2 > t_1$, R'_4 is stronger than R'_2. At time t_2, R'_1 remains unchallenged and it will remain in CR_{t_2} with a changed time stamp as $R'_1(\emptyset \Rightarrow B)@t_2$. The proof at time t_2 is $+\partial\neg A$. At time t_3, Alice presents the following arguments to prove A,

$$R_5 : B \Rightarrow \neg D,\ R_6 : \emptyset \Rightarrow E \text{ and } R_7 : E \Rightarrow A.$$

So the translated defeasible rules are

$$R'_5(B \Rightarrow \neg D)@t_3,\ R'_6(\emptyset \Rightarrow E)@t_3,\ R'_7(E \Rightarrow A)@t_3.$$

Now $R'3$ is defeated by R'_5 which then makes $R'4$ not applicable. Again the time stamp of R'_1 will be changed and as this rule was unchallenged by Bob at t_2 and its strength will increase as $R'_1(\emptyset \rightarrow B)@t_3$. So the proof at time $t3$ is $+\partial A$. If Bob does not present valid arguments in the next step, Alice wins the game as we allow an agent to upgrade the strength of unchallenged rules in the next time t_4. At t_4 Alice can upgrade the defeasible rules supporting the proof of $+\partial A$ into strict rules and subsequently prove $+\Delta A$.

4.5 Reconsideration and Strategy

In our argumentation framework, as in dialogue games in general, reconsideration is not possible as rules are either carried to next step as strict rules or facts, or they are defeated. If the rule is defeated, the agent can no longer argue based on this previous decision. Also, if the status of the rule is strengthened it cannot be defeated any longer. In addition no agent can put forward arguments conflicting with the argument she put forward in previous steps. This means that an agent cannot withdrawn an argument that is used by her opponent to support the opponent critical claim. Notice that in a step an agent can have more than one set of suitable arguments. Here we present some intuitions on how to efficiently distinguish between these choices.

An agent can argue with additional information even if it is not related with the current argument in order to block opponents future arguments at an early stage. For example, if at t_1, Alice presents two arguments as $R_1 : A \Rightarrow B$ and $R_2 :\Rightarrow \neg D$, which is defended by Bob at t_2 by $R_3 : C \Rightarrow \neg B$, thus R_2 is strengthened into a fact at t_3. Now at t_3, Alice passes argument $R_4 : E \Rightarrow B$. Thus, at t_4 Bob has only one argument to defend as $R_5 : D \Rightarrow \neg B$. Bob cannot put forward argument R_5 as R_2 is a fact and stronger than R_5. Hence Alice wins. This will save one step as if Alice had not passed R_2 at t_1, Bob will present R_4 at t_4 and it has to play R_2 at t_5.

5 Related Work

Modelling dialogue games in defeasible logic has been addressed by [10,9,11], and the present paper builds on some ideas of [10,9]. [11] focuses on persuasion dialogues and it includes in the process cognitive states of agents such as knowledge and beliefs. In addition it presents some protocols for some types of dialogues (e.g., information seeking, explanation, persuasion). The main reasoning mechanism is based on basic defeasible logic (see Section 3) and it ignores recent development in extensions of defeasible logic with modal and epistemic operators for representing the cognitive states of agents [6,7], and it does not cover adversarial dialogues. [10] provides an extension of defeasible logic to include the step of the dialogue in a very similar to what we have presented in the paper. A main difference is that the resulting mechanism just defines a metaprogram for an alternative computation algorithm for ambiguity propagating defeasible logic while the logic presented here is ambiguity blocking. In [9], the authors focus on rule scepticism and proposes to use a sequences of defeasible (meta) theories, and use meta-reasoning (meta-rules or high level rules) to assess the strength of rules for the theories at lower levels.

Inference System (IS) was proposed in [15] to capture dialogue games. A theory in IS is represented by $T_{IS} = (R, \leq)$ where R is set of strict and defeasible rules and $\leq$ is a partial preorder which resolves any conflicts on precedence between rules. Arguments are *justified*, *overruled* and *defensible* depending on the outcome of the dialogue game. [16] describes the burdens associated with IS. According to [16], there are three kinds of burden as (1) *Burden of persuasion* (2) *Burden of production* and (3) *Tactical Burden of proof*. In [15], players have fixed roles as the burden of prosecution lies on the proponent, leaving the opponent with the burden to *interfere*. [14] modified IS as it proposes switching of roles in *Litigation inference system* (LIS). A theory in LIS is represented as $T_{LIS} = (R, \leq, b_\pi, b_\delta)$ where $(R, \leq)$ is an IS theory. b_π, b_δ are burden of prosecution for proponent and opponent respectively. [17] modifies LIS and proposes *Augmented Litigation inference system* (ALIS) which generates the content of b_π, b_δ as a result of an argument-based reasoning. A theory in ALIS is represented as $T_{ALIS} = (R, \leq)$, where $(R, \leq)$ is an IS theory described by a language which has a predicate *burden*. $burden(p, l)$ means that on the player p is placed the burden of prosecution for the literal l. A dialogue move m has three components: (1) *pl(m)*, the player who made the move (2) *r(m)*, the role of the player, and (3) *a(m)* the argument put forward in the move. ALIS imposes a protocol to be followed by the players. The protocol in ALIS differs from the protocol proposed in LIS in the sense that (1) if in the adjacent previous step the opponent *weakly defeated* an argument proposed by the proponent, then in the current step the proponent can argue that the opponent now has a burden on that literal; and (2) If the players weakly defeat each other while in their opponent role, then the *plaintiff* can argue that *defendant* has the burden of proof.

6 Conclusion and Future Work

We have presented a dialogue game framework in defeasible logic. We have shown that strength of an unchallenged rule can be upgraded in a dialogue game. We plan to extend the framework to cover more types of dialogue games and to model the different types of burden of proof. We also intend to study cost functions related of to dialogue games and investigate how a strategy can be developed with the aim of maximising the payoff of the game [19]. In addition, this framework could be extended to model the behaviour of an agent φ in a dynamic environment ε. By representing the environment as one of the parties, the agent φ is enabled, by putting forward arguments, to reason on its environment in a both reactive and a proactive way. This would allow for a natural characterisation of the environment as the uncertainty in the environment can be modelled as the private knowledge of ε.

References

1. Antoniou, G., Billington, D., Governatori, G., Maher, M.J.: Representation results for defeasible logic. ACM Transactions on Computational Logic 2(2), 255–287 (2001)
2. Antoniou, G., Billington, D., Governatori, G., Maher, M.J., Rock, A.: A family of defeasible reasoning logics and its implementation. In: Proc. ECAI 2000, pp. 459–463 (2000)
3. Billington, D.: Defeasible logic is stable. Journal of Logic and Computation 3, 370–400 (1993)
4. Dung, P.M.: On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming, and n–person games. Artificial Intelligence 77, 321–357 (1995)
5. Governatori, G., Maher, M.J., Billington, D., Antoniou, G.: Argumentation semantics for defeasible logics. Journal of Logic and Computation 14(5), 675–702 (2004)
6. Governatori, G., Rotolo, A.: Defeasible logic: Agency, intention and obligation. In: Lomuscio, A.R., Nute, D. (eds.) DEON 2004. LNCS (LNAI), vol. 3065, pp. 114–128. Springer, Heidelberg (2004)
7. Governatori, G., Rotolo, A., Padmanabhan, V.: The cost of social agents. In: Proc. AAMAS 2006, pp. 513–520. ACM Press, New York (2006)
8. Grosof, B.N.: Representing e-commerce rules via situated courteous logic programs in RuleML. Electronic Commerce Research and Applications 3(1), 2–20 (2004)
9. Lundström, J.E., Hamfelt, A., Nilsson, J.F.: A Rule-Sceptic Characterization of Acceptable Legal Arguments. In: Proc. ICAIL 2007, pp. 283–284. ACM Press, New York (2007)
10. Hamfelt, A., Lundström, J.E., Nilsson, J.F.: A metalogic formalization of legal argumentation as game trees with defeasible reasoning. In: Proc. ICAIL 2005, pp. 250–251. ACM Press, New York (2005)
11. Letia, I.A., Varic, R.: Defeasible protocols in persuasion dialogues. In: Proc. WE-IAT 2006, IEEE Computer Society Press, Los Alamitos (2006)
12. Maher, M.J., Rock, A., Antoniou, G., Billington, D., Miller, T.: Efficient defeasible reasoning systems. International Journal of Artificial Intelligence Tools 10(4), 483–501 (2001)
13. Nute, D.: Defeasible logic. In: Handbook of Logic in Artificial Intelligence and Logic Programming, vol. 3, pp. 353–395. Oxford University Press, Oxford (1994)
14. Prakken, H.: Modelling defeasibility in law:logic or procedure? In: Fundamenta informaticae, pp. 253–271 (2001)
15. Prakken, H., Sartor, G.: Rules about rules: Assessing conflicting arguments in legal reasoning. In: Artificial Intelligence and Law, pp. 331–368 (1996)
16. Prakken, H., Sartor, G.: Presumaptions and burdens of proof. In: Proc. Jurix 2006, pp. 21–30. IOS Press, Amsterdam (2006)
17. Prakken, H., Sartor, G.: Formalising arguments about the burden of persuation. In: Proc. ICAIL 2007, pp. 97–106. ACM Press, New York (2007)
18. Prakken, H.: Relating protocols for dynamic dispute with logics for defeasible argumentation. Synthese 127, 187–219 (2001)
19. Roth, B., Riveret, R., Rotolo, A., Governatori, G.: Strategic Argumentation: A Game Theoretical Investigation. In: Proc. ICAIL 2007, pp. 81–90. ACM Press, New York (2007)

Implementing Iterated Belief Change Via Prime Implicates

Zhi Qiang Zhuang[1], Maurice Pagnucco[2], and Thomas Meyer[3]

[1] School of Computer Science and Engineering, The University of New South Wales, Sydney, NSW 2052, Australia
zqzh390@cse.unsw.edu.au

[2] ARC Centre of Excellence for Autonomous Systems and National ICT Australia, School of Comp. Sci. and Eng., The University of New South Wales, Sydney, NSW 2052, Australia
morri@cse.unsw.edu.au
http://www.cse.unsw.edu.au/~morri/

[3] National ICT Australia and School of Computer Science and Engineering, The University of New South Wales, Sydney, Australia
Thomas.Meyer@nicta.com.au
http://www.cse.unsw.edu.au/~tmeyer/

Belief change is concerned with modelling the way in which an idealised (rational) reasoner maintains their beliefs and the way in which those beliefs are modified as the reasoner acquires new information. The AGM [1,3,5] framework is the most widely cited belief change methodology in the literature. It models the reasoner's belief state as a set of sentences that is logically closed under deduction and provides for three belief change operations: expansion, contraction and revision. Each of the AGM belief change operations is motivated by *principles of rationality* that are formalised by way of *rationality postulates*.

Pagnucco [10] formalised a way of implementing the AGM belief change operations using a *knowledge compilation* technique involving *prime implicates* in order to improve computational efficiency. This technique exploits the *epistemic entrenchment* construction for AGM belief change by Gärdenfors and Makinson [4] by introducing the notion of a *compiled epistemic entrenchment*. It has a number of significant features: (a) the belief change operators constructed satisfy the AGM postulates; (b) when compilation has been effected only subsumption checking and some simple syntactic manipulation is required in order to contract (or revise) the reasoner's belief state.

The aim of this paper is twofold. Firstly, to supply algorithms for Pagnucco's [10] formally specified prime-implicate technique for AGM belief change thus facilitating implementation. Secondly, to provide an empirical analysis of a Java-based implementation of these algorithms (and thus the formal technique).

Implementations of AGM belief change include the work of Dixon and Wobcke [2] and Williams [16] who adopt theorem proving techniques to realise each of the operations. The work here can be seen as an attempt to improve the computational efficiency of implementations like these through the use of knowledge compilation. Prime implicates have been used in *truth maintenance systems* by Reiter and de Kleer [11] and by Kean [8]. However these approaches lack any preferential structure that allows them to be used for more general forms of belief change. Gorogiannis and Ryan [6] use an alternate form of knowledge compilation—binary decision diagrams (BDDs)—but focus on the complexity of specific operations.

1 Background

We assume a fixed finite propositional object language $\mathcal{L}$ with the standard logical connectives. Cn represents the classical consequence operator while $\vdash$ represents the classical consequence relation (i.e., $\Gamma \vdash \phi$ iff $\phi \in Cn(\Gamma)$). We also adopt the following linguistic conventions to simplify the presentation. Lower case Greek letters ϕ, ψ, χ, ...denote sentences of $\mathcal{L}$. Upper case Greek letters Δ, Γ, ...denote sets of formulas. Theories (closed under logical consequence) are used to represent reasoners' belief states in AGM belief change and will be denoted by upper case Roman characters H, K, ...(i.e., $K = Cn(K)$) that are termed *belief sets*. The inconsistent belief set is denoted $K_\perp$. Lower case Roman characters l, k, ...denote literals (both positive and negative). Upper case Roman characters C, D, ...denote clauses.

Prime implicates are minimal length clauses (in terms of set inclusion) implied by a knowledge base. Transforming a knowledge base into a set of prime implicates gives a uniform, and logically equivalent, way of expressing the sentences in the knowledge base that can be exploited to enhance computational efficiency. We denote the set of prime implicates of an arbitrary set of sentences Γ by $\Pi(\Gamma)$. When Γ is a set of clauses, prime implicates are easily computed by repeated application of resolution and removing subsumed (i.e., implied) clauses [15]. This strategy can be improved by using incremental methods [9,7], and the use of efficient data structures [11]. Compiling a knowledge base into prime implicate form can lead to an exponential number of implicates in the number of atoms (see [13]).

1.1 AGM Belief Change

Alchourrón, Gärdenfors and Makinson (AGM) [1,3,5] introduced one of the most widely adopted belief change frameworks. The AGM considers three types of transformations on beliefs sets as new information is acquired: *belief expansion* by ϕ in which this new information is added to the initial belief set K without removal of any existing beliefs ($K + \phi$); *belief contraction* by ϕ where belief in ϕ is suspended ($K \dot{-} \phi$); and, *belief revision* by ϕ where ϕ is assimilated into K while existing beliefs may need to be suspended in order to maintain consistency ($K * \phi$). These operations are motivated by *rationality criteria* and characterised by a set of *rationality postulates* which—for the sake of brevity—we do not reproduce here but rather refer the interested reader to the AGM account [1,3,5].

One element of the AGM that does interest us here is the notion of *epistemic entrenchment* that can be used to construct a contraction or revision function satisfying the AGM rationality postulates. An epistemic entrenchment is an ordering over the sentences in K that can be used to determine which sentences to remove during belief contraction and revision. Formally, an epistemic entrenchment relation $\leq$ satisfies the following properties [4].

(EE1) If $\phi \leq \psi$ and $\psi \leq \gamma$ then $\phi \leq \gamma$
(EE2) If $\{\phi\} \vdash \psi$ then $\phi \leq \psi$

(EE3) For any ϕ and ψ, $\phi \leq \phi \wedge \psi$ or $\psi \leq \phi \wedge \psi$
(EE4) When $K \neq K_\perp$, $\phi \notin K$ iff $\phi \leq \psi$ for all ψ
(EE5) If $\phi \leq \psi$ for all ϕ then $\vdash \psi$

(EE1)–(EE3) entail that $\leq$ is a total pre-order. In other words, a ranking of sentences. (EE4) says that non-beliefs are minimally entrenched while (EE5) says that tautologies are maximally entrenched. The following condition [4] tells us how to define a contraction function using epistemic entrenchment.

$$(C\dot{-}) \quad \psi \in K\dot{-}\phi \text{ iff } \psi \in K \text{ and either } \phi < \phi \vee \psi \text{ or } \vdash \phi$$

In the principal case a belief ψ is retained when contracting K by ϕ provided that there is additional evidence to support it ($\phi < \phi \vee \psi$).

1.2 Compiling Epistemic Entrenchment

Pagnucco [10] introduces a theory to reduce the amount of information required to construct an AGM contraction function via epistemic entrenchment by compiling the relation using prime implicates. An observation by Rott [12] is central to this idea.

Proposition 1. *[12] For arbitrary sentence $\psi \in \mathcal{L}$, $\{\phi : \psi \leq \phi\} = Cn(\{\phi : \psi \leq \phi\})$.*

Put simply, if we were to "cut" the epistemic entrenchment relation at any level, the beliefs that are at least this entrenched would form a set that is deductively closed (i.e., a belief set)[1].

A compiled epistemic entrenchment is essentially one in which we take each cut and replace it by its prime implicates thus significantly reducing the number of sentences required to be explicitly represented in the entrenchment.

Definition 1. *($\leq_\Pi$)*

Given an epistemic entrenchment ordering $\leq$ satisfying properties (EE1)–(EE5) we define a compiled epistemic entrenchment ordering $\leq_\Pi$ as follows. For any two clauses C, D, $C \leq_\Pi D$ iff all of the following hold:

1. *$C \leq D$;*
2. *$C \in \Pi(\{\phi : \phi \leq \psi\})$ for some $\psi \in \mathcal{L}$; and,*
3. *$D \in \Pi(\{\phi : \phi \leq \chi\})$ for some $\chi \in \mathcal{L}$*

Note that the empty clause is less entrenched than all clauses which we denote $\perp \leq C$.

We omit the theorems that show the correctness of this and most of the following definitions as they can be found in Pagnucco [10].

One concept we require is the (maximal) level of entrenchment of a clause.

Definition 2. *Let C be a clause, $max_{\leq_\Pi}(C) = \{D :$ such that*

1. *D is a clause where $D \subseteq C$, and*
2. *there is no other clause $D' \subset C$ and $D \leq_\Pi D'$.$\}$*

[1] The term "cut" is due to Rott [12] and we shall adopt it here.

We can now define AGM contraction using the compiled entrenchment by appropriately modifying condition (C$\dot{-}$). The following definition deals with contraction of K by a single clause and we shall show how to extend it shortly. It is a simplified version of the definition in [10]. Note that while $max_{\leq_\Pi}(C)$ is a set of clauses, all these clauses are at the same level of entrenchment and so we abuse this notation in the following by allowing this set to be represented by any one of its elements.

Definition 3. *Given an epistemic entrenchment relation $\leq$ and a clause C we define the contracted compiled epistemic entrenchment $\leq_\Pi^C$ of a compiled epistemic entrenchment $\leq_\Pi$ by C as: $D \leq_\Pi^C E$ iff either*

1. *$D \leq E$ and $max_{\leq_\Pi}(C) < max_{\leq_\Pi}(C \cup D), max_{\leq_\Pi}(C \cup E)$, or*
2. *$A \leq B$ and $A = C \cup F$ with $max_{\leq_\Pi}(C) < C \cup D \cup F$ and $B = E \cup G$ with $max_{\leq_\Pi}(C) < C \cup E \cup G$ (note F and G may be empty).*

Condition (1) uses (C$\dot{-}$) directly to determine when clauses in the original compiled epistemic entrenchment should be retained. The second condition determines whether clauses that do not satisfy condition (1) and are removed should be replaced by weaker clauses. If we do not allow replacement by weaker clauses, the AGM postulates will not be satisfied. We see that this definition is correct as far as single clauses are concerned.

Theorem 1. *[10] Let K be a belief set, $\dot{-}$ an AGM contraction function satisfying (K$\dot{-}$1)–(K$\dot{-}$8) and $\leq$ an epistemic entrenchment relation defined from $\dot{-}$ by condition (C$\leq$). Furthermore, let C be a clause. Then $K \dot{-} C = Cn(\{D : \bot <_\Pi^C D\})$.*

Pagnucco uses some properties of epistemic entrenchment to show that contraction by an arbitrary formula can be easily achieved by converting the formula into CNF and contracting by the minimally entrenched conjuncts.

Definition 4. *Let $\phi \in \mathcal{L}$, the set of minimal conjuncts of ϕ is defined as follows: $min_{\leq_\Pi}(CNF(\phi)) = \{C_i \in CNF(\phi) : max_{\leq_\Pi}(C_i) \leq_\Pi max_{\leq_\Pi}(C_j)$ for all $C_j \in CNF(\phi)\}$.*

We now extend our previous result to show that this produces the desired contraction.

Theorem 2. *[10] Let K be a belief set, $\dot{-}$ an AGM contraction function satisfying (K$\dot{-}$1)–(K$\dot{-}$8) and $\leq$ an epistemic entrenchment relation defined from $\dot{-}$ by condition (C$\leq$). Furthermore, let ϕ be a sentence.*
Then $K \dot{-} \phi = \bigcap_{C_i \in min_{\leq_\Pi}(CNF(\phi))} Cn(\{D : \bot <_\Pi^{C_i} D\})$.

Belief revision can be implemented by combining contraction and expansion via the Levi Identity: $K * \phi = (K \dot{-} \neg\phi) + \phi$ where AGM expansion is simply deductive closure (i.e., $K + \phi = Cn(K \cup \{\phi\})$).

2 Algorithms

The basis for the knowledge compilation approach we adopt is the compiled epistemic entrenchment relation. As noted above it is a total pre-order over clauses (i.e., a ranking). We model this pre-order using ranked levels identified by integers; the greater the

Algorithm 1. Algorithm for finding the level of entrenchment of a clause C in belief set K represented as a compiled epistemic entrenchment relation. On termination the algorithm returns the level at which the clause C is entrenched, -1 if the belief set K does not contain C or ∞ if C is a tautology.

```
FIND-LEVEL($K, C$)
1   **if** $C$ is a tautology **then**
2       **return** $\infty$
3   $L \leftarrow L_K$
4   **while** $L$ is not empty **do**
5       $m \leftarrow$ highest level in $L$
6       $L \leftarrow L \setminus \{m\}$
7       **for** each clause $D \in K(m)$ **do**
8           **if** $D \subseteq C$ **then**
9               **return** $m$
10  **return** $-1$
```

integer, the higher the level of epistemic entrenchment. Each formula (implicate) has an associated integer level of entrenchment.

This means that the belief change operations of expansion and revision which involve the addition of information, require that the level of entrenchment for the formula representing the new information be supplied as well. This is not how the respective AGM operations work however we note that this is required only to allow for iterated belief change. If we were only interested in a single expansion or revision, then this level of entrenchment would not be required.

We adopt the following notation to simplify the algorithms in this section. For a belief set K we denote by $K(n)$ the set of clauses *explicitly* stored at level n in the compiled epistemic entrenchment. Similarly, $K(n+)$ denotes the set of clauses explicitly retained at level n or greater while $K(n-)$ denotes those clauses explicitly retained at level n or less. L_K is a set of integers denoting the levels of the compiled epistemic entrenchment at which clauses are explicitly maintained. $L_K(n+)$ (respectively, $L_K(n-)$), denotes the set of integer levels in the compiled entrenchment containing clauses greater than or equal to (respectively, less than or equal to) n. A function *computePI()* takes a set of clauses and returns the corresponding prime implicates.

2.1 Find Level

The first algorithm that we require is one to determine the rank (i.e., level) of a clause in the compiled epistemic entrenchment relation. This is required for belief expansion and also to implement belief contraction using the ($C\dot{-}$) condition.

The heart of Algorithm 1 is lines 4–9. Starting with the highest level of the compiled entrenchment[2] the algorithm looks through each successive level of entrenchment

[2] Keep in mind that the compiled entrenchment—unlike AGM entrenchment—does not explicitly maintain tautologies.

Algorithm 2. Algorithm for expanding belief set K by clause C at level n

```
EXPANSION($K, C, n$)
1  $h \leftarrow$ FIND-LEVEL($K, C$)
2  if $h \geq n$ then
3      return
4  if $n \notin L_K$ then
5      $L_K \leftarrow L_K \cup \{n\}$
6  $L \leftarrow L_K(n-)$
7  while $L$ is not empty do
8      $m \leftarrow$ highest level in $L$
9      $L \leftarrow L \setminus \{m\}$
10     if $m = n$ then
11         $P \leftarrow computePI(K(m+) \cup \{C\})$
12     else
13         $P \leftarrow computePI(K(m+))$
14     if $P$ is empty then
15         $K \leftarrow K_\perp$
16         return
17     $N \leftarrow P \setminus K(m+)$
18     for each clause $D \in K(m-)$ do
19         for each clause $D' \in N$ do
20             if $D' \subseteq D$ then
21                 remove $D$ from $K$
22     add all clauses in $N$ to level $m$ of $K$
```

to determine whether the clause in question is subsumed by one of the implicates at that level. As soon as such an implicate is found, the level at which this implicate is entrenched corresponds to the level at which the clause is entrenched by Proposition 1.

2.2 Belief Expansion

Belief expansion using a compiled entrenchment differs slightly from AGM expansion because we are interested in iterated belief change. This is in keeping with implementations like those of Dixon and Wobcke [2] and Williams [16] and the belief change framework of Spohn [14]. Accordingly, the belief expansion operation takes three inputs: a belief set K, a clause C to be added and a level n at which it is to be added to the compiled entrenchment. Note that if a formula is to be added rather than a clause, the formula can be converted to CNF and each clause added at that level.

Lines 2–3 of Algorithm 2 determine whether the clause is already implied by the compiled entrenchment at a level greater than or equal to n and, if so, nothing is required to be done. Lines 4–5 concern the case in which the clause needs to be added at a level that does not explicitly exist in the compiled entrenchment in which case n is simply added to the levels in L_K. The remaining lines 7–22 take care of adding C to the

Algorithm 3. Algorithm for contracting a belief set K by a single clause C. On termination it returns the resulting compiled entrenchment for belief set $K\dot{-}C$.

```
SINGLE-CLAUSE-CONTRACTION($K, C$)
1  $n \leftarrow$ FIND-LEVEL($K, C$)
2  if $n = -1$ or $n = \infty$ then
3      return $K$
4  remove $C$ from $K$
5  $L \leftarrow L_K(n-)$
6  while $L$ is not empty do
7      $m \leftarrow$ highest level in $L$
8      $L \leftarrow L \setminus \{m\}$
9      $N \leftarrow \{\}$
10     for each clause $D \in K(m)$ do
11         $h \leftarrow$ FIND-LEVEL($K, C \cup D$)
12         if $n = h$ then
13             remove $D$ from $K$
14             for each literal $l \in C$ do
15                 $k \leftarrow$ FIND-LEVEL($K, \{\neg l\} \cup D$)
16                 if $k = m$ then
17                     $N \leftarrow N \cup \{\{\neg l\} \cup D\}$
18             for each clause $E$ in $K((n+1)+)$ do
19                 if $C \cup D \subseteq E$ then
20                     $E \leftarrow (E \setminus C) \cup D$
21                     $N \leftarrow N \cup \{E\}$
22     $P \leftarrow computePI(N \cup K(m+))$
23     $P \leftarrow P \setminus K(m+)$
24     add all clauses in $P$ to level $m$ of $K$
25 return $K$
```

compiled entrenchment and re-computing the prime implicates at each level less than or equal to n. Line 11 is executed the first time through the loop, adding C to the compiled entrenchment at level n while line 12 is executed on subsequent iterations. Lines 11 and 12 re-compute the prime implicates at each level. Lines 17–21 take care of removing any subsumed clauses from the compiled entrenchment to ensure that only those clauses that need to be represented are maintained. Lines 14–16 handle the case of expansion into inconsistency. Note that the function *computePI()* can be implemented using an incremental technique [7,9] to reduce the amount of computation required.

2.3 Belief Contraction

As with the theoretical development of belief contraction using a compiled epistemic entrenchment relation we begin by considering the algorithm for belief contraction by a single clause C. Algorithm 3 uses Definition 3 which is motivated by Gärdenfors and

Algorithm 4. Algorithm for intersecting two compiled entrenchments representing belief sets K_1 and K_2. On termination it returns the intersection of the two compiled entrenchments.

```
INTERSECTION($K_1, K_2$)
 1  $K \leftarrow$ empty belief set
 2  $L_1 \leftarrow L_{K_1}$
 3  $L_2 \leftarrow L_{K_2}$
 4  for each level $n \in L_2$
 5      if $n \notin L_1$ then
 6          $L_1 \leftarrow L_1 \cup \{n\}$
 7  while $L_1$ is not empty do
 8      $m \leftarrow$ highest level in $L_1$
 9      $L_1 \leftarrow L_1 \setminus \{m\}$
10      $N \leftarrow \{\}$
11      for each clause $C \in K_1(m)$ do
12          for each clause $D \in K_2(m+)$ do
13              if $C \subseteq D$ and FIND-LEVEL$(K, D) = -1$ then
14                  $N \leftarrow N \cup \{D\}$
15      for each clause $C \in K_2(m)$ do
16          for each clause $D \in K_1(m+)$ do
17              if $C \subseteq D$ and FIND-LEVEL$(K, D) = -1$ then
18                  $N \leftarrow N \cup \{D\}$
19      $P \leftarrow computePI(N)$
20      add all clauses in $P$ to level $m$ of $K$
21  return $K$
```

Makinson's condition (C$\dot{-}$). Lines 1–3 determine whether it is necessary to perform contraction. Definition 3 and condition (C$\dot{-}$) tell us that clauses at a higher level of entrenchment to C are unaffected by contraction. The while loop encompassing lines 6–24 considers the clauses explicitly entrenched at levels less than or equal to that of C in the compiled entrenchment, determining which are to be removed and whether they are replaced by weaker clauses in order to adhere to the AGM postulates. Definition 3(2) (and Condition (C$\dot{-}$)) tell us that a clause D in the compiled entrenchment that is at a level less than or equal to that of clause C should be removed whenever $C = C \vee D$ which is the condition tested on lines 11 and 12 and the clause is subsequently removed at line 13. Having determined to remove the clause we now need to determine whether the clause D should be replaced by logically weaker clauses. This is done in lines 18–21 of the algorithm. Lines 14–17 add clauses that are required by the Recovery postulate (K$\dot{-}$5). The remaining lines 22–24 re-compute the prime implicates as required.

To contract a compiled entrenchment by an arbitrary sentence requires a slightly special form of intersection that takes into account subsumed clauses. This is in keeping with our desire to implement iterated belief change. Lines 4–7 ensure that all levels of both compiled entrenchments are considered (which is required for subsumed clauses). A clause C at level m of K_1 is checked against all clauses at level j of K_2 for all $j \geq m$.

If there are clauses that C subsumes but C is not in the resulting belief set K then the subsumed clause is added to level m of K since it is minimal. As the same procedure has to be performed for both belief sets, we have two for loops in lines 11–14 and 15–18 of the algorithm that iterate over each of the compiled entrenchments for both belief sets achieving this task. Finally, in lines 19–20 prime implicates are re-computed.

We are now in a position to implement the algorithm to contract by arbitrary sentences converted into CNF. By Theorem 1 we only need to consider the minimally entrenched clauses in the CNF of the formula, contract K by each in turn and take the intersection of the result. Algorithm LEAST-ENTRENCHED-CLAUSE(K, S) takes care of determining the least entrenched clauses while ARBITRARY-SENTENCE-CONTRACTION(K, S) performs the individual contractions and takes the required intersections using the preceding algorithm.

Algorithm 5. Algorithm for contracting belief set K by an arbitrary sentence S in CNF

```
LEAST-ENTRENCHED-CLAUSE$(K, S)$
1   $N \leftarrow \{\}$
2   for each clause $C \in S$ do
3       if $N \leftarrow \{\}$ then
4           $N \leftarrow \{C\}$
5       else
6           $D \leftarrow$ first element in $N$
7           if FIND-LEVEL$(K, C) >$ FIND-LEVEL$(K, D)$ then
8               $N \leftarrow \{C\}$
9           elif FIND-LEVEL$(K, C) =$ FIND-LEVEL$(K, D)$ then
10              $N \leftarrow N \cup \{C\}$
11  return $N$

ARBITRARY-SENTENCE-CONTRACTION$(K, S)$
1   $I \leftarrow$ empty belief set
2   $N \leftarrow$ LEAST-ENTRENCHED-CLAUSE$(K, S)$
3   for each clause $D \in N$ do
4       $K^{'} \leftarrow K$
5       $I^{'} \leftarrow$ SINGLE-CLAUSE-CONTRACTION$(K^{'}, D)$
6       if $I$ is empty belief set then
7           $I \leftarrow I^{'}$
8       else
9           $I \leftarrow$ INTERSECTION$(I, I^{'})$
10  $K \leftarrow I$
```

Note that belief revision is now easily achieved by combining belief contraction and belief expansion via the Levi Identity ($K * \phi = (K \dot{-} \phi) + \phi$) so we omit a specific algorithm for this straightforward operation.

3 Analysis

The foregoing algorithms were implemented in Java and subjected to empirical analysis as we describe here. The experiments performed were conducted on randomly generated 3-clauses using belief revision rather than just expansion or contraction. The reason for this latter decision is that using expansion alone may lead to an inconsistent belief set and contraction may involve a lot of degenerate cases (attempting to contract a clause that is not implied by the compiled entrenchment) which would skew the results. Adopting revision is guaranteed to require one of expansion or contraction to be performed and possibly both.

In the first experiment the compiled entrenchment is continually revised by a randomly generated 3-clause which is added at a random level of entrenchment between 1 and 20. Each 3-clause is generated from a vocabulary of 5 propositional letters to ensure that there is a reasonable chance for interaction among the clauses added to the compiled entrenchment (otherwise the experiments will tend to be uninteresting). For each number of clauses 50 trials were conducted and the average time taken recorded. The resulting graph is show in Figure 1(a). We see that initially, with few clauses, the average running time increases dramatically. At about 50 clauses the graph begins to level off. This indicates that there is indeed an advantage to compiling the entrenchment relation using prime implicates. After some initial work, changes tend to be minor. Note however that our choice to restrict the language to 5 propositional letters may also be a factor as after some time, clauses may be more likely to either already occur in the compiled entrenchment. This requires no work in terms of revision. Note however that there is also an increased likelihood of conflict occurring.

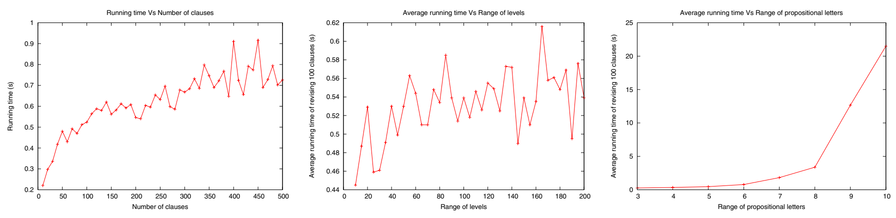


Fig. 1. (a) Running time vs number of clauses; (b) running time vs range of levels; (c) running time vs range of propositional letters

The second experiment involves adjusting the coarseness of the compiled entrenchment by varying the maximum number of levels of entrenchment allowed. In this case the compiled entrenchment is revised by 100 randomly generated 3-clauses (again, chosen from a vocabulary of 5 propositional letters) but with the maximum number of levels increasing by 10 each time. 50 trials were conducted for each data point. While there is a reasonable fluctuation in the graph—shown in Figure 1(b)—a closer look reveals a general upward trend. This is to be expected. In conducting either expansion or contraction, the crucial phase is to re-compute prime implicates. For expansion this occurs at all levels less than or equal to the one at which the clause is added. For contraction it occurs at all levels less than or equal to the one at which the clause resides. With more levels of entrenchment, the possibility of having to do this over a greater number of levels increases. Even with the use of an incremental algorithm for re-computing prime implicates this incurs a significant cost.

The final experiment varies the range of propositional letters used to generate 3-clauses from 3 to 10. Again 50 trials are conducted for each data point with 100 3-clauses generated for revision. As expected we observe a significant increase in the average running time as the number of propositional letters increases. However, we have not at this stage attempted many optimisations in (re-)computing prime implicates which may help to ameliorate this situation.

4 Conclusions

In this paper we have introduced algorithms that implement Pagnucco's [10] formal framework for implementing AGM belief revision through the use of prime implicates to compile the epistemic entrenchment relation. We have also subjected these algorithms to an initial empirical analysis. This analysis indicates that the compilation of the epistemic entrenchment relation can lead to an improvement in the average running time for belief revision. Further empirical analysis is expected to confirm these findings.

Apart from a larger range of experiments, further work involves trying to optimise parts of the algorithms we have presented, in particular the generation of prime implicates and the cost of subsumption checking. These can be achieved through a better choice of data structures and the employment of heuristic techniques commonly used in modern SAT solvers. Another avenue for future research is the extension of our algorithms to first-order logic where the definition of prime implicates becomes more problematic but where a restricted definition may afford some purchase.

References

1. Alchourrón, C.E., Gärdenfors, P., Makinson, D.: On the logic of theory change: Partial meet contraction and revision functions. J. of Symbolic Logic 50, 510–530 (1985)
2. Dixon, S.E., Wobcke, W.: The implementation of a first-order logic AGM belief revision system. In: Proc. of the Fifth IEEE Int. Conf. on Tools in Art. Int. (1993)
3. Gärdenfors, P.: Knowledge in Flux: Modeling the Dynamics of Epistemic States. Bradford Books, MIT Press, Cambridge Massachusetts (1988)
4. Gärdenfors, P., Makinson, D.: Revisions of knowledge systems using epistemic entrenchment. In: Proc. of 2nd Conf. on Th. Aspect of Reas. About Knowl., pp. 83–96 (1988)
5. Gärdenfors, P., Rott, H.: Belief revision. In: Handbook of Logic in AI and Logic Programming vol. IV: Epistemic and Temporal Reasoning, OUP, pp. 35–132 (1995)
6. Gorogiannis, N., Ryan, M.D.: Implementation of belief change operators using BDDs. Studia Logica 70(1), 131–156 (2002)
7. Jackson, P.: Computing prime implicates incrementally. In: Proceedings of the Eleventh Conference on Automated Deduction (June 1992)
8. Kean, A.: A formal characterisation of a domain independent abductive reasoning system. Technical Report HKUST-CS93-4, Dept. of Computer Science, HKUST (1993)
9. Kean, A., Tsiknis, G.: An incremental method for generating prime implicants/implicates. Journal of Symbolic Computation 9, 185–206 (1990)

10. Pagnucco, M.: Knowledge compilation for belief change. In: Sattar, A., Kang, B.-H. (eds.) AI 2006. LNCS (LNAI), vol. 4304, pp. 90–99. Springer, Heidelberg (2006)
11. Reiter, R., de Kleer, J.: Foundations of assumption-based truth maintenance systems: Preliminary report. In: Proc. of the Nat. Conf. in AI, pp. 183–188 (1987)
12. Rott, H.: Preferential belief change using generalized epistemic entrenchment. Journal of Logic, Language and Information 1, 45–78 (1992)
13. Schrag, R., Crawford, J.M.: Implicates and prime implicates in random 3-SAT. Artificial Intelligence 81(1-2), 199–222 (1996)
14. Spohn, W.: Ordinal conditional functions: A dynamic theory of epistemic states. In: Causation in Decision, Belief Change, and Statistics, II, pp. 105–134. Kluwer, Dordrecht (1988)
15. Tison, P.: Generalization of consensus theory and application to the minimization of boolean functions. IEEE Trans. on Elec. Computers 4, 446–456 (1967)
16. Williams, M.-A.: Iterated theory change: A computational model. In: Proc. of the Fourteenth International Joint Conference on Artificial Intelligence, pp. 1541–1550 (1995)

Applying MCRDR to a Multidisciplinary Domain

Ivan Bindoff[1], Byeong Ho Kang[1], Tristan Ling[1], Peter Tenni[2], and Gregory Peterson[2]

[1] University of Tasmania, School of Computing
{ibindoff, bhkang, trling}@utas.edu.au
[2] University of Tasmania, Unit for Medical Outcomes and Research Evaluations
{Peter.Tenni, G.Peterson}@utas.edu.au

Abstract. This paper details updated results concerning an implementation of a Multiple Classification Ripple Down Rules (MCRDR) system which can be used to provide quality Decision Support Services to pharmacists practicing medication reviews (MRs), particularly for high risk patients. The system was trained on 126 genuine cases by an expert in the field; over the course of 19 hours the system had learned 268 rules and was considered to encompass over 80% of the domain. Furthermore, the system was found able to improve the quality and consistency of the medication review reports produced, as it was shown that there was a high incidence of missed classifications under normal conditions, which were repaired by the system automatically. However, shortcomings were identified including an inability to handle absent data, and shortcomings concerning standardization in the domain, proposals to solve these shortcomings are included.

1 Introduction

Sub-optimal drug usage is a serious concern both in Australia and overseas [1, 2], resulting in at least 80,000 hospital admissions annually - approximately 12% of all medical admissions - the majority of these concerning elderly patients [3]. MRs are seen as an effective way to improve drug usage. However, the quality of MRs produced is inconsistent across reviewers.

This paper continues discussion commenced in a earlier publications by the authors in 2006 and 2007 in which an Intelligent Decision Support System was developed in an attempt to improve the quality of MRs [4, 5]. It was suggested that to improve the consistency and quality of MRs it would be prudent to develop medication management software which includes Intelligent Decision Support features. Prior to this, the majority of incarnations of medication management software for producing MRs has lacked any genuinely "Intelligent" form of Decision Support features [6]. In response to this suggestion, a software system for medication management was developed that utilized the MCRDR method to provide Intelligent Decision Support Services in the multidisciplinary field of MR. [7, 8].

2 Medication Reviews

MR is a burgeoning area in Australia and other countries, with MRs seen to be an effective way of improving drug usage and reducing drug related hospital admissions, particularly in the elderly and other high risk patients [1, 3].

To perform a MR, Pharmacists assess potential Drug Related Problems (DRPs) in a patient by examining various patient records, primarily their medical history, any available pathology results, and their drug regime (past and current) [9].

The expert looks for a variety of indicators between the case details provided checking for known problems, such as an: Untreated Indication – where a patient has a medical condition which requires treatment but doesn't have the treatment; Contributing Drugs – where a patient has a condition and is on a drug which can cause or exacerbate said condition; High Dosage – where a patient is potentially on a too high dosage because of a combination of drugs with similar ingredients; Inappropriate Drug – where a patient is on a drug that is designed to treat a condition they don't seem to have or is contraindicated in their condition; and many others besides. Once these indicators have been identified a statement is produced explaining each problem, or potential problem, and often what the appropriate course of action is.

3 Methodology

In order to produce a medication management system with intelligent decision support features it was necessary to produce two major software elements. The first was a standard implementation of a database "front-end" from which it is possible for a user to enter all the details of a given patient's case, or at least those parts which are relevant to the chosen domain, and which was sufficiently computationally expressive to be machine understandable which shall be referred to after this point simply as the "Domain Model". The second was an implementation of a Multiple Classification Ripple Down Rules engine which can sufficiently encapsulate the types of conditions and knowledge required for the domain and facilitate the design of an interface from which the engine can be operated, particularly during the Knowledge Acquisition phase.

3.1 Domain Model

The design of the database to store the MR cases was considered to be relatively trivial, and was given relatively little consideration during initial development. The preliminary design was taken from existing medication management software packages, and then modified as required to allow for basic machine readability. The 126 cases considered in this study were then inserted into the database using a simple script which converted them from their current Mediflags [10] format.

3.2 Ripple Down Rules

Ripple Down Rules (RDR) is an approach to building KBSs that allows the user to incrementally build the knowledge base while the system is in use, with no outside assistance or training from a knowledge engineer [8]. It generally follows a forward-chaining rule-based approach to building a KBS. However, it differs from standard rule based systems since new rules are added in the context in which they are suggested.

Observations from attempts at expert system maintenance led to the realisation that the expert often provides justification for why their conclusion is correct, rather than providing the reasoning process they undertook to reach this conclusion. That is, they say ‘why’ a conclusion is right, rather than ‘how’. An example of this would be the expert stating “I know case X has conclusion 1 because they exhibit features r, g and n”. Furthermore, experts are seen to be particularly good at providing comparison between two cases and distinguishing the features which are relevant to their different classifications [11]. With these observations in mind an attempt was made at producing a system which mimicked this approach to reasoning, with RDR being the end result.

The resultant RDR structure is that of a binary tree or a decision list [12], with exceptions for rules which are further decision lists. The decision list model is more intuitive since, in practice, the tree would have a fairly shallow depth of correction [13]. The inferencing process works by evaluating each rule in the first list in turn until a rule is satisfied, then evaluating each rule of the decision list returned by that satisfied rule similarly until no further rules are satisfied. The classification that was bound to the last rule that was satisfied is given.

3.3 Multiple Classification Ripple Down Rules

The RDR method described above is limited by its inability to produce multiple conclusions for a case. To allow for this capability - as this domain must - MCRDR should be considered [14] to avoid the exponential growth of the knowledge base that would result were compound classifications to be used.

MCRDR is extremely similar to RDR, preserving the advantages and essential strategy of RDR, but able to return multiple classifications. Contrasting with RDR, MCRDR evaluates all rules in the first level of the knowledge base then evaluates the next level for all rules that were satisfied and so on, maintaining a list of classifications that should fire, until there are no more children to evaluate or none of the rules can be satisfied by the current case [13].

3.3.1 Knowledge Acquisition

Knowledge Acquisition is required when a case has been classified incorrectly or is missing a classification. It is divided into three separate steps: Acquiring new classification (or conclusion), locating the new rule, and acquiring the new rule.

Acquiring the new classification is trivial; the system merely prompts the expert to state it [13]. To acquire the new rule the expert is asked to first select valid conditions from the current case that indicate a given classification. The rule they have created thus far is then compared against the cornerstone case base. If any cornerstone cases would fire on this new rule the expert is asked to select extra condition(s) for the rule from a difference list (see Table 1) between the presented case and one of the cornerstone cases. A cornerstone case is a case for which the knowledge had previously been modified and which is valid under the current context [15]. The system then retests all cornerstone cases in the list against the appended set of conditions, removing cases from the list that are no longer satisfied. The system repeats this process until there are no remaining cornerstone cases in the list to satisfy the rule [13] or alternatively the expert has stated that the cornerstone cases that remain *should* fire on the new rule, indicating the classification was simply missed on it earlier.

Table 1. Example of a decision list from [8, 16-18]. The list can contain negated conditions

Cornerstone case	*Current test case*	*Difference list*
Rain	Rain, Meeting	Meeting
Meeting	Meeting	Not applicable
Hot		Not(Hot)

To determine where the new rule must go it must first be determined what type of wrong classification is being made. The three possibilities are listed in **Table 2**.

Table 2. The three ways in which new rules correct a knowledge base [13]

Wrong Classifications	To correct the Knowledge Base
Wrong classification to be stopped	Add stopping rule at the end of the path
Wrong classification replaced by correct	Add a rule at the end of the path
A new independent classification	Add a rule at the root

4 Results and Discussion

The system was handed over to the expert with absolutely no knowledge or conclusions pre-loaded. The expert was wholly responsible for populating the knowledge base. Over the course of ~19 hours they were able to add the rules required to correctly classify 126 genuine MR cases that had been pre-loaded into the system.

4.1 Growth of Knowledge Base

It is observed in Figure 1 that the number of rules in the system progressed linearly as more cases were analysed, at an average rate of 2.04 rules per case. Figure 1 suggests that the system was still learning heavily until around the 250th rule, at which point the learning rate began to drop off and a plateau began to develop. Previous RDR systems have been shown to produce a flattening pattern when the knowledge base reaches approximately 80% of domain coverage [13].

4.2 Correct Conclusions Found

It was estimated by the expert at cessation of the experiment that the system had encapsulated around 80% of the domain [19], this estimation is supported by the evidence shown in Figure 2. It can be seen that the average number of correct classifications the system provided rose quite steadily into the 80th percentile, although the percentage correct from case to case did vary quite a lot, as is to be expected when results are still largely influenced by the early heavy learning phase.

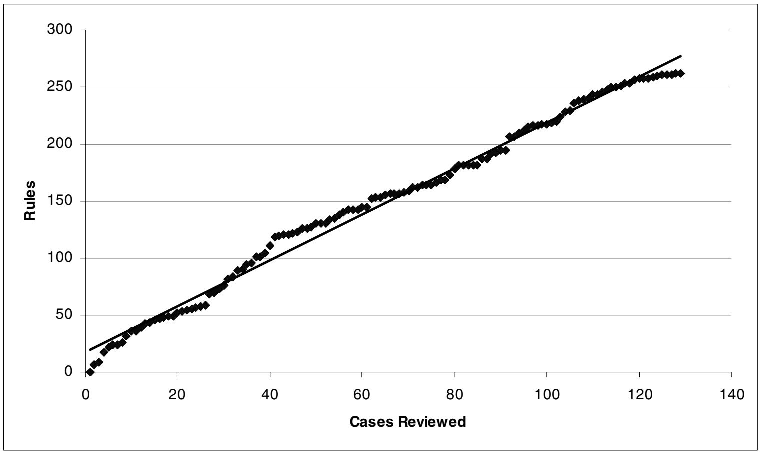


Fig. 1. The number of rules in the system grows linearly, but begins to flatten off

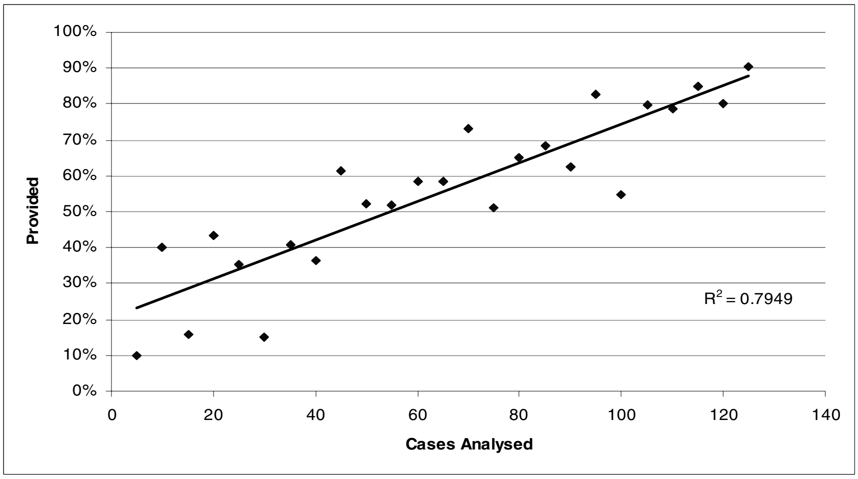


Fig. 2. The grouped average percentage of correct conclusions provided

The expert predicted potential classification rates in the order of 90% with this system [19], and it would appear that his estimation was reasonable. Past systems of this nature have demonstrated that despite a flattening pattern commencing, the domain coverage has still continued to grow by an additional 10-15%, although it is conceded that an extremely large number of cases had to be considered for this to be possible [13, 15]. These figures are justified by following the trend-line in Figure 2 which shows the average of correct conclusions provided by the system for each group of 5 cases analysed, although it is conceded that this trend-line is only a rough approximation. Even following the flattening pattern demonstrated in Figure 1 it is easy to imagine the system reaching the order of 90% or above, given another 50-150 cases to train with. It should be noted that this is not actually a large amount, when it is considered that it took only ~19 hours of expert time to classify the first 126 cases and it is expected that fewer rules will need to be written for any future cases, given that the system already covers 80% of the domain.

4.3 Percentage of Classifications Missed

Figure 3 showns that the percentage of cases that received updated classifications reduced dramatically even after only a small number of cases, suggesting the system was rapidly helping to reduce the expert's rate of missed classifications by suggesting the classifications for them. The trend-line is approximate. However, a clearly linear downward progression would be expected without the system assisting the expert.

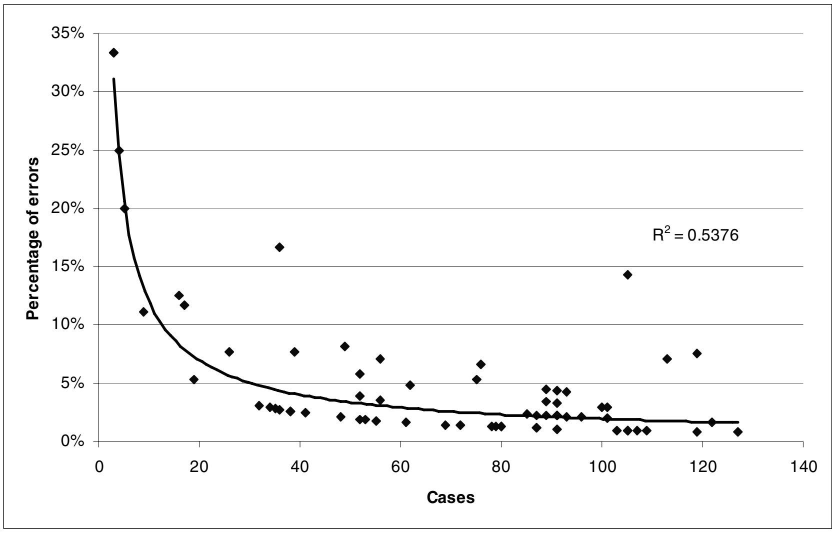


Fig. 3. The percentage of cases that gained new classifications

4.4 Total Errors Per Case

It was found that the rate of error in each case was quite high, averaging 13.5% and reaching over 50%. Shown in Figure 4 is the average errors made in each group of 10 cases considered. The local maxima around case 40 and 100 roughly match periods when the expert's data sources changed.

4.5 Maintainability and Usability

It has previously been demonstrated that the complex nature of the multidisciplinary domain of MR did not damage the maintainability and usability of the system [4, 18]. This result did not markedly change through the course of additional testing.

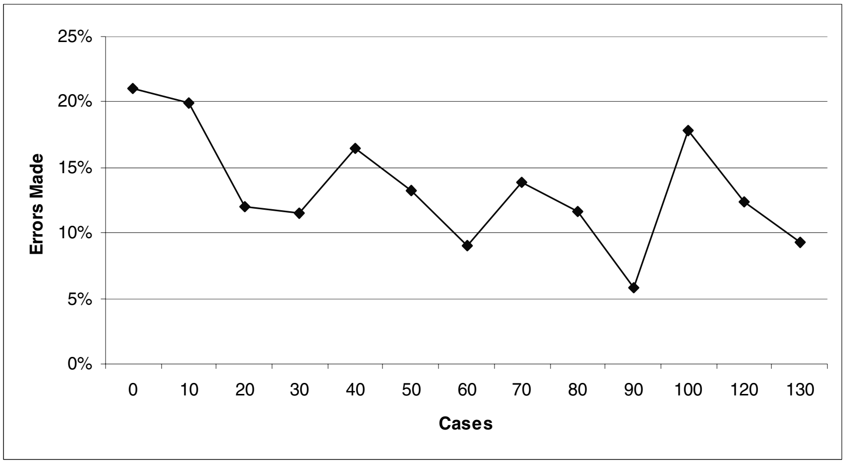


Fig. 4. The final percentage of classifications missed by expert per case

4.5.1 Structure of the Knowledge Base

It can be determined from Table 3 that the structure of the knowledge base tree was extremely shallow and branchy, suggesting the possibility of an excessive number of exceptions has not eventuated.

Table 3. Structure of the Knowledge Base Tree, 137 branches had a length of 2

Tree Property	Value
Average Length	1.79
Length 0	41
Length 1	86
Length 2	137
Length 3	25
Length 4	5
Length 5	5

4.6 Observations

The expert was queried regularly during the course of the experimentation, and extensively upon completion. Their comments revealed that although overall satisfied with the power of the system, they were dissatisfied primarily in two areas. Firstly, with the expressability of the domain model, as the expert found themselves unable to create certain desirable rules. For example, it was impossible for the expert to create a rule about a patient's pathology results which expressed the more abstract concept of "increasing" or "decreasing". Similarly, they were challenged with insufficiently detailed levels of grouping for the medications, with the expert finding that grouping on drug types, generic drugs, and commercial drugs too restrictive. Secondly, with fundamental limitations of the MCRDR method, since the expert was unable to create rules which used conclusions of the case as conditions in a new rule. This means the expert is unable to create rules which infer missing attributes of the case, or synonymous attributes, and then create a rule based on these inferred attributes. Further to this, it means the expert cannot necessarily represent the knowledge as efficiently or as faithfully to their own interpretation of it.

5 Conclusions

Initial experimentation suggested that the proposed method using MCRDR could successfully represent knowledge where the knowledge sources (human experts) are inconsistent. The system is shown to have reached about an 80% correct classification rate with less than 20 expert hours and only 126 cases classified – an excellent outcome in the circumstances. The knowledge base structure did not show any major deviations from what would be anticipated in a normal MCRDR system.

From a MR perspective the system was seen to be capable of: providing classifications for a wide range of Drug Related Problems; learning a large portion of the domain of MRs quickly; producing classifications in a timely manner; and importantly, vastly reducing the amount of missed classifications that would otherwise be expected of the reviewer. It is expected that a future incarnation of this system, would be capable of achieving classification rates over 90% [19].

However, it was identified that there are several shortcomings in the current incarnation of the system. Particularly it is known that the domain model is not sufficiently expressive.

Further to this, it was observed that limitations to the method restricted the expert in the creation of genuinely inferred knowledge from the case. This type of functionality is not available in the MCRDR method to date, and to accommodate it will require extensive addition to the method, which is talked about below.

6 Further Work

The current system stands as a satisfactory proof of concept and even in its infancy is considered by the expert to be considerably more powerful than any other medication management system to date. It is clear however that the issues of standardisation and representation of data is still a hurdle, and the only obvious solution is to continue prototyping until a sufficiently expressive domain model is settled upon.

The other issues, concerning the limitations of the method, are still being investigated. It is proposed that to add further conditions based on existing conditions or conclusions of the case it is necessary to add to the underlying tree-like exception based rule structure of the classic MCRDR knowledge base. Imagine a case where the expert wishes to create a rule that reads “IF ConcA & ConcB & Att1 THEN ConcE”. To remain true to the exception based nature of MCRDR this rule must be represented simply as “If Att1 THEN ConcE”, but should be considered only when ConcA and ConcB are already known to be true. To achieve this effect one might add a set of switches to the “If Att1 THEN ConcE” rule. Then, when a requisite conclusion fires it would also turn on its corresponding switch and evaluate that rule. In a situation where all switches on the rule had been activated, and the condition of the rule itself were satisfied, the rule would then fire. This method would result in a graph like structure which maintains every feature of an MCRDR tree, simply extending it with optional extra layers which are dependent on one or more of the above layers' conclusions. It would even be possible for lower layers to be dependent on higher layers. A possible resultant structure is represented diagrammatically in Figure 5 below. Further research is currently being undertaken to bring this concept into fruition.

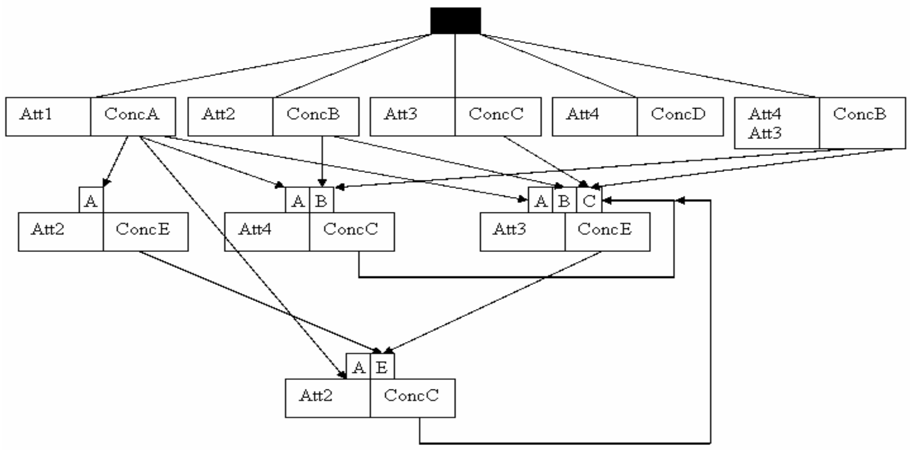


Fig. 5. A graph like structure such as that proposed

References

1. Peterson, G.: Continuing evidence of inappropriate medication usage in the elderly, in Australian Pharmacist 23, 2 (2004)
2. Bates, D., Cullen, D., Laird, N., Petersen, L., Small, S., Servi, D., Laffel, G., Sweitzer, B., Shea, B., Hallisey, R.: Incidence of adverse drug events and potential adverse drug events. Implications for prevention. ADE Prevention Study Group. JAMA , pp.29–34 (1995)
3. Peterson, G.: The future is now: the importance of medication review, in Australian Pharmacist, pp. 268–75 (2002)
4. Bindoff, I., Tenni, P., Kang, B., Peterson, G.:Intelligent Decision Support for Medication Review. In: Advances in Knowledge Acquisition and Management, Conference. Location (2006)
5. Bindoff, I., Tenni, P., Peterson, G., Kang, B., Jackson, S.: Development of an intelligent decision support system for medication review. J. Clin. Pharm. Ther. 32, 81–88 (2007)
6. Kinrade, W.: Review of Domiciliary Medication Management Review Software, Pharmacy Guild of Australia (2003)
7. Aamodt, A., Plaza, E.: Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches. AICom - Artificial Intelligence Communications 7, 39–59 (1994)
8. Compton, P., Kang, B., Preston, P., Mulholland, M.: Knowledge Acquisition without Analysis. In: Knowledge Acquisition for Knowledge-Based Systems, Conference. Location (1993)

9. Tenni, P., Peterson, G., Jackson, S., Hassan, O. to I. Bindoff (2005)
10. MediFlags, http://www.mediflags.com/
11. Compton, P., Jansen, R.: A philosophical basis for knowledge acquisition. In: European Knowledge Acquisition for Knowledge-Based Systems, Conference. Location (1989)
12. Rivest, R.: Learning Decision Lists. Machine Learning 2, 229–246 (1987)
13. Kang, B., Compton, P., Preston, P.: Multiple Classification Ripple Down Rules (1994)
14. Kang, B., Compton, P., Preston, P.: Multiple Classification Ripple Down Rules: Evaluation and Possibilities. In: AIII-Sponsored Banff Knowledge Acquisition for Knowledge-Based Systems, Conference. Location (1995)
15. Preston, P., Edwards, G., Compton, P.: A 2000 Rule Expert System Without a Knowledge Engineer. In: AIII-Sponsored Banff Knowledge Acquisition for Knowledge-Based Systems, Conference. Location (1994)
16. Compton, P., Jansen, R.: Cognitive aspects of knowledge acquisition. In: AAAI Spring Consortium, Conference. Location (1992)
17. Kang, B., Compton, P.: A Maintenance Approach to Case Based Reasoning (1994)
18. Bindoff, I.: An Intelligent Decision Support System for Medication Review, in Computing, vol. 65. University of Tasmania, Hobart (2005)
19. Tenni, P. to I. Bindoff (2005)

Continuity of Fuzzy Approximate Reasoning and Its Application to Optimization*

Takashi Mitsuishi[1] and Yasunari Shidama[2]

[1] University of Marketing and Distribution Sciences, Kobe 651-2188, Japan
takashi_mitsuishi@red.umds.ac.jp
[2] Shinshu University, Nagano 380-8553, Japan

Abstract. This paper describes a mathematical framework for studying a nonlinear feedback control. The fuzzy control discussed here is the nonlinear feedback control in which the feedback laws are determined by IF-THEN type fuzzy production rules through approximate reasoning introduced by Nakamori. To prove existence of optimal control, we applied compactness of a set of membership functions in L^2 space and continuity of the approximate reasoning, and prepared some propositions concerning approximate reasoning of Nakamori model. By considering fuzzy optimal control problems as problems of finding the minimum (maximum) value of the integral cost (benefit) function on an appropriate set of membership functions, the existence of fuzzy optimal control is shown.

Keywords: Fuzzy control, Nakamori model, Optimization, Functional analysis.

1 Introduction

In 1965 Zadeh introduced the notion of fuzziness [1], and then Mamdani has applied it to the field of control theory using what is called Mamdani method [2]. This method is one of the ways to represent numerically the control given by human language and sensitivity, and it has been applied in various practical control plants. However, unlike the theory of classical control and modern control, systematized considerations have not yet been discussed sufficiently. Moreover, in practical use, fuzzy membership functions, which represent input and output states in optimal control system, are decided on the basis of the experience of experts in each peculiar plant.

The authors have been studying to establish the automatic and computational determination of fuzzy membership functions, which give optimal controls in fuzzy control system. We also have been studying to find algorithms that compute optimal solutions. The optimization of fuzzy control discussed in this paper is different from conventional method such as classical control and modern control e.g. a linear matrix inequality approach [3]. We consider fuzzy

* The paper was supported in part by Grant-in-Aid for Young Scientists (B) #19700225 from Japan Society for the Promotion of Science (JSPS).

optimal control problems as problems of finding the minimum (maximum) value of the performance function with feedback law constructed by approximate reasoning with the way of functional analysis [4] [5] [6] [7] [8]. In our previous studies, some approximate reasoning: Mamdani method and T-S fuzzy model, etc., were analyzed about its two kind of continuity for the optimization. Using these approaches and results, one can obtain optimal fuzzy control using almost all approximate reasoning. However Nakamori method contains another calculation different from some major methods above, therefore we need to study its continuity and existence of optimal solution.

To guarantee the convergence of optimal solution the compactness of the set of membership functions in L^2 space is proved. And assuming approximate reasoning to be a functional on the set of membership functions, its continuity is proved. Then, we show that the system has an optimal feedback control by essential use of compactness of sets of fuzzy membership functions.

2 Fuzzy Control

2.1 Feedback System

In this paper we assume that the feedback part in this system is calculated by approximate reasoning presented by Nakamori. Using an idea and framework mentioned in the following section 3 and 4, the existence of optimal control based on fuzzy rules is designed.

$\mathbb{R}^n$ denotes the n-dimensional Euclidean space with the usual norm $\|\cdot\|$. Let $f(v_1, v_2) : \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}^n$ be a (nonlinear) vector valued function which is Lipschitz continuous. In addition, assume that there exists a constant $M_f > 0$ such that

$$\|f(v_1, v_2)\| \leq M_f \left(\|v_1\| + |v_2| + 1\right)$$

for all $(v_1, v_2) \in \mathbb{R}^n \times \mathbb{R}$. Consider a system given by the following state equation:

$$\dot{x}(t) = f(x(t), u(t)),$$

where $x(t)$ is the state and the control input $u(t)$ of the system is given by the state feedback

$$u(t) = \rho(x(t)).$$

For a sufficiently large $r > 0$, $B_r = \{x \in \mathbb{R}^n : \|x\| \leq r\}$ denotes a bounded set containing all possible initial states x_0 of the system. Let T be a sufficiently large final time. Then, we have

Proposition 1. *Let $\rho : \mathbb{R}^n \to \mathbb{R}$ be a Lipschitz continuous function and $x_0 \in B_r$. Then, the state equation*

$$\dot{x}(t) = f(x(t), \rho(x(t))) \tag{1}$$

has a unique solution $x(t, x_0, \rho)$ on $[0, T]$ with the initial condition $x(0) = x_0$ such that the mapping

$$(t, x_0) \in [0, T] \times B_r \mapsto x(t, x_0, \rho)$$

is continuous.

For any $r_2 > 0$, *denote by* Φ *the set of Lipschitz continuous functions* $\rho : \mathbb{R}^n \to \mathbb{R}$ *satisfying*

$$\sup_{u \in \mathbb{R}^n} |\rho(u)| \leq r_2.$$

Then, the following 1) *and* 2) *hold* [9].
1) *For any* $t \in [0,T], x_0 \in B_r$ *and* $\rho \in \Phi$, $\|x(t, x_0, \rho)\| \leq r_1$, *where*

$$r_1 = e^{M_f T} r + (e^{M_f T} - 1)(r_2 + 1). \tag{2}$$

2) *Let* $\rho_1, \rho_2 \in \Phi$. *Then, for any* $t \in [0,T]$ *and* $x_0 \in B_r$,

$$\|x(t, x_0, \rho_1) - x(t, x_0, \rho_2)\| \leq \frac{e^{L_f\,(1+L_{\rho_1})t} - 1}{1 + L_{\rho_1}} \sup_{u \in [-r_1, r_1]^n} |\rho_1(u) - \rho_2(u)|, \tag{3}$$

where L_f *and* L_{ρ_1} *are the Lipschitz constants of* f *and* ρ_1.

2.2 Nakamori Fuzzy Model

In this section we briefly explain the Nakamori fuzzy model [10] which decides feedback output in the previous nonlinear system for the convenience of the reader.

Assume the feedback law ρ consists of the following m IF-THEN type fuzzy control rules.

$$\begin{aligned}
&\text{RULE 1:} \quad IF\ x_1\ is\ A_{11}\ and\ \ldots\ and\ x_j\ is\ A_{1j}\ and\ \ldots\ and\ x_n\ is\ A_{1n}\\
&\qquad\qquad THEN\ y\ is\ B_1\\
&\qquad\qquad \vdots\\
&\text{RULE } i\text{:} \quad IF\ x_1\ is\ A_{i1}\ and\ \ldots\ and\ x_j\ is\ A_{ij}\ and\ \ldots\ and\ x_n\ is\ A_{in}\\
&\qquad\qquad THEN\ y\ is\ B_i\\
&\qquad\qquad \vdots\\
&\text{RULE } m\text{:} \quad IF\ x_1\ is\ A_{m1}\ and\ \ldots\ and\ x_j\ is\ A_{mj}\ and\ \ldots\ and\ x_n\ is\ A_{mn}\\
&\qquad\qquad THEN\ y\ is\ B_m
\end{aligned} \tag{4}$$

Here, m is the number of fuzzy production rules, and n is the number of premise variables $x_1, x_2, \ldots, x_n$. Let $\mu_{A_{ij}}$ $(i = 1, 2, \ldots, m;\ j = 1, 2, \ldots, n)$ be membership functions defined on a certain closed interval of the fuzzy set A_{ij}. Let μ_{B_i} $(i = 1, 2, \ldots, m)$ be membership functions defined on a certain closed interval of the fuzzy set B_i. For simplicity, we write "IF" and "THEN" parts in the rules by the following notation:

$$\mathcal{A}_i = (\mu_{A_{i1}}, \ldots, \mu_{A_{in}}) \quad (i = 1, \ldots, m),$$

$$\mathcal{A} = (\mathcal{A}_1, \ldots, \mathcal{A}_m), \quad \mathcal{B} = (\mu_{B_1}, \ldots, \mu_{B_m}).$$

Then, the IF-THEN type fuzzy control rules (4) is called a fuzzy controller, and is denoted by $(\mathcal{A}, \mathcal{B})$. In the rules, the tuple of premise variable $x = x_1, x_2, \ldots, x_n$

is called an input information given to the fuzzy controller $(\mathcal{A}, \mathcal{B})$, and y is called an control variable.

Mamdani method [2] is widely used in fuzzy controls because of its simplicity and comprehensibility. Nakamori fuzzy model does not infer each rule, and defines the inference result of all rules as a weighted average of agreement degree as follows. When input information x is given to the fuzzy controller $(\mathcal{A}, \mathcal{B})$, then one can obtain the amount of operation $\rho_{\mathcal{AB}}(x)$ from the controller through the following calculation [10]:

Step 1: The degree of each rule i is calculated by

$$\alpha_{\mathcal{A}_i}(x) = \prod_{j=1}^{n} \mu_{A_{ij}}(x_j) \quad (i = 1, \dots, m).$$

Step 2: The inference result of the entire rules is calculated by

$$\gamma_{\mathcal{AB}}(x, y) = \frac{\sum_{i=1}^{m} \alpha_{\mathcal{A}_i}(x)\mu_{B_i}(y)}{\sum_{i=1}^{m} \alpha_{\mathcal{A}_i}(x)}.$$

Step 3: Defuzzification stage. The center of gravity of the inference result is calculated by

$$\rho_{\mathcal{AB}}(x) = \frac{\int y\gamma_{\mathcal{AB}}(x, y)dy}{\int \gamma_{\mathcal{AB}}(x, y)dy}.$$

In this study, when $x(t) = (x_1(t), x_2(t), \dots, x_n(t)) = x = (x_1, x_2, \dots, x_n)$ (for simplicity) is given, the state feedback $u(t) = \rho(x(t))$ consists of this approximate reasoning.

3 Compactness of a Set of Membership Functions

In this section, we introduce two sets of fuzzy membership functions and study their topological properties. Then we can show that a set of admissible fuzzy controllers is compact and metrizable with respect to an appropriate topology on fuzzy membership functions.

In the following fix $r > 0$, a sufficiently large $r_2 > 0$ and a final time T of the control (1). Put r_1 be the positive constant determined by (2). We also fix $\Delta_{ij} > 0$ $(i = 1, 2, \dots, m;\ j = 1, 2, \dots, n)$ and $\delta > 0$. Let $C[-r_1, r_1]$ be the Banach space of all continuous real functions on $[-r_1, r_1]$. Denote by $L^2[-r_2, r_2]$ the Hilbert space of all square integrable, Legesgue measurable real function on $[-r_2, r_2]$. We consider the following two sets of fuzzy membership functions.

$$F_{\Delta_{ij}} = \{\mu \in C[-r_1, r_1] : 0 \le \mu(x) \le 1 \text{ for } \forall x \in [-r_1, r_1],$$
$$|\mu(x) - \mu(x')| \le \Delta_{ij}|x - x'| \text{ for } \forall x, x' \in [-r_1, r_1]\}$$

and

$$G = \left\{\mu \in L^2[-r_2, r_2] : 0 \le \mu(y) \le 1 \text{ a.e. } y \in [-r_2, r_2]\right\}.$$

The set $F_{\Delta_{ij}}$, which is more restrictive than G, contains triangular, trapezoidal and bell-shaped fuzzy membership functions with gradients less than positive value Δ_{ij}. Consequently, if $\Delta_{ij} > 0$ is taken large enough, $F_{\Delta_{ij}}$ contains almost all fuzzy membership functions which are used in practical applications. We assume that the fuzzy membership function $\mu_{A_{ij}}$ in "IF" parts of the rules (4) belongs to the set $F_{\Delta_{ij}}$. On the other hand, we also assume that the fuzzy membership function μ_{B_i} in "THEN" parts of the rules (4) belongs to the set G. In the following, we endow the space $F_{\Delta_{ij}}$ with the norm topology on $C[-r_1, r_1]$ and endow the space G with the weak topology on $L^2[-r_2, r_2]$. Then for all $i = 1, 2, \ldots, m$ and $j = 1, 2, \ldots, n$, $F_{\Delta_{ij}}$ is a compact subset of $C[-r_1, r_1]$. And G is compact metrizable for the weak topology on $L^2[-r_2, r_2]$ [11] [12]. Put

$$\mathcal{F} = \prod_{i=1}^{m} \left\{ \prod_{j=1}^{n} F_{\Delta_{ij}} \right\} \times G^m,$$

where G^m denotes the m times Cartesian product of G. Then, every element $(\mathcal{A}, \mathcal{B})$ of $\mathcal{F}$ is a fuzzy controller given by the IF-THEN type fuzzy control rules (4). By the Tychonoff theorem we have

Proposition 2. *$\mathcal{F}$ is compact and metrizable with respect to the product topology on $C[-r_1, r_1]^{mn} \times G^m$ [9].*

To avoid making the denominator of the fractional expressions γ (step 2) and ρ (step 3) of the Nakamori approximate reasoning in the previous section equal to 0, for any $\delta > 0$ and $\sigma > 0$, we consider the set

$$\mathcal{F}_{\delta\sigma} = \left\{ (\mathcal{A}, \mathcal{B}) \in \mathcal{F} : \int_{-r_2}^{r_2} \gamma_{\mathcal{AB}}(x, y) dy \geq \delta, \right.$$
$$\left. \sum_{i=1}^{m} \alpha_{\mathcal{A}_i}(x) \geq \sigma \text{ for all } x \in [-r_1, r_1]^n \right\}, \tag{5}$$

which is a slight modification of $\mathcal{F}$. If δ and σ is taken small enough, it is possible to consider $\mathcal{F} = \mathcal{F}_{\delta\sigma}$ for practical applications. We say that an element $(\mathcal{A}, \mathcal{B})$ of $\mathcal{F}_{\delta\sigma}$ is an admissible fuzzy controller. Then, we have the following

Proposition 3. *The set $\mathcal{F}_{\delta\sigma}$ of all admissible fuzzy controllers is compact and metrizable with respect to the product topology on $C[-r_1, r_1]^{mn} \times G^m$.*

Proof. We first note that a sequence $\{(\mathcal{A}^k, \mathcal{B}^k)\} \subset \mathcal{F}$ converges to $(\mathcal{A}, \mathcal{B})$ in $\mathcal{F}$ for the product topology if and only if, for each $(i = 1, 2, \ldots, m)$,

$$\|\alpha_{\mathcal{A}_i^k} - \alpha_{\mathcal{A}_i}\|_\infty \equiv \sup_{x \in [-r_1, r_1]^n} |\alpha_{\mathcal{A}_i^k}(x) - \alpha_{\mathcal{A}_i}(x)| \to 0$$

and

$$\mu_{B_i^k} \to \mu_{B_i} \text{ weakly on } L^2[-r_2, r_2].$$

Assume that a sequence $\{(\mathcal{A}^k, \mathcal{B}^k)\}$ in $\mathcal{F}_{\delta\sigma}$ converges to $(\mathcal{A}, \mathcal{B}) \in \mathcal{F}$. Fix $x \in [-r_1, r_1]^n$. Then, it is easy to show that

$$\int_{-r_2}^{r_2} \gamma_{\mathcal{AB}}(x,y)dy = \lim_{k\to\infty} \int_{-r_2}^{r_2} \gamma_{\mathcal{A}^k\mathcal{B}^k}(x,y)dy \geq \delta$$

and

$$\sum_{i=1}^{m} \alpha_{\mathcal{A}_i}(x) = \lim_{k\to 0} \sum_{i=1}^{m} \alpha_{\mathcal{A}_i^k}(x) \geq \sigma.$$

And this implies $(\mathcal{A}, \mathcal{B}) \in \mathcal{F}_{\delta\sigma}$. Therefore, $\mathcal{F}_{\delta\sigma}$ is a closed subset of $\mathcal{F}$, and hence it is compact metrizable. □

4 Lipschitz Continuity of Nakamori Method

In this paper, for any tuple $(\mathcal{A}, \mathcal{B}) \in \mathcal{F}_{\delta\sigma}$, we define the feedback function $\rho_{\mathcal{AB}}$ in section 2 on the basis of the rules by Nakamori approximate reasoning method.

$$\rho_{\mathcal{AB}}(x) = \frac{\int_{-r_2}^{r_2} y\gamma_{\mathcal{AB}}(x,y)dy}{\int_{-r_2}^{r_2} \gamma_{\mathcal{AB}}(x,y)dy},$$

where

$$\alpha_{\mathcal{A}_i}(x) = \prod_{j=1}^{n} \mu_{A_{ij}}(x_j) \quad (i = 1, \ldots, m)$$

and

$$\gamma_{\mathcal{AB}}(x,y) = \frac{\sum_{i=1}^{m} \alpha_{\mathcal{A}_i}(x)\mu_{B_i}(y)}{\sum_{i=1}^{m} \alpha_{\mathcal{A}_i}(x)}.$$

Proposition 4. *Let $(\mathcal{A}, \mathcal{B}) \in \mathcal{F}_{\delta\sigma}$. Then, the following* 1) *and* 2) *hold.*
1) *$\rho_{\mathcal{AB}}$ is Lipschitz continuous on $[-r_1, r_1]^n$.*
2) *$|\rho_{\mathcal{AB}}(x)| \leq r_2$ for all $x \in [-r_1, r_1]^n$.*

Proof. 1) For any $x, x' \in [-r_1, r_1]^n$ and any $i = 1, 2, \ldots, m$, we have

$$|\alpha_{\mathcal{A}_i}(x) - \alpha_{\mathcal{A}_i}(x')| \leq \sqrt{n} L_i \|x - x'\|, \tag{6}$$

where $L_i = \max_{j=1,2,\ldots,n} \Delta_{ij}$. Noting that $|\mu_{B_i}(y)| \leq 1$, $\sum_{i=1}^{m} \alpha_{A_i}(x)\mu_{B_i}(y) \leq m$ and $\sigma \leq \sum_{i=1}^{m} \alpha_{A_i}(x) \leq m$ for all $y \in [-r_2, r_2]$, it follows from (6) that

$$|\gamma_{\mathcal{AB}}(x,y) - \gamma_{\mathcal{AB}}(x',y)|$$

$$\leq \frac{m}{\sigma^2} \left(\sum_{i=1}^{m} |\alpha_{A_i}(x) - \alpha_{A_i}(x')| + \sum_{i=1}^{m} \mu_{B_i}(y)|\alpha_{A_i}(x) - \alpha_{A_i}(x')| \right)$$

$$\leq \frac{2m\sqrt{n}}{\sigma^2} \sum_{i=1}^{m} L_i \|x - x'\|.$$

Hence the mapping $\gamma_{\mathcal{AB}}$ is Lipschitz continuous on $[-r_1, r_1]^n$.

Put

$$g(x) = \int_{-r_2}^{r_2} y\gamma_{\mathcal{AB}}(x,y)dy \text{ and } h(x) = \int_{-r_2}^{r_2} \gamma_{\mathcal{AB}}(x,y)dy,$$

then for any $x, x' \in [-r_1, r_1]^n$, we have

$$|g(x) - g(x')| \le {r_2}^2 L_\gamma \|x - x'\|, \quad |g(x)| \le \frac{m{r_2}^2}{\gamma},$$

$$|h(x) - h(x')| \le 2r_2 L_\gamma \|x - x'\| \text{ and } h(x) \le \frac{2mr_2}{\gamma}$$

where L_γ is Lipschitz constant of $\gamma_{\mathcal{AB}}$. Since $h(x) \ge \delta$ for all $x, x' \in [-r_1, r_1]^n$, it follows from previous expressions that

$$|\rho_{\mathcal{AB}}(x) - \rho_{\mathcal{AB}}(x')| \le \frac{|h(x')||g(x) - g(x')| + |g(x')||h(x) - h(x')|}{h(x)h(x')}$$

$$\le \frac{4m{r_2}^3 L_\gamma}{\sigma\delta^2}\|x - x'\|$$

and the Lipschitz continuity of $\rho_{\mathcal{AB}}$ on $[-r_1, r_1]^n$ is proved.
2) For any $y \in [-r_2, r_2]$, we have

$$y\gamma_{\mathcal{AB}}(x,y) \le r_2\gamma_{\mathcal{AB}}(x,y).$$

Noting that $h(x)$ is non zero and

$$\int_{-r_2}^{r_2} y\gamma_{\mathcal{AB}}(x,y)dy \le r_2 \int_{-r_2}^{r_2} \gamma_{\mathcal{AB}}(x,y)dy.$$

Then

$$|\rho_{\mathcal{AB}}(x)| = \left| \frac{\int_{-r_2}^{r_2} y\gamma_{\mathcal{AB}}(x,y)dy}{\int_{-r_2}^{r_2} \gamma_{\mathcal{AB}}(x,y)dy} \right| \le r_2$$

for all $x \in [-r_1, r_1]^n$. □

Let $(\mathcal{A}, \mathcal{B})$ be a fuzzy controller given by the IF-THEN type fuzzy control rules (4). We say that the system (1) is a fuzzy feedback system if the control function $u(t)$ is given by the state feedback $u(t) = \rho_{\mathcal{AB}}(x(t))$, where $\rho_{\mathcal{AB}}(x(t))$ is the amount of operation from the fuzzy controller $(\mathcal{A}, \mathcal{B})$ for an input information $x(t)$.

It is easily seen that every bounded Lipschitz function $\rho : [-r_1, r_1]^n \to \mathbb{R}$ can be extended to a bounded Lipschitz function $\tilde{\rho}$ on $\mathbb{R}^n$ without increasing its Lipschitz constant and bound. In fact, define $\tilde{\rho} : \mathbb{R}^n \to \mathbb{R}$ by

$$\tilde{\rho}(x) = \tilde{\rho}(x_1, \ldots, x_n)$$

$$= \begin{cases} \rho(x_1, \ldots, x_n), & \text{if } x \in [-r_1, r_1]^n \\ \rho(\varepsilon(x_1)r_1, \ldots, \varepsilon(x_n)r_1), & \text{if } x \notin [-r_1, r_1]^n, \end{cases}$$

where

$$\varepsilon(u) = \begin{cases} 1, & \text{if } u > r_1 \\ -1, & \text{if } u < -r_1. \end{cases}$$

Let $(\mathcal{A}, \mathcal{B}) \in \mathcal{F}_\delta$. Then it follows from proposition 4 and the fact above that the extension $\tilde{\rho}_{\mathcal{AB}}$ of $\rho_{\mathcal{AB}}$ is Lipschitz continuous on $\mathbb{R}^n$ with the same Lipschitz constant of $\rho_{\mathcal{AB}}$ and satisfies $\sup_{u \in \mathbb{R}^n} |\tilde{\rho}_{\mathcal{AB}}(u)| \le r_2$. Therefore, by proposition 1 the state equation (1) for the feedback law $\tilde{\rho}_{\mathcal{AB}}$ has a unique solution $x(t, x_0, \tilde{\rho}_{\mathcal{AB}})$ with the initial condition $x(0) = x_0$ [13]. Though the extension $\tilde{\rho}_{\mathcal{AB}}$ of $\rho_{\mathcal{AB}}$ is not unique in general, the solution $x(t, x_0, \tilde{\rho}_{\mathcal{AB}})$ is uniquely determined by $\rho_{\mathcal{AB}}$ using the inequality (3) of 2) of proposition 1. Consequently, in the following the extension $\tilde{\rho}_{\mathcal{AB}}$ is written as $\rho_{\mathcal{AB}}$ without confusion.

5 Minimize the Performance Function

The performance index of this fuzzy feedback control system is evaluated with the following integral performance function:

$$J = \int_{B_r} \int_0^T w(x(t, \zeta, \rho_{\mathcal{AB}}), \rho_{\mathcal{AB}}(x(t, \zeta, \rho_{\mathcal{AB}}))) dt d\zeta, \tag{7}$$

where $w : \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}$ is a positive continuous function. The following theorem guarantees the existence of a fuzzy controller $(\mathcal{A}, \mathcal{B})$ (IF-THEN type fuzzy control rules) which minimizes and maximize the previous performance function (7).

Theorem 1. *The mapping*

$$(\mathcal{A}, \mathcal{B}) \mapsto \int_{B_r} \int_0^T w(x(t, \zeta, \rho_{\mathcal{AB}}), \rho_{\mathcal{AB}}(x(t, \zeta, \rho_{\mathcal{AB}}))) dt d\zeta$$

has a minimum (maximum) value on the compact metric space $\mathcal{F}_{\delta\sigma}$ defined by (5).

Proof. Since compactness of $\mathcal{F}_{\delta\sigma}$ is already derived by proposition 3, it is sufficient to prove that the performance function is continuous on $\mathcal{F}_{\delta\sigma}$. Routine calculation gives the estimate

$$\begin{aligned} &\sup_{x \in [-r_1, r_1]^n} |\rho_{\mathcal{A}^k \mathcal{B}^k}(x) - \rho_{\mathcal{AB}}(x)| \\ &\le \frac{1}{\delta^2 \sigma} \Bigg\{ \sum_{i=1}^m \left| \int_{-r_2}^{r_2} \mu_{B_i^k}(y) dy - \int_{-r_2}^{r_2} \mu_{B_i}(y) dy \right| \\ &\quad + \sum_{i=1}^m \left| \int_{-r_2}^{r_2} y \mu_{B_i^k}(y) dy - \int_{-r_2}^{r_2} y \mu_{B_i}(y) dy \right| \Bigg\} \\ &\quad + \frac{r_2(2 + r_2)}{\delta^2 \sigma} \left(1 + \frac{m}{\sigma}\right) \sum_{i=1}^m \|\alpha_{\mathcal{A}_i^k} - \alpha_{\mathcal{A}_i}\|_\infty \end{aligned}$$

$$\leq \frac{\sqrt{2r_2}}{\delta^2\sigma}\left(1+\frac{r_2}{\sqrt{3}}\right)\sum_{i=1}^{m}\left(\int_{-r_2}^{r_2}|\mu_{B_i^k}(y)-\mu_{B_i}(y)|^2dy\right)^{\frac{1}{2}}$$
$$+\frac{r_2(2+r_2)}{\delta^2\sigma}\left(1+\frac{m}{\sigma}\right)\sum_{i=1}^{m}\|\alpha_{\mathcal{A}_i^k}-\alpha_{\mathcal{A}_i}\|_\infty.$$

Assume that $(\mathcal{A}^k,\mathcal{B}^k)\to(\mathcal{A},\mathcal{B})$ in $\mathcal{F}_{\delta\sigma}$ and fix $(t,\zeta)\in[0,T]\times B_r$. Then it follows from the estimate above that

$$\lim_{k\to\infty}\sup_{x\in[-r_1,r_1]^n}|\rho_{\mathcal{A}^k\mathcal{B}^k}(x)-\rho_{\mathcal{A}\mathcal{B}}(x)|=0. \tag{8}$$

Hence, by 2) of proposition 1, we have

$$\lim_{k\to\infty}\|x(t,\zeta,\rho_{\mathcal{A}^k\mathcal{B}^k})-x(t,\zeta,\rho_{\mathcal{A}\mathcal{B}})\|=0. \tag{9}$$

Further, it follows from (8), (9) and 1) of proposition 1 that

$$\lim_{k\to\infty}\rho_{\mathcal{A}^k\mathcal{B}^k}(x(t,\zeta,\rho_{\mathcal{A}^k\mathcal{B}^k}))=\rho_{\mathcal{A}\mathcal{B}}(x(t,\zeta,\rho_{\mathcal{A}\mathcal{B}})). \tag{10}$$

Noting that $w:\mathbb{R}^n\times\mathbb{R}\to\mathbb{R}$ is positive and continuous, it follows from (9), (10) and the Lebesgue's dominated convergence theorem [14] that the mapping is continuous on the compact metric space $\mathcal{F}_{\delta\sigma}$. Thus it has a minimum (maximum) value on $\mathcal{F}_{\delta\sigma}$, and the proof is complete. $\square$

6 Conclusion

This paper presents the analysis of the Nakamori fuzzy model. Lipschitz continuity on the premise variables and the continuity on the set of membership functions of its approximate reasoning are proved. Lipschitz continuity is one of the sufficient conditions for the existence of the unique solution of the state equation in the nonlinear feedback control. On the other, the continuity on the set of membership functions guarantees the existence of a rule set, in other words a tuple of membership functions, which minimize the integral cost function with compactness of the set of membership functions. The compact membership function set $\mathcal{F}_{\delta\sigma}$ in this paper includes the class of triangular, trapezoidal, and bell-shaped functions which are used in application generally. It is recognized that in various applications it could be a useful tool in analyzing the convergence of fuzzy control rules modified recursively.

References

1. Zadeh, L.A.: Fuzzy algorithms. Information and Control 12, 94–102 (1968)
2. Mamdani, E.H.: Application of fuzzy algorithms for control of simple dynamic plant. Proc. IEE 121 12, 1585–1588 (1974)

3. Tanaka, K., Taniguchi, T., Wang, H.O.: Robust and optimal fuzzy control: A linear matrix inequality approach. In: 1999 International Federation of Automatic Control World Congress, pp. 213–218 (1999)
4. Shidama, Y., Yang, Y., Eguchi, M., Yamaura, H.: The compactness of a set of membership functions and its application to fuzzy optimal control. The Japan Society for Industrial and Applied Mathematics 1, 1–13 (1996)
5. Mitsuishi, T., Kawabe, J., Shidama, Y.: Continuity of Fuzzy Controller. Mechanized Mathematics and Its Applications 1(1), 31–38 (2000)
6. Mitsuishi, T., Shidama, Y.: Minimization of Quadratic Performance Function in T-S Fuzzy Model. In: Proc. International Conference on Fuzzy Systems (FUZZ-IEEE2002), pp. 75–79 (2002)
7. Mitsuishi, T., Bancerek, G.: Lattice of Fuzzy Sets. Journal of Formalized Mathematics 11(4), 393–398 (2003)
8. Mitsuishi, T., Shidama, Y.: Continuity of Product-Sum-Gravity Method on L^2 Space Using Fuzzy Number for Premise Variable. In: Proc. 2007 9th International Symposium on Signal Processing and its Applications (2007) CD-ROM
9. Mitsuishi, T., Kawabe, J., Wasaki, K., Shidama, Y.: Optimization of Fuzzy Feedback Control Determined by Product-Sum-Gravity Method. Journal of Nonlinear and Convex Analysis 1(2), 201–211 (2000)
10. Nakamori, Y., Ryoke, M.: Identification of fuzzy prediction models through hyperellipsoidal clustering. IEEE Transactions on Systems, Man and Cybernetics SMC-24(8), 1153–1173 (1994)
11. Mitsuishi, T., Wasaki, K., Kawabe, J., Kawamoto, N.P., Shidama, Y.: Fuzzy optimal control in L^2 space. In: Proc. 7th IFAC Symposium Artificial Intelligence in Real-Time Control, pp. 173–177 (1998)
12. Mitsuishi, T., Kawabe, J., Wasaki, K., Kawamoto, N.P., Shidama, Y.: Membership functions in L^2 Space and its Applications to Fuzzy Optimal Control. In: Proc. IEEE International Conference on Systems, Man and Cybernetics, vol. III, pp. 51–55 (1999)
13. Miller, R.K., Michel, A.N.: Ordinary Differential Equations. Academic Press, New York (1982)
14. Dunford, N., Schwartz, J.T.: Linear Operators Part I: General Theory. John Wiley & Sons, New York (1988)

Tomographic Reconstruction of Images from Noisy Projections - A Preliminary Study

A.P. Dalgleish[1], D.L. Dowe[1], and I.D. Svalbe[2]

[1] Clayton School of IT, Monash University, Australia
[2] School of Physics, Monash University, Australia

Abstract. Although Computed Tomography (CT) is a mature discipline, the development of techniques that will further reduce radiation dose are still essential. This paper makes steps towards projection andreconstruction methods which aim to assist in the reduction of this dosage, by studying the way noise propagates from projection space to image space. Inference methods Maximum Likelihood Estimation (MLE), Akaike's Information Criterion (AIC) and Minimum Message Length (MML) are used to obtain accurate models obtained from minimal data.

1 Introduction

The optimal reconstruction of images from traditional Computer Tomographic (CT) projections, in the presence of noise, is a real-world problem. High-resolution CT scans deliver a large amount of radiation to the patient. CT scans are used frequently by physicians to diagnose different forms of cancer and many other conditions. The number of CT scans performed has increased, from 3 million in 1980, to 62 million in 2006 [1]. CT scans now account for half the radiation dose that an American receives per year.

We are interested in understanding how noise propagates through projections, to the reconstructed images, through a known reconstruction method, the Finite Radon Transform. It is hoped that through this understanding, we can take steps towards obtaining lower dose CT scans with the same resolution.

Minimum Message Length (MML), Maximum Likelihood Estimation (MLE) and Akaike's Information Criterion (AIC) are used in this context to model the noise in the CT reconstruction data, as well as modelling the underlying data. It is hoped that the use of these modelling techniques will allow a means to reconstruct images from minimal data.

Current CT methods of reconstructing images from projection data include Filtered Back Projection and Fourier Inversion. These methods use approximations in the methods they use for the reconstruction of projection data, the Fourier Inversion for example interpolates between projection values to obtain the reconstructed image. This masks how the noise propagates through the projections to the reconstructed image.

The reconstruction method used here is an exact reconstruction method, called the Finite Radon Transform (FRT). This method does not use any approximations or any other a priori methods of data manipulation. There are however two constraints that are imposed on this method of reconstruction; they are that the data is assumed to be in form of a discrete matrix, and that this matrix is mapped to a torus.

The Finite Radon Transform is used in this preliminary study because it has been shown [2] that projections from real X-ray CT data can be mapped into the FRT projection form via the use of the Mojette transform [3], as an intermediary stage. This approach bypasses the need for approximation or interpolation in the reconstruction method, as the FRT is exactly invertible. All of the required data processing can be done on one-dimensional projection data. Also, as the CT data that is acted upon is digital data, it can be assumed to be discretized.

Understanding the propagation of noise from FRT through Mojette to real CT projection data and the reconstructed image is an essential prelude to the practical application of the FRT for tomographic reconstruction.

Noise in real CT projections arises from many different sources, these include X-ray scatter, Quantum noise of the X-rays (X-ray photon statistics), the data acquisition system - the electronic noise produced by this system, can combine with the photon noise.

By studying the effect that noise has on the inversion of projection data, it is hoped that we can use this information to assist in the correction of noisy FRT projection data. This information could then be used to assist in the reconstruction of projection data from other reconstruction methods.

The FRT has not been used in this context before, and that makes this research new and novel. Whilst the FRT is not comparable with any of the current methods of CT projection and reconstruction, it is possible to convert the projection data from conventional CT projections to another projection-reconstruction method, the Mojette method. This method is much closer to conventional CT methods than the FRT method. Once the data is in the form of the Mojette projection-reconstruction method, this data can be converted to the Finite Radon Transform.

The beauty of FRT is the fact that it has a fixed pre-defined set of projections, so there is no need for us to choose projection angles. For a noise free set of projection data of a PxP matrix, when P is a prime number, then the inverse also has a well defined algorithm, which makes the inverse simple for noise free data [4]. The exactness of FRT reconstruction method is the property that we have exploited for this research, in conjunction with the data estimation techniques Maximum Likelihood Estimation (MLE), Akaike's Information Criterion (AIC) and Minimum Message Length (MML).

2 Discrete Projection Techniques

The Finite Radon Transform was used for this research because it provides a means for exact reconstruction without the need for any approximations. The Mojette Projection and Reconstruction methods provide the link between the Finite Radon Transform, and conventional CT methods, since there is a known mapping between the Finite Radon Transform and the Mojette method, and there is also a known mapping between the Mojette method and conventional CT data.

2.1 Finite Radon Transform (FRT) Projections

The data used for the Finite Radon Transform (FRT) is in the form of a discrete, square matrix of size $P \times P$, called an 'Image Matrix', where P is a prime number. As stated before, the image matrix is also assumed to be mapped to a torus, allowing projections of the image data to wrap around the sides of the image matrix [5]. The next 'pixel' of the image array that is part of the same projection to be sampled is determined by moving m pixels across and 1 pixel down. Since the image matrix is a torus, the movements can wrap around the sides.

The definition of the Finite Radon Transform given here is quite concise. For an in-depth discussion of the Finite Radon Transform and the projection and reconstruction methods, see [4,5].

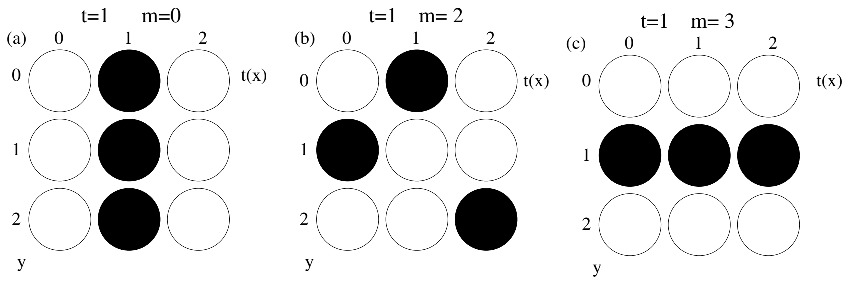


Fig. 1. FRT projections for a $P \times P = 3 \times 3$ image matrix (a) A projection where $t = 1$ and $m = 0$. (b) A projection where $t = 1$ and $m = 2$. (c) Projection $m = P$ and $t = 1$ sums the pixel values along row 1.

2.2 Mojette Projections

The Mojette technique is an asymmetric projection technique, where the projection angles are specified by discrete rational angles $\tan^{-1}(q/p)$ and the beam spacing by $1/\sqrt{p^2 + q^2}$, where p and q are integers and $q > 0$. Thus the projection value becomes the sum of the pixels, whose centres lie on the line $t = qx - py$ [4]. This means that image matrix no longer has to be mapped to a torus. The Mojette projections are close in form to that of the real projection data acquired in a traditional CT scan.

For further information on the Mojette Transform see [4,6].

3 Modelling Techniques

Recall from the above sections that the FRT and Mojette projection and reconstruction methods are efficient and effective, projection and reconstruction methods. However they both suffer problems in the presence of noise. There are several corner based Mojette inversion methods [6], in which noise in the projection matrix is propagated in from the corners, and as the corners of the projection matrix tend to have larger amounts of noise, the propagation of noise can be substantial. In the Finite Radon Transform noise from projection data is carried from projection space to image space. The FRT is an exact reconstruction method that spreads a uniform amount of noise from projections into the image.

The poor performance of these methods in the presence of noise provides the need for a method of noise estimation, outlined below are three proposed methods for noise estimation, namely Maximum Likelihood Estimation (MLE), Akaike's Information Criterion (AIC) and Minimum Message Length (MML).

The research consisted of two main noise models that were added to the image matrix, the image matrix which was a fixed binary image. In the first of these models, Gaussian noise was added to the image matrix, with known $\mu = 0$, we varied the sigma values from $0 < \sigma < 1$. The value of μ was assumed to be known when using the modelling techniques, this model is known as the 'No Cut' model. The data modelling techniques were used to infer the value of σ.

For the other noise model, Gaussian noise was added to the image in two distinct regions, that is two different values for σ, namely σ_l and σ_r, once again we set a known $\mu_l = \mu_r = \mu = 0$. These regions were separated via a straight line, whose centre lay in the centre pixel, the line had associated with it an angle θ, allowing us to specify different regions of noise. This is referred to as the 'Cut' method, since the line 'cuts' the image into two noise regions. In this case, data modelling techniques were used to infer the value of θ as well as the value of the two σ values, from the two distinct noise regions. Figure 2 demonstrates an example 'cut' of a 7×7 image, it also shows the assumption that was used in our testing, this was that any pixel through which the line passes through the centre is counted as part of the σ_r side.

The data modelling methods that are outlined below all use the following method for encoding θ, $A = \sum_{i=1}^{N}(\phi(i))$, where A represents the total number of angles that have been examined for a particular matrix size P, $N = \frac{P-1}{2}$, and ϕ is the Euler's Totient. If we assume that the angles are ordered by size, then to refer to a certain angle, we can simply refer to the angle via an index i, where $0 < i < A$. Therefore we encode θ as $\log{(A)} + \log{(8)}$, where $\log{(8)}$ encodes the octant, which the line passes through. The value of A is fixed for a particular value of P, due to the fact that we only examine rational angles in the range $0 \leq \theta < 45$. The FRT and Mojette both use rational angles for constructing the projection space.

3.1 Maximum Likelihood Estimation (MLE)

Maximum Likelihood is based on the concept of conditional probability. The 'Likelihood' of parameter d given an event E is $P(d|E)$, in this case we use the observed data to estimate the parameters.

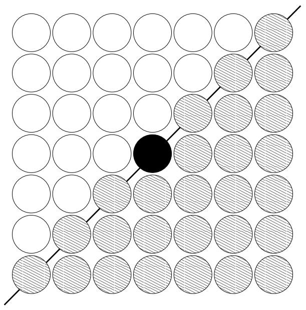


Fig. 2. An example cut, $\theta = 45°$, σ_l applies to unshaded pixels, σ_r applies to shaded pixels, μ is set to 0 and is known

A drawback of Maximum Likelihood is that it will always prefer a more complex model, thus in our tests it always chooses the model with the 'cut', as per (2), over the model with no 'cut', as per (1),

$$L(x|\mu,\sigma) = \tfrac{1}{2} \times P^2 \times \log(2\pi) + P^2 \times \log(\sigma) + \tfrac{1}{2\sigma^2} \times \textstyle\sum_{i=0}^{P^2-1}(x_i - \mu)^2 \qquad (1)$$

where P is the size of the image matrix, and x_i is a pixel in the matrix and

$$L(x|\mu,\sigma_l,\sigma_r) = n_l \times \log(\sqrt{2\pi}\sigma_l) + \tfrac{1}{2\sigma_l^2}\textstyle\sum_{i=0}^{n_l-1}(x_i-\mu)^2 + n_r \times \log(\sqrt{2\pi}\sigma_r) + \\ \tfrac{1}{2\sigma_r^2}\textstyle\sum_{i=n_l}^{P^2-1}(x_i-\mu)^2 + \log(A) + \log(8) \qquad (2)$$

where n_l and n_r refer to the number of pixels to the left and right of the line which separates the two noise regions, $n_l + n_r = P^2$ and $n_l \leq n_r$.

In (1) there is assumed to be one Normal noise distribution to which all pixels belong, and as the image matrix is square, and of size $P \times P$, there are P^2 pixels. For (2), the noise is assumed to come from two separate noise distributions that are partitioned by the 'cut'.

3.2 Akaike's Information Criterion (AIC)

Maximum Likelihood Estimation can be improved by using the 'Akaike Information Criterion' (AIC), this was proposed by Akaike in 1970, and is a method of performing *penalised maximum likelihood.* The addition of this penalty factor means that, although there may be a model with a higher [negative] log likelihood, this may not be chosen if its model is sufficiently complex [7].

Akaike's Information Criterion attempts to minimize the quantity

$$AIC = 2 \times L + 2K \qquad (3)$$

where L refers to the [negative] log-likelihood, such as either (1) or (2) above, and K refers to the number of free parameters.

3.3 Minimum Message Length (MML) Models

MML is a technique for modelling data, originally proposed by Wallace and Boulton [8], and based on Occam's Razor, 'the simplest explanation is the best'. This means that with MML if there are two messages that model the data equally well, the one with the shortest length will be chosen. If we consider our data as a string S, then a message (I) is defined as being two parts, $I = H \ : \ A$, where ':' is concatenation, H is a hypothesis about the source of the data string S, and A is an encoding of the data using a code the would be optimally efficient if the hypothesis H, were true [9].

For further discussion of the MML coding technique see [8,9,10], for a detailed comparison between AIC and MML, see [11]. The approximation to the message length that was explored in this research, is the Quadratic Approximation to Message Length method, for a full discussion of this method see [12,10].

The general form of this approximation is

$$MessageLength \approx -\log(\boldsymbol{\theta}) + L + \frac{1}{2}\log F + \frac{D}{2}(1 - \log(12)) \qquad (4)$$

where $\boldsymbol{\theta}$ is the prior, L is the [negative] log-likelihood, F is the determinant of the expected Fisher Information Matrix and D is the dimensionality.

Minimum Description Length (MDL) is a method that is very similar to MML, for more information on MDL see [13]. For a discussion of the relationship between MML and MDL see [14, sec. 11.4] and [10, sec. 10.2].

There were two possible models for MML to use.
These message length for the 'no cut' model (with $D = 1$) is given by

$$\begin{aligned} &MessageLength_{noCut} = \log(2R\sigma) + \tfrac{1}{2} \times P^2 \times \log(2\pi) + P^2 \times \log(\sigma) + \\ &\tfrac{1}{2\sigma^2} \times \textstyle\sum_{i=0}^{P^2-1}(x_i - \mu)^2 \times \tfrac{1}{2}\log\left(\tfrac{3}{\sigma^2} \times \tfrac{P^2}{\sigma^2}\right) + \tfrac{D}{2}(1 - \log(12)) \end{aligned} \qquad (5)$$

where P refers to the size of the image matrix, and $R = 6$.
In this message length the prior is $\frac{1}{2R\sigma}$ (for $e^{-R} \leq \sigma \leq e^{R}$) , L is given by (1) and the expected Fisher Information is $\left(\frac{3}{\sigma^2} \times \frac{P^2}{\sigma^2}\right)$.

The message length for the 'cut' model (with $D = 2$) is given by

$$\begin{aligned} &MessageLength_{cut} = \log(2R\sigma_l) + \log(2R\sigma_r) + \log(A) + \log(8) + n_l \times \\ &\log(\sqrt{2\pi}\sigma_l) + \tfrac{1}{2\sigma_l^2}\textstyle\sum_{i=0}^{n_l-1}(x_i - \mu)^2 + n_r \times \log(\sqrt{2\pi}\sigma_r) + \tfrac{1}{2\sigma_r^2}\sum_{n_l}^{P^2-1}(x_i - \mu)^2 + \\ &\tfrac{1}{2} \times \log\left(\tfrac{3}{\sigma_l^2} \times \tfrac{3}{\sigma_r^2} \times \left(\tfrac{n_l}{\sigma_l^2} + \tfrac{n_r}{\sigma_r^2}\right)\right) + \tfrac{D}{2}(1 - \log(12)) \end{aligned} \qquad (6)$$

where n_l and n_r refer to the number of pixels to the left and right of the line which separates the two noise regions, $n_l + n_r = P^2$ and $n_l \leq n_r$, P refers to the size of the matrix, and $R = 6$.
In this Message Length the prior is given by $\frac{1}{2R\sigma_l} \times \frac{1}{2R\sigma_r}$ (for $e^{-R} \leq \sigma_l \leq e^{R}$ and $e^{-R} \leq \sigma_r \leq e^{R}$), L is given by (2) and the expected Fisher Information is given by $\left(\frac{3}{\sigma_l^2} \times \frac{3}{\sigma_r^2} \times \left(\frac{n_l}{\sigma_l^2} + \frac{n_r}{\sigma_r^2}\right)\right)$.

4 Results

Testing involved varying values of σ for the 'No Cut' model, where $0 < \sigma < 1$, and varying σ_l, σ_r and θ, for the 'Cut' model, where $0 < \sigma_l < 1$, $0 < \sigma_r < 1$, and $0 \leq \theta < 45$. Tables 1 and 2 show sample results from our testing. All tests were run on an AMD Athlon X2 4400, with 2Gb RAM, with run-times in the order of 35 seconds for $P = 101$.

The Kullback-Leibler (KL) Distance [10, Ch. 4] [11, sec. 6.1.4] for two different Normal distributions, from $N(\mu_1, \sigma_1^2)$ to $N(\mu_2, \sigma_2^2)$ is given by (7).

$$KL = \log \frac{\sigma_2}{\sigma_1} + \frac{\sigma_1^2 - \sigma_2^2 + (\mu_1 - \mu_2)^2}{2\sigma_2^2} \tag{7}$$

Table 1. Results of an average over 100 runs per entry, with varied matrix (P) sizes, with one region of noise having value of $\sigma = 0.9$ and known $\mu = 0$, the true model in this case has no 'cut'. In Tables 1 and 2 bold entries denote minimum at each P.

Method	P	Mean KL $\times 10^3$	σ_{KL} $\times 10^4$
MLE	7	3.3775	9.0906
AIC	7	3.1871	8.0018
MML	7	**2.8444**	2.8233
MLE	13	3.0014	4.5181
AIC	13	3.4823	4.7807
MML	13	**2.7876**	1.3227
MLE	23	2.9298	3.0811
AIC	23	2.9376	3.5868
MML	23	**2.7816**	0.8471
MLE	29	2.9090	2.6427
AIC	29	2.8756	2.4625
MML	29	**2.7771**	0.5874
MLE	31	2.8610	2.5891
AIC	31	2.8588	2.5151
MML	31	**2.7746**	0.6432
MLE	101	2.8050	0.9532
AIC	101	2.7964	1.3412
MML	101	**2.7767**	0.1920

The results in Table 1 refer to experiments in which Gaussian noise was added to a predefined image, in this case the noise was assumed to be Gaussian noise spread across one region. Unlike the previous experiments, there is no 'cut' and no θ and there is only one σ value. These results highlight the predisposition of MLE and AIC to prefer the more complicated model. Given the chance of a model that has two distinct regions of noise associated with it (therefore two separate σ values and a θ value) and a model with only one region of noise (so only a single σ), the results show that AIC and MLE will choose the more complicated

Table 2. Results of an average over 100 runs per entry, with varied matrix (P) sizes, with two distinct regions of noise having values of $\sigma_l = 0.7$, $\sigma_r = 0.4$, $\theta = atan(0.5)$ and known $\mu = 0$. σ_{KL} refers to the standard deviation of the distribution of KL-Distances, averaged over 100 runs per entry, the true model in this case has a 'cut'.

Method	P	Mean KL $\times 10^3$	σ_{KL} $\times 10^4$
MLE	7	3.6339	4.9118
AIC	7	**3.6287**	4.2386
MML	7	22.9901	4.4659
MLE	13	3.4470	2.2414
AIC	13	3.5218	1.9877
MML	13	**3.4436**	2.1782
MLE	23	**3.3648**	1.2160
AIC	23	3.3754	1.0974
MML	23	3.3769	0.8456
MLE	29	**3.3706**	0.9451
AIC	29	3.3777	0.9969
MML	29	3.3769	0.8456
MLE	31	3.3658	0.9798
AIC	31	3.3770	0.8570
MML	31	**3.3584**	0.8657
MLE	101	3.3549	0.2566
AIC	101	3.3572	0.2738
MML	101	**3.3525**	0.6010

model more often than MML. The benefits of this choice are highlighted in the KL Distance values for the various data modelling techniques.

It should be noted that it is not inherently clear for AIC whether the number of free parameters refers to both discrete and continuous free parameters, or just continuous free parameters, testing was done on both, and the results of Tables 1 and 2 present the results when only continuous free parameters were considered, however the difference in KL values for the two separate cases were minimal.

The results in Table 2 refer to experiments in which Gaussian noise was added to a predefined image, however in this case there was assumed to be two regions of noise that may or may not have the same σ value. The noise was assumed to be from two separate normal distributions, with a known μ value common to the two normal distributions, and a separate unknown σ value for each of the distributions. The regions were partitioned by a line, that went through the center pixel of the image, and has associated with it an angle θ, hence the noise model has a 'cut'.

The noise in both the 'cut' and 'no cut' cases was added directly to the image itself and not to the projections. This was done as a prelude to the addition of noise to projection data. This circumvents the need to 'untangle' the projections before performing the noise estimation methods, but work will be done on this at a later stage.

Unlike MLE, both AIC and MML can be used as a means of model selection, that is, these noise estimation methods can be used to choose between two different noise models. In our tests, these methods were used to estimate the values of the σ or the two σ values and the θ value. MLE will always choose the model with two σ values and the θ value, as this will maximise the likelihood. AIC and MML will attempt to choose between this model and the model with only one σ value.

5 Conclusion

This research is a preliminary study into how noise propagates through projections from projection space to image space. The use of data modelling techniques should allow for the use of our understanding of noise propagation, to help us model noisy projection data. Minimum Message Length (MML) is well suited to this, since it works well when there are small amounts of data from which we are attempting to infer parameters.

At this point however the study has been limited to the study of noise estimation in image space, with Maximum Likelihood Estimation (MLE), Akaike's Information Criterion (AIC) and Minimum Message Length (MML) as methods for data modelling.

The Finite Radon Transform is used in this context since it provides a means of projecting and reconstructing image data exactly, as it does not use any approximations or interpolation, so it is clear how the noise is propagating. MLE, AIC and MML are being used in this context as it for the data modelling methods. For these initial tests they all have a similar performance, however these tests are concerned with data that is much simpler than would be found in real world CT data.

It is our hope that through continued study of this we will be able to construct a noise tolerant method of projection and reconstruction of Tomographic data, that will allow for CT scans to have the same anatomical detail with a reduction in the amount of radiation that the patient receives.

The MML work would be enhanced by more properly modelling the spatial correlation within images [15,16].

References

1. Rabin, R.C.: With Rise in Radiation Exposure, Experts Urge Caution on Tests. The New York Times (June 19, 2007)
2. Svalbe, I., van der Spek, D.: Reconstruction of Tomographic Images using Analog Projections and the Digital Radon Transform. Linear Algebra and its Applications 339(15), 125–145 (2001)
3. Normand, N., Guédon, J., Philippé, O., Barba, D.: Controlled Redundancy for Image Coding and High-speed Transmission. Proc. SPIE Visual Communications and Image Processing 2727, 1070–1081 (1996)
4. Kingston, A., Svalbe, I.D.: Projective Transforms on Periodic Discrete Image Arrays. Advances In Imaging and Electron Physics 139, 75–177 (2006)

5. Matus, F., Flusser, J.: Image Representations via a Finite Radon Transformation. IEEE Transactions: Pattern Analysis and Machine Intelligence 15(10), 996–1006 (1993)
6. Normand, N., Kingston, A., Éuenou, P.: A Geometry Driven Reconstruction Algorithm for the Mojette Transform. In: Discrete Geometry for Computer Imagery, 13th International Conference, pp. 122–133 (2006)
7. Akaike, H.: Statistical prediction information. Annals of the Institute of Statistical Mathematics 22, 203–217 (1970)
8. Wallace, C.S., Boulton, D.M.: An Information Measure for Classification. Computer Journal 11(2), 185–194 (1968)
9. Wallace, C.S., Dowe, D.L.: Minimum Message Length and Kolmogorov Complexity. Computer Journal 42(4), 270–283 (1999)
10. Wallace, C.S.: Statistical and Inductive Inference by Minimum Message Length. Springer, New York (2005)
11. Dowe, D.L., Gardner, S., Oppy, G.R.: Bayes not Bust! Why Simplicity is no problem for Bayesians. British Journal for the Philosophy of Science (December 2007) (forthcoming)
12. Wallace, C.S., Freeman, P.R.: Estimation and inference by compact coding. Journal of the Royal Statistical Society. Series B (Methodological) 3(49), 240–265 (1987)
13. Rissanen, J.: Modelling by the shortest data description. Automatica 14, 465–471 (1978)
14. Comley, J.W., Dowe, D.L.: Minimum Message Length and Generalized Bayesian Nets with Asymmetric Languages. Advances in Minimum Description Length: Theory and Applications. Ch. 11, pp. 265–294 (2005)
15. Wallace, C.S.: Intrinsic Classification of Spatially Correlated Data. Computer Journal 41(8), 602–611 (1998)
16. Visser, G., Dowe, D.L.: Minimum Message Length Clustering of Spatially-Correlated Data with Varying Inter-Class Penalties. In: 6th IEEE International Conference on Computer and Information Science (ICIS 2007), pp. 17–22 (July 2007)

Automated Intelligent Abundance Analysis of Scallop Survey Video Footage

Rob Fearn[1], Raymond Williams[1], Mike Cameron-Jones[1], Julian Harrington[2], and Jayson Semmens[2]

[1] School of Computing, University of Tasmania, Tasmania, Australia
[2] Marine Research Laboratories, Tasmanian Aquaculture and Fisheries Institute, Tasmania, Australia
{rcfearn, R.Williams, Michael.Cameron-Jones, Julian.Harrington, Jayson.Semmens}@utas.edu.au

Abstract. Underwater video is increasingly being pursued as a low impact alternative to traditional techniques (such as trawls and dredges) for determining abundance and size frequency of target species. Our research focuses on automatically annotating survey scallop video footage using artificial intelligence techniques. We use a multi-layered approach which implements an attention selection process followed by sub-image segmentation and classification. Initial attention selection is performed using the University of Southern California's (USCs) iLab Neuromorphic Visual Toolkit (iNVT). Once the iNVT has determined regions of potential interest we use image segmentation and feature extraction techniques to produce data suitable for analysis within the Weka machine learning workbench environment.

Keywords: Scallop Survey Video Transects, Automated Video Annotation.

1 Introduction

The Tasmanian Aquaculture and Fisheries Institute (TAFI) have been collecting underwater video of commercial and recreational scallop beds for over five years. The footage has been collected with the intention of developing minimally intrusive techniques for the quantitative assessment of scallop abundance. TAFI currently use dredging as their main methods of assessment of the commercial fishery but traditional techniques can have a destructive impact on the marine environment and its inhabitants [1][2]. Dredge surveys also require the use of a commercial scallop vessel, whereas video surveys can be conducted on one of TAFI's own research vessels. Recently, the Tasmanian scallop industry have begun collecting their own underwater video in order to find new beds and determine the health of known beds, as they can do this all year round, including periods when by law they are not allowed to have their dredge on board the vessel. The drawback of the video approach is that footage can be collected (particularly by Industry) much faster than it can be manually annotated and as a result TAFI's accumulating video footage has remained relatively untouched for a number of years.

Our research implements a multilayered approach to automatically annotating the scallop bed video footage captured by TAFI using Artificial Intelligence (AI) techniques. At this stage of our research the task being tackled is the identification and counting of commercial scallops from video footage of commercial scallop beds. We discuss the steps taken in video frame selection, analysis of conspicuous regions, segmentation, feature extraction and classification using the Weka Machine Learning Toolkit [3].

2 Video Footage Characteristics

The video footage used in this study varies in many aspects including environment, background colour and camera movement. TAFI's current underwater video system relies on a video camera being tethered to a vessel a "drop camera" and is therefore affected by the speed the vessel is travelling at and the overall smoothness of the surface. The sea is seldom perfectly calm or flat, so the footage is subject to a constant up-and-down and rolling motion that is not perfectly uniform making it difficult to compensate for its effects.

TAFI have recently purchased a Remotely Operated Vehicle (ROV) and as a result it will be possible in the future to avoid the current undulation problem. However, for archived footage and footage collected by Industry vessels it is necessary to select sections of footage where the camera is deemed to be at an appropriate distance from the sea bed as footage from the ROV should be. When the camera is too high the scallops become ill-defined due to water clarity. When the camera is too low, individual scallops take over the entire frame. It is estimated that the preferred distance from the sea bed is approximately one third from the top of the camera's range of vertical movement in the current camera footage.

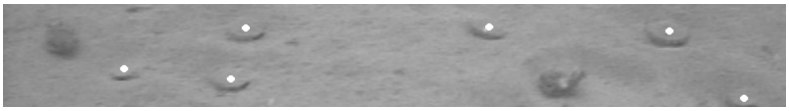


Fig. 1. A cropped sample of commercial scallop bed footage provided by TAFI. The location of commercial scallops within the image is indicated by the white dots.

Our primary focus in this study is on commercial scallop beds. Fortunately this footage involves comparatively sandy sea beds as opposed to some recreational scallop beds, for which some footage is also available, but there is less of a management need to examine this footage. Unfortunately, live scallops within the commercial scallop bed footage are usually partially buried under the sand, making it difficult to differentiate between sea bed and scallops. Thus annotation, whether manual or automatic, relies upon the somewhat darker shadow cast by the protruding perimeter of buried scallops which in many cases is crescent shaped (see Fig. 1).

3 Methodology

Our research breaks down the problem of counting commercial scallops into five main tasks. Frames from the video are first analysed by the iLab Neuromorphic

Visual Toolkit (iNVT) to determine areas of potential interest which are then extracted as greyscale sub-images. Segmentation is performed on the sub-images to create a secondary binary sub-image. Feature extraction is then performed on both sub-images and this information is used to generate data in a suitable format for performing classification via the Weka Machine Learning Toolkit.

3.1 iLab Neuromorphic Visual Toolkit (iNVT)

To reduce the required search space within an image [4] we use the iNVT to identify areas of potential interest prior to performing more complex segmentation and feature extraction techniques. iNVT is a software application developed by the University of Southern California (USC) and is designed to identify salient (conspicuous) regions within an image. Saliency is determined in a number of ways including intensity, orientation and colour.

iNVT generates a saliency map of the image and then uses a winner-takes-all neural network approach to determine the most salient region within an image [5]. It is possible to keep iNVT running for a specified number of attention shifts whereby salient regions are ignored once they have been *found.* iNVT outputs an ordered series of x and y coordinates specifying the centroid of each salient region found in the main image. Each set of coordinates is then fed into MATLAB and a 100 x 100 sub-image is extracted for further processing including segmentation and feature extraction.

3.2 Segmentation

Some of the feature extraction techniques used for classification allow us to use greyscale sub-images without the need to perform segmentation. We also create a dataset based on a binary version of each sub-image. Segmentation is a four stage process including blurring, contrast stretching, thresholding and clean-up.

The sub-image first has a Gaussian blur applied to it to help smooth the details of the sub-image and reduce the likelihood of visible interlacing lines caused from the extraction of still frames from compressed video. Blurring also helps to avoid regions with narrow channels or sections being split into two regions during thresholding.

The sub-image has its contrast stretched to help accentuate the typically darker region of shadow created by the edge of the scallop. Thresholding is then performed around the mean of the newly generated sub-image. The cleanup process is outlined in Fig. 2. We reduce the thresholded sub-image to either zero or one regions of potential interest based on the following rules:

1. All regions touching the sub-image boundary are removed as it is likely that a region located around the edge of the sub-image is not the area of interest that iNVT found.
2. Regions with too small an area (less than 60 pixels) are removed. Visual inspections of the binary sub-images have indicated that areas smaller than 60 pixels are generally featureless.
3. If more than one region still remains in the sub-image after the first two stages are performed, the centroid for each region is calculated and the region with its centroid closest to the centre of the sub-image is chosen as the final region of interest. (The sub-image is generated around the point at which the iNVT

determined a region of potential interest therefore the most central region should be most relevant).

It is important to note that it is possible that regions disposed of in the first and third steps of the cleanup process may still be picked up in another sub-image as it is possible for the iNVT to find two areas of interest within close proximity of each other. Further, when processing a sequence of frames, items not identified in one may be identified in others.

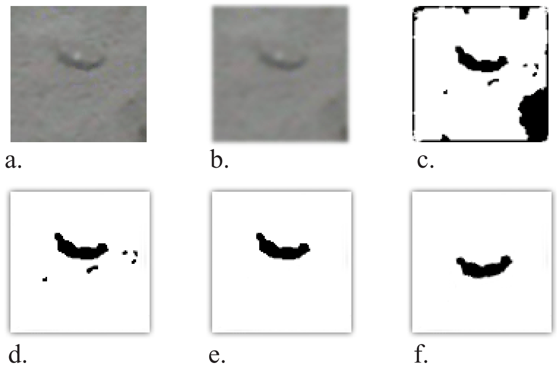


Fig. 2. a) Original sub-image. b) Gaussian blur and contrast stretch. c) Binary sub-image generated using thresholding around the mean. d) Regions touching the border of the sub-image are removed. e) Regions with an area < 60 pixels are removed. f) PCA may be applied to rotate and centre the remaining region around its major axis.

Erosion and dilation were also trialled on the sub-image's regions in an attempt to reduce the breaking up of regions prior to the cleanup process. However it was found that this process resulted in a loss of definition within the regions and provided little help rejoining regions that had been separated by the thresholding process. Ellipse fitting, whereby an ellipse is fitted to the lower margin of a region in an effort to define the area that the scallop occupies, was also unsuccessfully tried as a potential final step in the segmentation process whereby an ellipse could be fitted to the underside of a region in an effort to find the potential space a scallop may occupy.

3.3 Feature Extraction

Various features have been extracted from both the greyscale and segmented binary sub-images. These include: seven invariant moments [6], the order in which an area of interest was initially selected by iNVT, the location and distance of the centroid of a region from the centre of the sub-image and the division of a region into segments (Fig. 3). When dividing a region into segments Principal Components Analysis is used to rotate the region so that its major axis is aligned with the x axis of the sub-image.

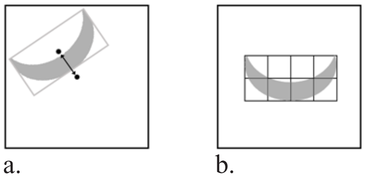


Fig. 3. a) The Distance to Center is measured from the centre of the sub-image to the centroid of the region. X and Y coordinate information can also be extracted as a feature using the location of the region's centroid. b) The region is rotated and centered on its major axis using PCA. In this example the region has been divided using a 4x2 grid. The region's percentage of coverage in each division is then recorded to form a dataset.

Invariant moments were also extracted from the greyscale sub-image using a mask derived from the binary region. The mask was used at its original size and was also grown by varying amounts in order to encapsulate the surrounding areas of the region within the greyscale image. Results for all experiments are discussed in detail throughout section 0.

3.4 Instance Classification

We have consistently used 10 fold cross-validation with the same six classifiers from the Weka Machine Learning Toolkit to generate our results. We consider these classifiers will give us a reasonable representation of the overall performance of the system based on initial trials with other Weka classifiers. The classifiers are as follows:

- Multilayer Perceptron (MLP) - (a Neural Network)
- Naïve Bayes (NB)
- IB1 (Single Nearest Neighbour)
- Multiboost AB (MB-AB)
- NBTree (NBT – Naïve Bayes Tree)
- Decision Table (Dec T)

4 Experiments and Results

It is necessary to manually label each sub-image as either a scallop or non-scallop for system evaluation. This process is an arduous task and we have as yet not had the opportunity to ground truth the data. Consequently, the dataset undoubtedly includes some misclassified instances; however, we are confident that enough instances have been classified correctly to give a clear indication of the system's overall performance. Table 1 and 2 outline how the dataset is affected by the segmentation and feature extraction processes. It also highlights the affects of various sub-image sizes ranging from 30x30 to 100x100. The *exceeds perimeter* row denotes the number of instances that exceed one or more of the original image's perimeters. These instances cannot be extracted as a complete sub-image and are therefore discarded.

100x100 was chosen as the most appropriate size for a sub-image without further tests being performed on larger sub-images. As shown in Table 1, by increasing the size of the sub-image beyond 100x100 we will at most avoid excluding a maximum of four extra scallop regions. At the same time we will reduce the number of useable instances by about 30 if we consider the average increase in the loss of useable instances across the *exceeds perimeter* row. Scallop instances that have been lost in the cleanup process after extraction of the sub-image are currently excluded from any quantitative assessments we make of commercial scallops within the video footage.

Table 1. The effect the sub-image size and cleanup process have on the total number of scallop instances available for training and testing

	30x30	40x40	50x50	60x60	70x70	80x80	90x90	100x100
Exceeds perimeter	76	119	136	172	205	232	262	295
Region on border	624	431	233	105	46	22	13	4
Area < 60	113	194	234	202	134	93	51	47
Training/Testing	14	83	224	348	442	480	501	481

Table 2. The effect the sub-image size and cleanup process have on the total number of non-scallop instances available for training and testing

	30x30	40x40	50x50	60x60	70x70	80x80	90x90	100x100
Exceeds perimeter	293	414	525	619	730	815	900	1005
Region on border	622	428	227	121	61	33	18	8
Area < 60	1752	1691	1594	1384	1092	881	678	519
Training/Testing	32	166	353	575	816	970	1103	1167

The results presented below represent the findings we have made whilst testing the final stage of our multilayered system. All results are presented as the percentage of correctly classified instances in Weka with the Standard Deviation given in parentheses. We have at all times worked with as many instances from the original dataset as possible. A total of 1648 instances (481 scallops and 1167 non-scallops) remain after the cleanup process is complete. Some of the cleanup process experiments resulted in greater losses of instances. In these cases the number of scallop and non-scallop instances classified by Weka has been written next to the input column description. When considering the fact that non-scallop instances account for approximately 70% of the total instances, this sets a minimum baseline level of performance that a classifier must exceed to be at all useful.

The results in Table 3 were generated to test the performance of the system using invariant moments as the only feature extraction technique. By feeding the invariant moments from both sub-images into Weka the average overall performance of the system is improved by approximately 5%.

Table 3. Invariant moment feature extraction using the greyscale and binary images and a combination of both. All sub-images are 100x100

Sub-image	MLP	NB	IB1	MB-AB	NBT	Dec T	AVG
Grey & Binary	76 (3.7)	71 (3.1)	71 (5.1)	71 (1.7)	77 (3.8)	76 (3.3)	**74**
Binary	73 (1.9)	61 (3.3)	67 (3.8)	71 (1.0)	73 (1.7)	70 (2.2)	**69**
Grey	71 (2.4)	62 (3.1)	67 (2.3)	71 (0.1)	71 (3.7)	74 (3.4)	**69**

Table 4 outlines the impact of not discarding instances when the segmented region has an area less than 60 pixels. The addition of these instances increases the occurrence of non-scallops to approximately 75% and ultimately reduces the overall performance of the system compared to the best average result of 74% in table 4. This confirms the findings from our visual inspections that scallop instances with an area less than 60 are likely to lack features suitable for classification against non-scallop instances.

Table 4. Performance of grey and binary sub-image when no area rule is applied to the cleanup process resulting in a greater number of instances for classification

Sub-image - No Area Rule	MLP	NB	IB1	MB-AB	NBT	Dec T	AVG
No Area, Grey & Binary (523/1247)	78 (2.3)	69 (4.2)	74 (3.2)	76 (1.8)	77 (1.6)	77 (1.8)	**75**

Table 5 demonstrates that by including a distance to centre (DTC) measurement from the centroid of the segmented region to the centre of the sub-image and the X and Y coordinate of this centroid we can marginally improve the performance of the system compared to the results in Table 4.

Table 5. Invariant moments from greyscale and binary sub-images with the inclusion of a distant to centre (DTC) measurement and the X and Y coordinate of the segmented regions centroid

Sub-image	MLP	NB	IB1	MB-AB	NBT	Dec T	AVG
DTC	78 (2.7)	72 (3.0)	73 (5.0)	71 (0.4)	78 (4.1)	79 (3.9)	**75**
DTC, X&Y	78 (3.4)	74 (2.8)	73 (4.2)	71 (0.4)	77 (4.9)	79 (3.1)	**75**

Applying a mask to the greyscale image (using the segmented binary region) does little to improve the overall performance of the system. The mask is grown by a specific amount (outlined in Table 6) and is used as a means of encapsulating features in close proximity to the segmented region in the binary sub-image. The system performs at its best when the mask is grown by 10 to 15 pixels and a slight advantage can be gained by including the distance to centre measurement and the coordinates of the region's centroid. However this advantage is marginal and when we compare the results of the individual classifier learning methods in Tables 5 and 6 we see that the best performing classifiers in Table 5 have their performance reduced in table 6 suggesting that although the average performance is similar, the addition of a mask into our system offers no real advantage.

Table 6. Invariant moments using the segmented image as a mask over the greyscale image

Sub-image mask	MLP	NB	IB1	MB-AB	NBT	Dec T	AVG
5x5	76 (2.3)	70 (3.5)	73 (4.9)	74 (4.7)	74 (5.1)	74 (4.2)	**74**
10x10	77 (2.9)	72 (4.2)	74 (3.4)	75 (3.2)	76 (3.1)	76 (2.8)	**75**
15x15	79 (1.7)	73 (2.9)	72 (3.5)	76 (2.9)	77 (3.3)	76 (3.0)	**76**
20x20	77 (2.7)	71 (2.3)	74 (3.5)	74 (3.0)	77 (2.6)	76 (1.8)	**75**
15x15, DTC, X & Y	78 (3.4)	73 (2.7)	75 (3.7)	75 (3.2)	78 (3.8)	78 (4.4)	**76**

The results presented in Table 7 outline the system's performance on the classification of the segmented binary sub-image when its region is divided in *n x m* sections. This approach outperforms all other results presented in this paper when the region is divided into 4 x 2 sections, suggesting that feature extraction methods that help to define the shape of the segmented region may help to improve the overall performance of the system. The results in the last row of Table 7 demonstrate the negative impact combining invariant moments and distance to centre features has on the overall classification performance.

Table 7. Classification of the percentage of area of a region divided into *n x m* sections

Region division X x Y	MLP	NB	IB1	MB-AB	NBT	Dec T	AVG
2x1	71 (0.2)	70 (0.1)	61 (3.0)	71 (0.1)	71 (0.1)	71 (0.1)	**69**
2x2	70 (1.4)	71 (0.1)	63 (3.5)	71 (0.1)	71 (0.1)	71 (0.1)	**70**
3x2	76 (3.7)	76 (3.0)	68 (3.3)	72 (2.3)	77 (2.6)	76 (2.3)	**74**
3x3	76 (3.2)	76 (2.2)	69 (4.0)	73 (2.3)	75 (2.5)	74 (2.7)	**74**
4x2	83 (2.3)	81 (2.0)	81 (2.1)	77 (1.8)	82 (2.1)	80 (2.5)	**81**
4x3	75 (1.7)	75 (3.3)	71 (3.7)	72 (1.8)	74 (3.1)	76 (3.4)	**73**
4x4	74 (3.4)	74 (2.0)	71 (3.1)	72 (2.1)	76 (1.9)	76 (2.9)	**74**
4x2, DTC, Inv-M	78 (2.5)	73 (2.8)	74 (2.7)	74 (3.2)	77 (3.9)	80 (3.2)	**76**

Table 8 shows more details of the performance of the six classifiers on the 4x2 segmentation results in Table 7. After examining the more detailed performance of all the results presented in this paper we have discovered that in more than 95% of cases the IB1 classifier produces a balanced error rate of no more than 2% difference between false positives and false negatives. From a purely quantitative assessment perspective this information, if proved sufficiently consistent, could provide us with a greater level of accuracy when assessing scallop abundance. However from an Artificial Intelligence perspective it is desirable to continue to work towards reducing the error rate.

Table 8. Breakdown of the classification results for 4x2 segmentation in Table 7

Segmentation 4x2	TP	TN	FP	FN
MLP	341	1022	145	140
NB	344	990	177	137
IB1	327	1009	158	154
Multi-AB	417	847	320	64
NB-Tree	314	1043	124	167
Dec-Table	313	1002	165	168

5 Conclusions and Further Work

A substantial amount of the research we have undertaken so far has been in developing a suitable multilayered system capable of automatically annotating commercial scallop bed video footage. Our results so far show promise but also leave room for improvement.

The experimentation using only invariant moments demonstrate that this approach produces at best mediocre results. The results produced by dividing the segmented region into sections and measuring the area occupied by the region in each section indicate that the emphasis of further work should be on finding feature extraction techniques that better describe the overall shape of the segmented region. This work will include tests using sampling techniques and pattern classification using an n-tuple classifier system [7].

The research and results covered within this paper are based on still images extracted from the video footage. iNVT is also capable of working with multiple frames or video sequences as needed for integrating video footage into the process to provide an automated system. It will be necessary to develop a tracking system for *found* instances across consecutive frames to avoid counting scallops more than once.

We are currently working on a new dataset which will include a minimum of 500 scallop instances. This dataset will be thoroughly scrutinised for accuracy and will be used primarily for testing classifier learning performance. We are also looking at broadening the scope of the system to incorporate other underwater domains with similar characteristics. We have recently obtained video footage of seahorse activity and intend to test our system on this footage in the near future.

References

1. Wilson, A.: First steps towards autonomous recognition of Monterey Bay's most common mid-water organisms: Mining the ROV video database on behalf of the Automated Visual Event Detection (AVED) system. Technical Report, MBARI, CA, USA (2003)
2. Walther, D., Edgington, D.R., Koch, C.: Detection and Tracking of Objects in Underwater Video. In: Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition, pp. 544–549 (2004)
3. Witten, I.H., Frank, E.: Data Mining: Practical Machine Learning Tools and Techniques, 2nd edn. Morgan Kaufmann, San Francisco (2005)
4. Itti, L., Koch, C., Niebur, E.: A Model of Saliency-Based Visual Attention for Rapid Scene Analysis. IEEE Transactions on Pattern Analysis and Machine Intelligence 11, 1254–1259 (1998)
5. Itti, L.: The iLab Neuromorphic Vision C++ Toolkit: Free tools for the next generation of vision algorithms. In: The Neuromorphic Engineer (2004)
6. Gonzalez, R.C., Woods, R.E.: Digital Image Processing using MATLAB. Prentice Hall, Englewood Cliffs (2004)
7. Schechner, Y.Y., Karpel, N.: A Trainable n-tuple Pattern Classifier and its Application for Monitoring Fish Underwater. In: 7th International Conference on Image Processing and its Applications. Manchester, UK, pp. 255–259 (1999)

Multiple Classifier Object Detection with Confidence Measures

Michael Horton, Mike Cameron-Jones, and Raymond Williams

School of Computing, University of Tasmania, TAS 7250, Australia
{Michael.Horton, Michael.CameronJones, R.Williams}@utas.edu.au

Abstract. This paper describes an extension to the Haar Classifier Cascade technique for object detection. Existing Haar Classifier Cascades are binary; the extension adds confidence measurement. This confidence measure was implemented and found to improve accuracy on two object detection problems: face detection and fish detection. For fish detection, the problem of selecting positive training-sample angle-ranges was also considered; results showed that large random variations that result in cascades covering overlapping ranges increases their accuracy.

1 Introduction

Object detection – finding instances of some class of object in two-dimensional images – is a common problem in computer vision. It is frequently studied in the form of human face detection. Many Artificial Intelligence techniques have been applied to the object detection problem. This paper describes an extension to the Haar Classifier Cascade method by adding confidence estimates. Its application to face and fish detection is considered and tested.

1.1 Haar Classifier Cascades

This approach to object detection classifies objects by how well they match simple rectangular patterns of light and dark. It was first proposed and implemented by Viola and Jones [1]. An image that may represent the object is classified as follows:

1. A *classifier* steps through several *stages* in turn; it stops and returns false if a stage returns false. If all stages return true, the classifier returns true.
2. A *stage* returns true if the sum of its *feature* outputs exceeds a chosen threshold.
3. A *feature* returns the sum of its *rectangle* outputs.
4. A *rectangle* returns the sum of pixel values in the image region bounded by that rectangle, multiplied by a chosen weight.

As any stage which returns false will cause the entire classifier to return false, false negatives are very harmful. The machine learning technique of boosting [2,3] is applied within each stage's training to keep false negatives to a minimum. In boosting, a classifier is trained and tested, then the training instances are reweighted to emphasise those that were misclassified. A new classifier is then trained, and the process repeated several times. All classifiers trained in the boosting process are used, but their outputs are weighted based on their accuracy.

A Haar Classifier Cascade for a given image will return true if the image *represents* the object; to determine if an image *contains* the object, the cascade must be run over every location in the image, at a variety of scales. The standard implementation will also only detect objects with similar orientation to objects in its training set. Jones and Viola observed that 'a classifier trained on all poses appears to be hopelessly inaccurate' but were able to detect faces at multiple orientations by combining the output of multiple cascades, each trained on faces with different orientation [4]. Each classifier covered faces with an in-plane rotation range of 30°. They also trained a decision tree to select the best orientation for each possible image region, which made their multiple orientation classifier faster, but less accurate. Kölsch and Turk investigated this as a method for hand detection [5]; they found that 'only about 15° of rotations can be efficiently detected with one detector, different from the method's performance on faces' and 'More importantly, the training data must contain rotated example images within these rotation limits'.

Existing enhancements to the cascade training process include threshold adjustment to minimize overall misclassification cost [6], use of motion-based features [7] and domain-specific adjustments to the boosting process [8,9].

1.2 Classifier Confidence

Some classifiers only return a single value giving the class they predict. Other classifiers give the probability of each class being true – a confidence estimate. On binary problems, this allows a user to specify a trade-off between false positives and true positives. Users may also specify a cost matrix, and select the classification whose sum of probability multiplied by misclassification costs is lowest.

1.3 Fish Detection

The intended application for the fish detector involves determining average fish size by detecting a sample of unoccluded fish in each image and matching a shape model to each fish. This is an extension of previous work on the same set of images [10]. Because the shape model only needs unoccluded fish, the evaluation process should only count undetected fish as false negatives if they are nearly (90%–100%) completely visible. If occluded shapes must be detected, the problem of training a Haar Classifier Cascade to find them has been studied [11]. Detections not representing fish are counted as false positives, while detections of partially (50%–90%) visible fish are regarded neither as true positives nor as false positives, as the precise ability of the later stages of the application to cope with such detections is not yet known.

In general, the fish cage environment is less varied than many object detection environments, but the underwater images have poor contrast and are frequently crowded (Fig. 5, [12,13]).

2 Haar Classifier Confidence

While the Haar Classification process described above is a binary classifier, its intermediate values are numeric and continuous. This suggests that the stages could derive a confidence measurement from the features, and the classifier could combine the stage confidences to return an overall confidence measurement. The method proposed and implemented here returns the confidence for a region by modifying the first two steps as follows:

1. A *classifier* steps through several *stages*, adding up their outputs; it stops if any stage returns negative. The confidence returned is this cumulative sum.
2. A *stage* returns the sum of its *feature* outputs minus its threshold.

2.1 Uses

The confidence measurement described above derives more information from a cascade than the binary classification does. This information may lead to more accurate classifiers. Two potential ways to use this information are proposed: hill-climbing to maximise the confidence of regions found by a binary classifier and measuring confidence across an entire image to build a 'confidence map'.

Hill-climbing. Here, the binary classifier is applied to find possible object regions in the image. The confidence-based classifier then considers each such region and tries small scale and position adjustments in a hill-climbing approach to improve its confidence. The process repeats until no improvements can be made. In operation, the mean number of hill-climbing steps required for all regions detected on a dataset was 1.5–1.7.

The exact scale and position adjustments implemented for a $w \times h$ pixel region cover the 9 scale adjustments from 1.05^{-4} to 1.05^{4} and the 81 position adjustments from $-\frac{w}{4}, -\frac{h}{4}$ pixels to $+\frac{w}{4}, +\frac{h}{4}$ pixels.

Confidence Maps. If the cascade confidence is measured across the image for a variety of region sizes, it is possible to build a 'confidence map' of the image. This will illustrate the confidence that there is an object centred upon each point in the image; the size of the most confident region centred on that pixel is also stored. To screen out similarly-placed positives, confidence indicating a $w \times h$ pixel positive region is only taken into account if no higher confidence region exists centred within $\frac{w}{4} \times \frac{h}{4}$ pixels. If a binary classification is required, a confidence threshold can be chosen, based on its trade-off between true positives and false positives. Fig. 5(e) illustrates a confidence map; fig. 5(f) shows the local maxima selected from that map.

3 Method

3.1 Implementation

The OpenCV graphics library[1] was used as a starting point for this work; it contains basic Haar Classifier training and testing source code. Additional routines were written for training on rotated regions, combining classifier outputs, hill-climbing and confidence mapping.

3.2 Face Detection

OpenCV is provided with four face detectors trained by Lienhart [14]. The most accurate cascade, haarcascade_frontalface_alt2, was selected for experimentation on the confidence-based methods. It was tested on 33 images from the MIT/CMU frontal face testing dataset [15], which contain a total of 143 faces.

3.3 Fish Detection

The provided dataset contained 235 fish images, which were split into equal-sized training and testing sets. As noted in section 1.3, the intention was to train a (nearly) fully-visible fish detector, and to test it on its ability to find mostly (90%–100%) visible fish. With that in mind, the training images were annotated with the locations of 409 mostly-visible fish and 722 regions containing no fish. Within the testing images, 597 mostly-visible fish were marked, along with 1,759 partially (50%–90%) visible fish to prevent excess false positives if those fish were detected.

Seven classifier cascades were trained, at 15°increments on angles from -45°to +45°. The dimensions of each cascade were chosen to detect objects with an aspect ratio of 3:1 and to have an area equivalent to $48 \times 16 = 768$ pixels (Table 1). This would not detect all possible fish, as it does not cover every possible orientation, but was sufficient for the images being used. Existing research on hand detection suggested that the positive training examples for a given angle should not be fixed at that angle, but should be perturbed by some random amount [5], as illustrated in Fig. 1. This was tested, with random variations of 0°to 90°. Each cascade was trained to 20 stages, with a maximum stage false positive rate of 50% and maximum stage false negative rate of 0.5%. Following Lienhart [16], the Gentle AdaBoost boosting algorithm was selected for boosting each stage.

[1] http://www.intel.com/technology/computing/opencv/

Table 1. Fish detection classifier region sizes in pixels

Angle	Width	Height
0°	48	16
15°	37	21
30°	32	24
45°	28	28

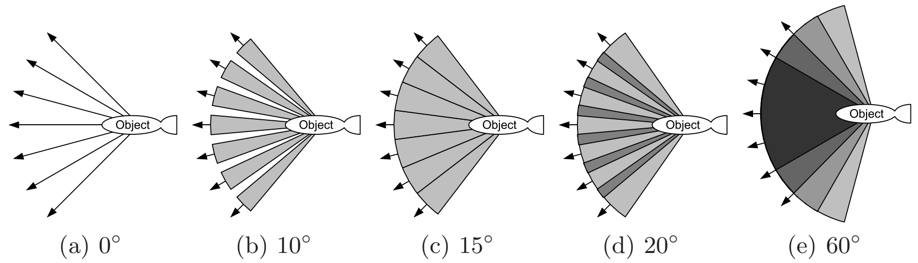


Fig. 1. Positive sample orientations for 7 cascades fixed on angles from -45° to +45° with random angle variation; darker areas show where random variations overlap

3.4 Evaluation

Object detection algorithms rarely return a positive for the exact image region marked in a human-created test set. Detectors must therefore be evaluated by their ability to almost match the position and size of the test regions. For these, the OpenCV default evaluation settings were used. These match a $w \times h$ pixel positive region to a $w_d \times h_d$ detection made s pixels away if $s < 0.3\sqrt{0.5(w^2 + h^2)}$ and $\frac{2}{3}w < w_d < \frac{3}{2}w$.

Following Lienhart [14], ROC curves for binary and hill-climbing classifiers were created by requiring multiple nearby positives to count as a positive classification. These were extended by reducing the number of stages in the cascade once the number of positives required reached zero. For confidence mapping, ROC Curves were generated by varying the confidence threshold required to return a positive classification.

4 Results

4.1 Face Detection

Hill-climbing to maximise confidence produced small consistent improvements in accuracy over the simple binary cascade, while confidence mapping resulted in a larger improvement (Fig. 2).

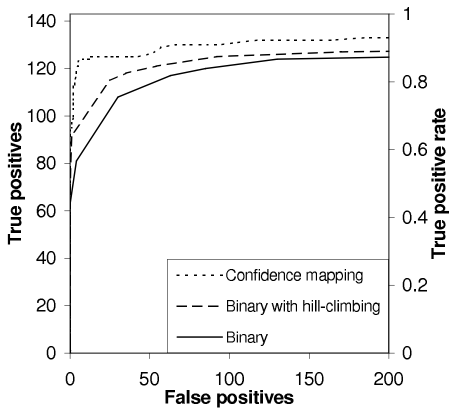


Fig. 2. ROC curves for face detection using a binary cascade, binary cascade followed by hill-climbing, and confidence mapping

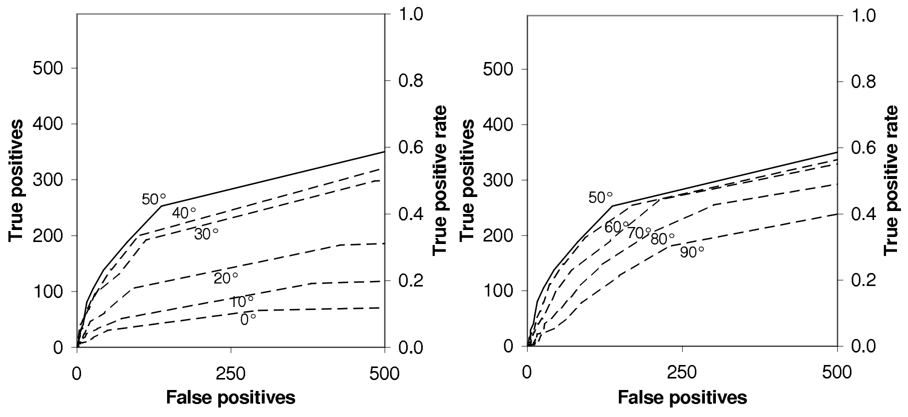
4.2 Fish Detection

The angle ranges needed for effective fish detection were quite high. Since the cascades were trained to detect fish in orientations 15°apart, they could be expected to perform best when the positive samples were randomly perturbed by 15°(±7.5°). Instead, the most effective cascade combination used random variation of 50°(±25°) (Figs. 3(a),3(b)). The best hill-climbing results for confidence mapping were similar, although 60°(±30°) variations had a slight advantage (Figs. 3(c),3(d)). When the best binary, hill-climbing and confidence mapping results were compared, the hill-climbing had a small advantage over the basic binary classifier, while confidence mapping was significantly worse (Fig. 4). Annotations used and detections made are shown in Fig. 5.

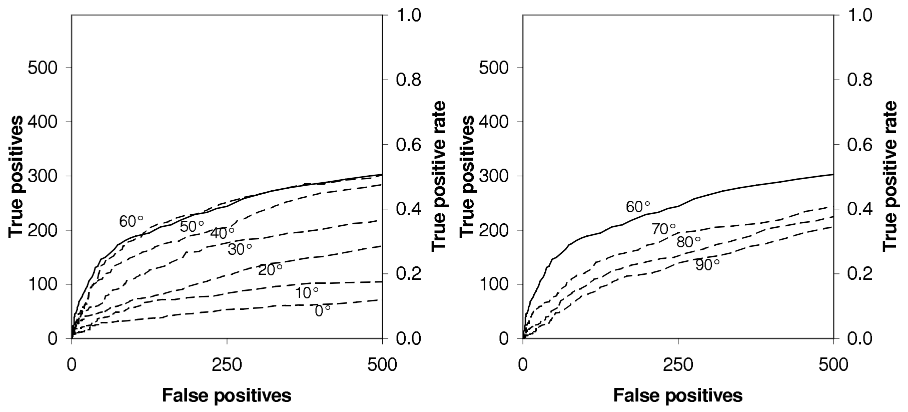
4.3 Confidence Modifications

Two modifications to the cascade confidence algorithms were also tested: permitting multiple negative stages and changes to the stage order. While the basic confidence calculation stops as soon as one stage returns negative, it is possible to let some or all stages return negative while adding up the final confidence. Changes to the stage order will also affect the final confidence for locations which are not passed by all stages; both reversed and random stage orders were tested.

Neither modification made any significant difference to either face or fish detection accuracy. Since both add complexity, and continuing to calculate accuracy after some stages return negative confidence increases running time, these changes were discarded.



(a) Binary cascades, 0°-50°variation (b) Binary cascades, 50°-90°variation



(c) Confidence map, 0°-60°variation (d) Confidence map, 60°-90°variation

Fig. 3. ROC curves for fish detection with different random angle variations

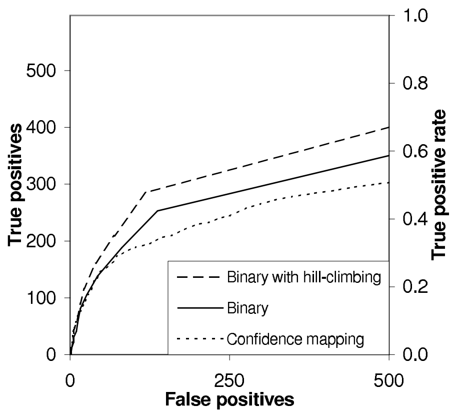
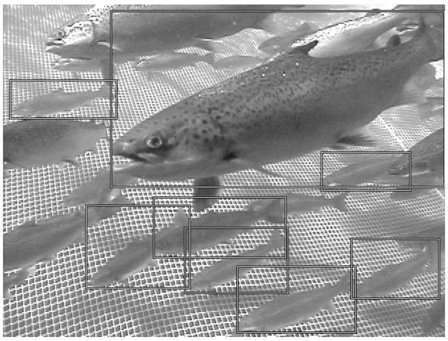
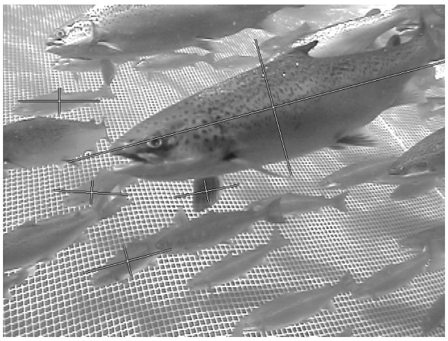
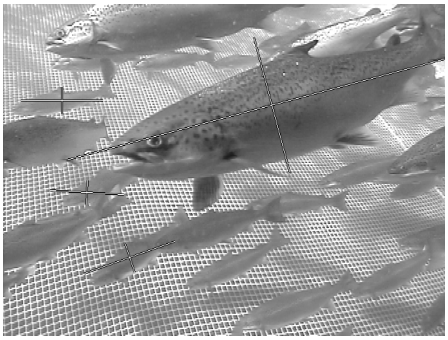
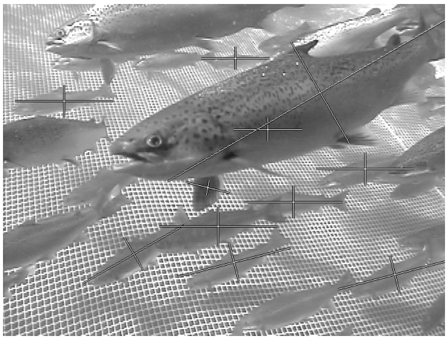


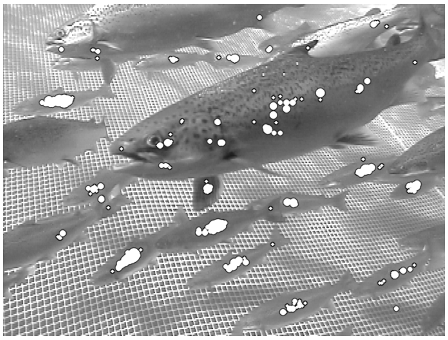
Fig. 4. ROC curves for fish detection using binary cascades, binary cascades followed by hill-climbing, and confidence mapping


(a) Image annotated with (nearly) fully-visible fish


(b) Binary cascades


(c) Binary cascades followed by hill-climbing


(d) Confidence mapping detections


(e) Confidence mapping visualisation

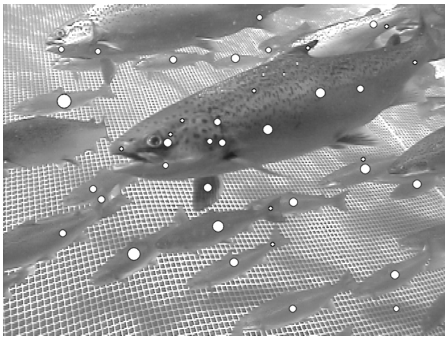

(f) Confidence mapping local maxima

Fig. 5. Example fish image with annotations and detections (image provided by AQ1 Systems Pty. Ltd.)

5 Conclusions and Further Work

A method for deriving confidence estimates from the Haar Classifier Cascade method has been invented and implemented. It was tested as a means of both adjusting cascade output using hill-climbing, and of classifying points in an image through confidence mapping. Hill-climbing improved the accuracy of both face and fish detection cascades, while confidence mapping was effective at face detection but poor on fish detection.

The problem of detecting fish at multiple in-plane orientations through combining multiple classifiers was also considered. While previous work suggested that the positive training samples should be randomly perturbed over a range equal to the separation between adjacent cascades, the most accurate fish cascades were trained with random angle variation of $50°(\pm 25°)$, a much larger range than the separation between cascades (15°). This property may also apply to other problems in detecting rotated objects.

Further work remains in investigating the poor performance of confidence mapping on the fish detection problem. This may be a consequence of the cascades being quite inaccurate compared to the existing face detectors, or it may result from the use of multiple classifiers on the same image.

Acknowledgements

This research was supported by a Tasmanian Graduate Research Scholarship. We would also like to thank AQ1 Systems Pty. Ltd. (www.aq1systems.com) for providing the fish images and the anonymous reviewers for their comments.

References

1. Viola, P., Jones, M.: Rapid object detection using a boosted cascade of simple features. In: Proc. IEEE International Conference on Computer Vision and Pattern Recognition, vol. 1, pp. 511–518 (2001)
2. Freund, Y., Schapire, R.E.: Experiments with a new boosting algorithm. In: Proc. International Conference on Machine Learning, vol. 148, p. 156 (1996)
3. Freund, Y., Schapire, R.: A short introduction to boosting. Journal of the Japanese Society for Artificial Intelligence 14(5), 771–780 (1999)
4. Jones, M., Viola, P.: Fast multi-view face detection. Mitsubishi Electric Research Lab TR-20003-96 (2003)
5. Kölsch, M., Turk, M.: Analysis of Rotational Robustness of Hand Detection with a Viola-Jones Detector. In: Proc. IAPR International Conference on Pattern Recognition (2004)
6. Brubaker, S., Wu, J., Sun, J., Mullin, M., Rehg, J.: On the Design of Cascades of Boosted Ensembles for Face Detection. Georgia Institute of Technology Technical Report GIT-GVU-05-28 (2005)
7. Viola, P., Jones, M., Snow, D.: Detecting Pedestrians Using Patterns of Motion and Appearance. International Journal of Computer Vision 63(2), 153–161 (2005)
8. Li, S., Zhang, Z.: FloatBoost learning and statistical face detection. IEEE Transactions on Pattern Analysis and Machine Intelligence 26(9), 1112–1123 (2004)

9. Xiao, R., Zhu, L., Zhang, H.: Boosting chain learning for object detection. In: Proc. IEEE International Conference on Computer Vision, pp. 709–715 (2003)
10. Williams, R.N., Lambert, T.J., Kelsall, A.F., Pauly, T.: Detecting Marine Animals in Underwater Video: Let's Start with Salmon. In: Proc. Americas Conference on Information Systems (2006)
11. Lin, Y.Y., Liu, T.L., Fuh, C.S.: Fast Object Detection with Occlusions. In: Pajdla, T., Matas, J(G.) (eds.) ECCV 2004. LNCS, vol. 3021, pp. 402–413. Springer, Heidelberg (2004)
12. Lines, J.A., Tillett, R.D., Ross, L.G., Chan, D., Hockaday, S., McFarlane, N.J.B.: An automatic image-based system for estimating the mass of free-swimming fish. Computers and Electronics in Agriculture 31(2), 151–168 (2001)
13. Mueller, R.P., Brown, R.S., Hop, H., Moulton, L.: Video and acoustic camera techniques for studying fish under ice: a review and comparison. Reviews in Fish Biology and Fisheries 16(2), 213–226 (2006)
14. Lienhart, R., Maydt, J.: An extended set of Haar-like features for rapid object detection. In: Proc. International Conference on Image Processing, vol. 1 (2002)
15. Rowley, H., Baluja, S., Kanade, T., Sung, K., Poggio, T.: Frontal Face Images (2003), http://vasc.ri.cmu.edu/idb/html/face/frontal_images/
16. Lienhart, R., Kuranov, A., Pisarevsky, V.: Empirical analysis of detection cascades of boosted classifiers for rapid object detection. In: DAGM 25th Pattern Recognition Symposium (2003)

Agent-Based Distributed Energy Management

Jiaming Li, Geoff Poulton, and Geoff James

CSIRO Information and Communications Technology Centre
Locked Bag 17, North Ryde NSW 1670, Australia
{firstname.lastname}@csiro.au

Abstract. This paper describes our research in technologies for the management and control of distributed energy resources. An agent-based management and control system is being developed to enable large-scale deployment of distributed energy resources. Local intelligent agents will allow consumers who are connected at low levels in the distribution network to manage their energy requirements and participate in coordination responses to network stimuli. Such responses can be used to reduce the volatility of wholesale electricity prices and assist constrained networks during summer and winter demand peaks. The management and control of very large numbers of distributed energy resources to create aggregated quantities of power can be used to improve the efficiency of the electricity network and market. In our system, the coordination of energy resources is decentralized. Energy resources coordinate each other to realize efficient autonomous matching of supply and demand in large power distribution networks. The information exchange is through indirect (or stigmergic) communications between resource agents and a broker agent. The coordination mechanism is asynchronous and adapts to change in an unsupervised manner, making it intrinsically scalable and robust.

1 Introduction

With the increasing gap between electricity supply and demand, the electricity industry in many countries is facing a number of new pressures. Distributed electricity generation technologies together with improved demand-side management techniques have been identified as a possible solution to this challenge [1]. The idea of controlling the switching of loads and generators to respond to price signals and network constraints is technically achievable and becoming more economically viable for businesses requiring greater supply reliability, flexibility, and lower cost to the consumers.

We have been developing multi-agent technology for the management and control of distributed energy resources [2-3], aimed at deployment in the Australian National Electricity Market within the next five years. A component of this work is the development of algorithms for coordinating distributed energy resources (DERs) comprising customer loads and generators. DERs are coordinated to aggregate sufficient distributed capacity to be of strategic value to market participants such as retailers and network businesses. Such aggregation is a significant challenge, particularly for large numbers of DERs and when centralized control techniques are not feasible.

Three methods of improving distribution have been reported in recent literature and are currently under trial [4]-[7], [2].

1. Price-based control [4]-[6], where human, “owners’’ of each resource are asked to respond to a varying, broker-determined price for power. Although mitigation of peak-period power usage has been reported, the disadvantages of the method are: (1a) Human owners may not exist for some resources, or may not be able or willing to respond when asked. (1b) There is no guarantee of the level of service to resources. (1c) The process may lead to customer dissatisfaction since it requires effort from them and they are being asked to choose between cost and comfort or convenience.
2. Agent-based market-oriented algorithms [7], with real or virtual currency, where one or more broker agents carry out a negotiation process with each resource agent to fix usage and price. Again, some performance advantages have been reported, but this method also has disadvantages, which are: (2a) Lack of simple scalability - existing market-based algorithms require hierarchies of brokers to negotiate with very large numbers of resources, leading to potentially fragile structures. (2b) Market-based algorithms also require adaptation or replication to account for relationships between resource controls actions at different times arising from their physical properties. (2c) Although the efficiency of market-based algorithms may be quantified there is no reported guarantee of an adequate level of service at resource or system level.
3. Central planning algorithms [2] to coordinate a group of distributed energy agents. One such algorithm developed by the authors combines predicted environmental conditions, models for the constraints and behaviour of loads and generators, and a system goal to calculate plans for each resource for a period into the future. A centralized genetic optimization algorithm was used in [2] to simultaneously calculate the plans for each resource. Although the plan can coordinate distributed agents under ideal situations, e.g. accurately predicted environmental conditions and no sudden changes for any agent, the disadvantages are: (3a) Lack of scalability to large numbers of resources - as the behaviour of all agents was optimized for a particular set of events the solution was not expected to scale well, particularly as genetic algorithms were used as the optimization tool and the assembly of agents must satisfy system global as well as local goals. (3b) No adaptability to changes in both local and global conditions - sudden changes in the situation of one or more agents are not anticipated or accounted for. For example, suppose that large quantities are added to or removed from a cool room; such an event would require re-optimization of the whole system.

In this paper we introduce an innovative distributed multi-agent coordination algorithm which coordinates distributed energy resources by attempting to enforce a time-variable supply cap on the power drawn from the grid. The information exchange is through indirect (or stigmergic) communications between resource agents and a broker agent. The coordination mechanism is asynchronous and adapts to change in an unsupervised manner, making it intrinsically scalable and robust. In the system, individual agents are selfish but compliant agents, and reasonably simple. However, the desired (complex) system response emerges out of low level agent coordination, which is in stark contrast to traditional centralized control systems. This work will bring potential solutions to the volatility of wholesale pool prices and an alternative way of dealing with network constraints during summer and winter peaks.

The method overcomes all of the difficulties mentioned above in regard to existing methods. Specifically, no human action or effort is required at the resource level; solutions implicitly include the satisfaction of (local) resource constraints and also offer system-level users a defined service, the reliability or `firmness' of which may be quantified; the system is scalable to very large numbers of resources, even with a single broker; resource agents act autonomously, so no central adaptation or replication is required when conditions change.

The paper will be organized as follows. Section 2 introduces the distributed energy resource agent coordination system. Section 3 describes how the system works based on our coordination algorithm and real time electricity market information. Section 4 gives a series of simulated experimental results and section 5 concludes the paper.

2 Distributed Energy Resource Agent Coordination System

In order to design a simpler, faster and scale-independent algorithm, a new distributed energy resource multi-agent coordination system has been developed. This system, which is shown in Fig.1, consists of four components:

- A group of resource agents. Each resource agent is the decision maker and controller of the resource. It can measure key properties of its resource, detecting changes and adapting its responses as required; can actively control to satisfy local and global goals; is selfish, in that it seeks to satisfy local goals first; cannot communicate directly with other agents, rather, messaging is carried out using a form of stigmergy, via a common stigspace (bulletin-board or similar); is compliant, in that it will always act to help satisfy broker goals as long as its own local goals are not compromised.
- One or more broker agents. Each broker agent makes decisions and exercises control on behalf of the broker. It receives information on predicted market and network usage and prices; interacts with resource agents through stigspace, where it can read and place information. It constructs global goals using market and predicted usage data. An example is a (possibly variable) grid supply "cap" on total power usage for a certain period of time.
- A "stigspace" (web-based bulletin board or similar) which is used for indirect (stigmergic) information exchange, and on which all agents may both place and read messages.
- A summarizing agent. It acts on stigspace information to produce derived information (for example predicted total resource agent demand over time). Such derived information is also placed in stigspace.

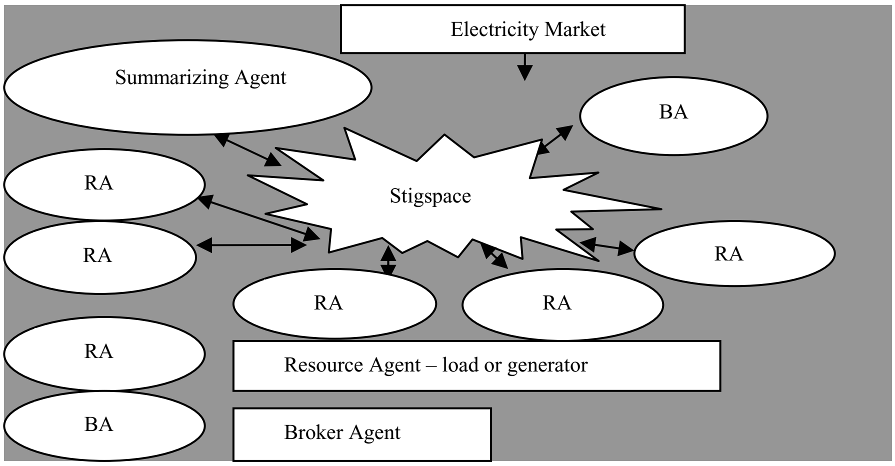


Fig. 1. Coordination System

3 How the System Works

The resource agents have information about their own local constraints and the environment. At convenient intervals the resource agents apply these constraints to a physical model of their resource to calculate a plan for electricity demand or supply for a period into the future. These plans are transferred into market cycles, and then sent to stigspace.

In stigspace summary data is computed from the plans by a summarising agent, the simplest summary data being the total predicted power demand in each interval. This is then made available to any resource agents that wish to use it and also to the broker agent.

The broker agent has knowledge of predicted electricity market price as well as information about the plans communicated by participating resource agents. The broker acts for electricity market participants, such as retailers and network operators, who provide additional information leading to a desired cap on the total demand for power drawn from the grid. This supply cap is placed in the stigsapce and is made available to any resource agents that wish to use it.

Although the broker agent has no direct control over resources, reward is set by prior agreement with resource agents, e.g., resultant overall cost savings passed on to resource agents. No auction is needed. Resource agents agree to help satisfy any global goals as long as local goals remain satisfied. Therefore, the resource agents, when they see total demand and supply cap from stigspace, can revise their plans using our CordCap algorithm (described in section 3.2) while continuing to adhere to their local constraints. The agents submit revised plans and the process is iterated in real time until it stabilises. The process is asynchronous: no explicit coordination is needed between plan submission, plan summing, and broker action.

When total demand is stable, the broker agent is in a position to buy power for the next time period. The process is repeated for every market cycle.

In Australia, the basic electricity market interval is 5 minutes. In our system, the constraints for resource agents are temperature bounds for a heating/cooling environment, and the plan calculated for electricity demand or supply is for the next half hour.

3.1 Australian Electricity Market

In Australia, the National Electricity Market Management Company (NEMMCO) was established to manage the operation of the wholesale electricity market and security of the power system. NEMMCO plays a central role in ensuring South-Eastern Australia's electricity supply through its responsibilities as market and system operator of the National Electricity Market (NEM). Within the NEM producers submit bids stating the amount of energy they can generate at what cost and consumers submit predictions for consumption. These are matched centrally, the lowest cost producers are instructed to supply energy, and a single price is set for all participants. This planning process is based on short-term forecasts of the volume of energy required over the next 24 hour period. Generators are scheduled in 5 minute dispatch intervals. Prices are set for each dispatch interval and provide a signal by which consumers can manage their individual demand. The dispatch and prediction intervals influence the time scale at which electricity management agents can operate. The large volumes of electricity used in the NEM make it impossible to store energy for future use. This means that the NEM is unable to respond quickly to significant unpredicted changes in demand. On the whole, the less oscillation there is in demand, the better. The unpredicted demand in NEM is reflected in NEMMCO peak electricity price.

3.2 The CordCap Algorithm

The CordCap algorithm is used by each resource agent to modify its power usage so as to help satisfy global goals – here the grid supply cap - and has been designed for agents whose actions are limited to on/off load switching. If the supply cap is not satisfied for certain time intervals, the agent's switching strategy in those intervals will be updated as described below. Each agent iterates the process until either (a) local and global goals are satisfied, (b) no further improvement is possible, or (c) a specified time limit is reached. Once this occurs, all agent actions are "locked in" for the next 5-minute interval, the 30-minute planning period is advanced by five minutes and the process begins again.

In the CordCap algorithm the resource agent modifies its predicted switching sequence to shift power consumption from each cap-violating interval into its left- and right-hand neighbours. The process includes three steps, and is carried out for all offending intervals. 1) Locate a random point t_x in the offending interval. 2) Partly shift power usage in the interval on the left and right of t_x into the left- and right-hand neighbouring intervals respectively. 3) Revise the resultant switching strategy to satisfy the resource agent's local constraints.

Fig. 2 illustrates the steps taken by the resource agent with a single violating interval. The black blocks show the time when power is being used. T_{max} and T_{min} are upper and lower temperature constraints of the cool-room and the curve is the

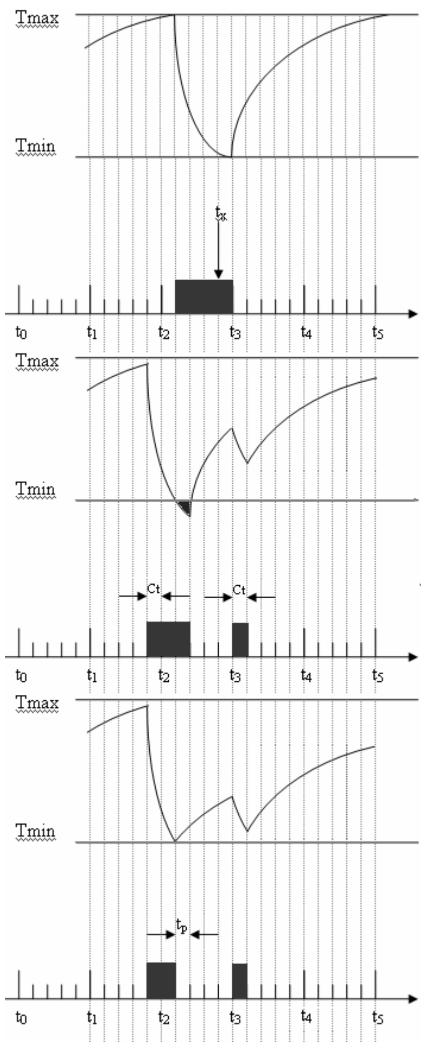


Fig. 2. Coordinating with Single Violating Interval

predicted internal temperature based on the resource model and currently planned power usage. If two or more intervals violate the cap, each interval is treated independently.

Step1: A time t_x is randomly selected between t_2 and t_3;

Step2: Power in $[t_2,t_x]$ (if any) is shifted so that its left-hand edge is C_t minutes outside the left interval boundary. Similarly, power in $[t_x,t_3]$ (if any) is shifted so that its right-hand edge is C_t minutes outside the right interval boundary. C_t has been determined by experiment to work for a wide range of loads.

- If the violating interval is k = 1, i.e., the period $[t_0, t_1]$, the power outside left interval boundary t_0 will vanish.
- If the violating interval is k = 6, i.e., the period $[t_5, t_6]$, the power outside right interval boundary t_6 will vanish.

Step3: Revise the modified switching strategy to satisfy the temperature constraints of the cool-room, using the following rule:

- If the predicted internal temperature is outside the temperature constraints $[T_{min}, T_{max}]$, leave the plan status t_p minutes, then join the plan as soon as possible

4 Experimental Results

We have implemented a system with one broker agent, one summarizing agent, one stigspace and a number of loads – cool room agents. The broker agent reads 5-minute pre-dispatch wholesale electricity price from NEMMCO, sets the grid supply cap using a formula which depends on predicted future prices and other factors and places it in stigspace. Each cool room agent calculates a plan of electricity demand for the next half hour which satisfies its internal temperature constraints. This plan is then transferred into mean power demand in 5 minute intervals and put in stigspace. The summarizing agent sums the mean power demand from all agents in each interval for the next half hour planning period, placing the total demand in stigspace. Then cool-room agents then apply the CordCap algorithm to satisfy the system supply cap whilst continuing to adhere to their local constraints as described in section 3. If necessary, this process is iterated as discussed above.

A series of experiments have been completed to investigate system coordination performance, which includes coordination scalability, maximum system demand reduction for a short-period supply cap, the effect of resource agent diversity on coordination performance and continuous coordination benefits for resource agents.

4.1 Room Model

The cool rooms considered have internal temperature constraints with boundaries at 1° and 6° Celsius. The internal temperature is governed by the model developed in [9] with most room features removed for the purposes of these experiments:

$$T_{ai}(t) = \frac{1}{C_a f + k}(C_a f T_{ai}(t-1) + Q_p(t) + kT_{ao}(t)) \tag{1}$$

where T_{ai} is internal temperature, T_{ao} is external air temperature, Q_p is the power capacity of the cool room plant, C_a is the thermal capacity of air in the cool room, f is the sampling rate, $k = A_w U_w + U_v$, A_w is a coefficient, and U_w, U_v are thermal resistances of the wall and ventilation path. We modelled non-identical resource agents by allowing different thermal capacities C_a in the room model. In our experiments, $f = 3000$, $A_g = 1/58.9$, $U_g = 6$, and $U_v = 0$.

4.2 Coordination of Constant Supply Cap

We modelled a set of cool rooms with a constant supply cap. Fig. 3 (left) shows initial room temperatures and power demands in each 5-minute interval for future half hour periods for the system with 3 cool rooms. A constant supply cap, 3kW, is applied to the system. Before coordination, the supply cap is not satisfied for the time 9:25 to 9:40. After several steps of coordination among resource agents, supply cap is satisfied as shown in Fig. 3 (right).

Table 1 lists the minimum supply cap achievable for systems comprising 10, 000 cool rooms with different range of parameters to test coordination performance and scalability. 'S' is system number; 'Capacity' is the power capacity of resource agents, 'T_{on}'/'T_{off}' are turn on/off time constant of resource agents, 'Cap' is the minimum system supply cap which could be satisfied, and 'Stp' is the number of coordination steps for resource agents to satisfy supply cap. Systems 1 to 3 have increasing diversity of resource agents. From the table, we can see that the system with more diverse resource agents will tolerate a smaller supply cap and the coordination is scalable for large numbers of agents.

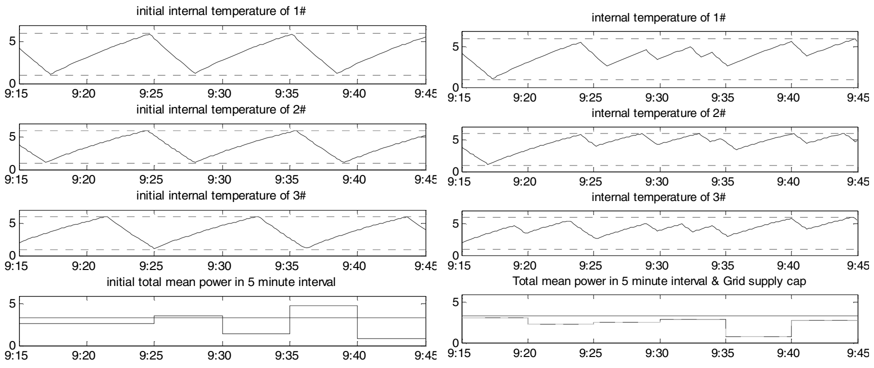


Fig. 3. Room & System States before coordination (left) and after coordination (right)

Table 1. Coordination System with 10,000 Resource Agents

S	Capacity (kW)	T_{on} (min)	T_{off} (min)	Cap (kW)	Stp
1	3	[9.5,17.8]	[8.8,16.7]	10000	6
2	[3,6.76]	[4.83,10.3]	[4.5,34.8]	7100	6
3	[1.5,6.8]	[4.83,37.3]	[4.5, 35]	5000	6

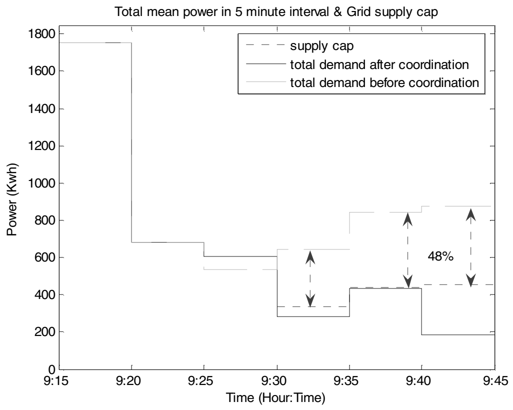


Fig. 4. 15-minute Advance Notice (System 2)

4.3 Demand Reduction for Short-Period Supply Cap with Advance Notice

In a real situation, a supply cap occurs only when the electricity price is high enough. The broker may give advance notice to resource agents before supplying the cap. To test the system response to different cap notice times, the following tests have been executed for a supply cap of 15 minutes' duration.

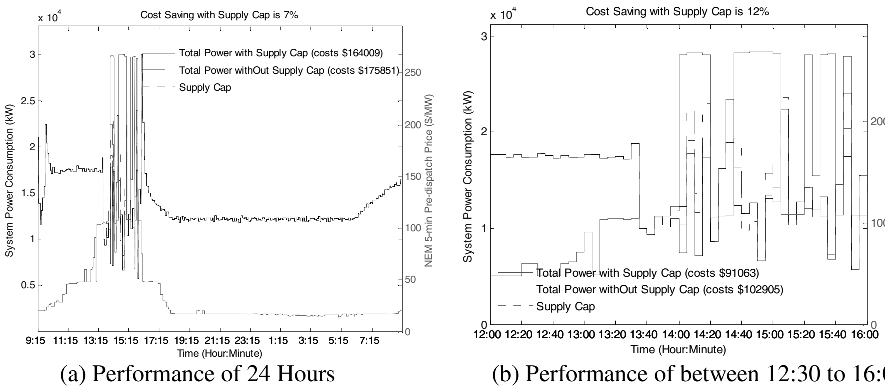
Suppose the cap occurs between 9:30 and 9:45. Advance cap notices of 5, 10 and 15 minutes were investigated. The demand of system 2 for 15 minutes advance notice is shown in Fig. 4, where supply cap, total demand before and after using CordCap algorithm are shown in different style lines respectively. Table 2 gives the percentage reduction in demand that could be achieved for different advance notice times. We can see that the more advance notice time is given, the more demand reduction can be achieved; the greater diversity of resource agents in the system, the more demand reduction can be achieved.

Table 2. Demand Reduction for Different Advance Notice Time

S	Advance Notice	Maximum Demand Reduction	S	Advance Notice	Maximum Demand Reduction
2	5 min.	42.5%	3	5 min.	50%
	10 min.	44%		10 min.	52%
	15 min.	45%		15 min.	52%

Table 3. Continuous Coordination System

Agent No	Total Cost with Cap	Total Cost without Cap	Saving
250	$4,096	$4,376	6%
500	$8,134	$8,710	7%
1000	$16,355	$17,273	5%
10000	$164,009	$175,851	7%



(a) Performance of 24 Hours (b) Performance of between 12:30 to 16:00

Fig. 5. Coordination System with 10,000 Resource Agents

4.4 Demand Reduction for Continuous Coordination System

Demand reductions of continuous coordination for system 2 have also been tested. The continuous coordination is simulated for 24 hours, in this example from 3rd January 2006 at 9:15 am to 4th January 2006 at 9:10 am. The supply cap is set based on the

NEMCCO 5-min pre-dispatch price: when the price is more than $170/Mw the supply cap will be 85% of total demand. Experimental results are shown in Table 3 and Fig. 5.

From the Table 3 we can see that for systems with different numbers of resource agents, from 250 to 10,000 agents, the total cost saving is about the same. This further demonstrates the system scalability. Fig. 5 shows the results for a 10,000-agent system. In this example the NEM electricity price is high from 12:30 to 16:00. The system cost in this period drops from $175851 to $ 164009 as a result of applying the supply cap and achieving a 12% demand reduction during this time.

5 Conclusions

An innovative distributed multi-agent coordination system has been introduced in this paper, which coordinates distributed energy resources to permit a supply cap on the power drawn from the grid. This system features separation of the coordination mechanism from the information exchange mechanism by using indirect (or stigmergetic) communications between resources and a broker. The coordination mechanism is asynchronous and adapts to change in an unsupervised manner, making it intrinsically scalable and robust. This system also features averaging (or more complex processing) of energy consumption plans over each market cycle before such information is communicated. This both reduces the message size and ensures that aggregated quantities of power created by coordination are aligned with the time intervals in which they are valued in the electricity market.

References

1. United States D.O.E. Report of the US Department of Energy's Power Outage Study Team, Final report (March 2002)
2. Guo, Y., Li, J., James, G.: Evolutionary Optimisation of Distributed Electrical Loads and Generators. In: Zhang, S., Jarvis, R. (eds.) AI 2005. LNCS (LNAI), vol. 3809, Springer, Heidelberg (2005)
3. Li, J., Poulton, G., James, G., Zeman, A.: Performance of Multi-agent Coordination of Distributed Energy Resources. WSEAS Transactions on System and Control 2(1), 52–58 (2007)
4. Regan, T., Sinnock, H., Davis, A.: Distributed Energy Neural Network Integration System: Year One Final Report. NREL/SR-560-34216 (June 2003)
5. Dimeas, L., Hatziargyriou, N.D.: Operation of a Multi Agent System for Microgrid Control. IEEE Transactions on Power Systems (November 2004)
6. Akkermans, H., Schreinemakers, J., Kok, K.: Microeconomic Distributed Control: Theory and Application of Multi-Agent Electronic Markets. In: Proceedings, 4th international joint conference on Autonomous Agents and Multiagent Systems (July 2005)
7. Ygge, F.: Market-Oriented Programming and its Application to Power Load Management. Ph.D. Thesis, ISBN 91-628-3055-4, Lund University (1998)
8. See http://www.nemmco.com.au/
9. Hudson, G., Underwood, C.P.: A Simple Building Modelling Procedure for MatLab/Simulin. In: Proceedings, International Building Performance and Simulation Conference, Kyoto (1999)

Adaptation Knowledge from the Case Base

Julie Main, Tharam S. Dillon, and Mary Witten

La Trobe University, Bundoora, Vic 3086, Australia
j.main@latrobe.edu.au, m.witten@latrobe.edu.au

Abstract. Case adaptation continues to be one of the more difficult aspects of case-based reasoning to automate. This paper looks at several techniques for utilising the implicit knowledge contained in a case base for case adaptation in case-based reasoning systems. The most significant of the techniques proposed are a moderately successful data mining technique and a highly successful artificial neural network technique. Their effectiveness was evaluated on a footwear design problem.

1 Case Retrieval and Case Adaptation

Case-based reasoning endeavours to solve a new problem by finding a similar past case, stored in a repository of past cases called a case base [1]. A stored case consists of the problem description and its associated solution. The problem case consists of the problem description alone. If there is an existing case with an identical problem description, its solution maps directly to become the problem case's solution. In the more likely event of a similar but not identical stored case being found, its solution needs to be modified to solve the problem case.

The first step in case-based reasoning is case retrieval [2]; given the new problem to solve, find the closest existing stored case(s). If the problem case and the retrieved case are not identical or within an acceptable level of difference, case adaptation must be carried out to modify the retrieved case's solution to take into consideration the differences between the cases' problem descriptions. The sort of adaptation required and what needs to be altered must be determined, and a strategy for performing this adaptation set in place [3]. Case adaptation continues to be one of the more difficult parts of the case-based reasoning cycle to solve in an automated fashion; that is, without the intervention or extraction of knowledge from a domain expert [4].

To achieve the worthwhile goal of automated adaptation, from where can we obtain knowledge about the relationships between a case's functional specification and its solution? This paper addresses this question. It looks at several techniques for gaining adaptation knowledge from existing cases: a relatively unsuccessful brute force technique, a moderately successful data mining technique and a highly successful technique using supervised artificial neural networks.

Several previous authors have attempted to use knowledge from the case base in adaptation. Some incorporated artificial neural networks [5][6][7][8], but our proposals are considerably different from this previous research.

1.1 Using Stored Case Information in Case Adaptation

One way of discovering knowledge in the case base is from the differences between cases in the case base. We can measure the distance between a retrieved case and a problem case by developing a distance vector of the distances between the related features in each case. For example, consider a case base of two cases, each having five input attributes and three solution components.

– Case 1 input attribute values: 1 0.5 0.8 0 1, component designs: 1A 2B 3C
– Case 2 input attribute values: 1 0.5 0.2 0 0, component designs: 1A 2B 3D

The difference between these cases' input features is the difference vector 0 0 0.6 0 1. It seems that input features 3 and 5 contribute to the design of the third solution component, as when these input features are different, there is a difference in the design of that component.

One technique that utilises difference vectors in case adaptation is, given an input case description and a retrieved case, find another difference vector between cases in the case base that is the same as the difference vector between the problem and retrieved cases. We can then determine the effect of the difference of the cases' problem features on the associated solutions for the cases by looking for differences between the solutions of the pair of cases associated with the matching input vector. Finally we apply the same transformation to the retrieved case's solution to derive the problem case's solution.

Initial proposed technique: Consider a case base of case descriptions with m problem description features and n solution description features. Let case P be a problem case with a problem case description vector $\mathbf{v}_{1p}$ (m components) and case R be a retrieved case with description vector $\mathbf{v}_{2ps}$ ($m + n$ components). If $\Delta\mathbf{v}_{p12}$ is the problem difference vector between case P and R then we seek two cases in the case base with case description vectors $\mathbf{v}_{3ps}$ and $\mathbf{v}_{4ps}$ with the same problem difference vector. We then determine the solution difference vector:

$$\Delta\mathbf{v}_{s34} = (s_{31} - s_{41}, s_{32} - s_{42}, \ldots, s_{3n} - s_{4n})$$

and propose a solution to case P by adding the solution description features of case R to $\Delta\mathbf{v}_{s34}$ giving $\mathbf{v}_{1ps}$:

$$\mathbf{v}_{1ps} = (p_{11}, p_{12}, \ldots, p_{1m}, s_{21} + s_{31} - s_{41},\, s_{22} + s_{32} - s_{42}, \ldots, s_{2n} + s_{3n} - s_{4n}).$$

This initial proposal has certain failings. The sparser the case base, the less likely it is that we will find a pair of cases with an identical difference vector to the retrieved/problem case difference vector. Additionally, the more features a case has, the more combinations of features there may be in the cases in the case base. Therefore, we need to be able to search our case base for pairs of cases with similar, as well as identical, difference vectors. It may be possible to apply a type of nearest neighbour search to find the closest matching case. A similarity metric must therefore be designed for difference vectors.

Using a neural network: Perhaps it is possible to train a single neural network to discover a pair of cases with a similar difference vector? If the network inputs are mapped to a difference vector's input features and each output corresponds to a particular difference vector between a pair of cases, this could theoretically find an appropriate difference vector more quickly than the brute force search technique of nearest neighbour searching. Problems include:

1. The greater the number of cases in the case base, the greater the number of output nodes in the ANN (and hence the lower the expected proportion of 1s to 0s in the output signals). If the proportion of 1s is very low, then by all outputs converging to 0, the nodes in error that should be 1, are a small proportion of the nodes.
2. The network learns with such specificity that it essentially becomes a look-up table for content-addressable memory [9]. Thus new problem cases, unless they have the same set of input features as one of the training cases, will not be able to select among the existing difference vectors successfully.
3. The network takes an excessive amount of time to train.

Alternatives to finding a specific pair of cases with a specific difference vector are needed. Statistical data mining techniques and neural networks provide possibly more generalised ways of using the knowledge from distance vectors.

2 Using Zhou and Dillon's Symmetrical Tau

A data mining technique such as Zhou and Dillon's Symmetrical Tau [10,11] can be applied to learn the relative importance of the relationship between a particular input feature and a particular solution feature. Consider a case base of designs where each case is made up of designs for components M and N, and each case is retrieved by searching for specific values for input features X, Y and Z. The Symmetrical Tau can be used to determine whether X, Y or Z is more important in the selection of a design for component M, and similarly for component N.

This technique requires the calculation of a Tau value for each relationship between an input feature and a component of the design. It could be expected that the higher the Tau value, the more influence an input feature has on that component. For each combination of input feature and solution component, a contingency table is drawn up. For example, for the combination of input feature Y and solution component N, we would have a contingency table as in Table 1. Each example is a difference vector between two cases. The total number of examples to use in the data mining process must be chosen. Using all examples is possible but may not be necessary. Selecting a smaller number of training examples (the most similar pairs or pairs of cases at random) is generally preferable.

Using this contingency table, the Symmetrical Tau value for that relationship between input feature and solution component is defined as [10][11]:

$$\tau = \frac{\sum\limits_{j=1}^{J}\sum\limits_{i=1}^{I}\frac{P_{ij}^2}{P_{+j}} + \sum\limits_{i=1}^{I}\sum\limits_{j=1}^{J}\frac{P_{ij}^2}{P_{i+}} - \sum\limits_{i=1}^{I}P_{i+}^2 - \sum\limits_{j=1}^{J}P_{+j}^2}{2 - \sum\limits_{i=1}^{I}P_{i+}^2 - \sum\limits_{j=1}^{J}P_{+j}^2}$$

Table 1. Contingency table example

		Component N different		
		Yes	No	
Input feature Y different	Yes	A	B	E
	No	C	D	F
		G	H	

A:	no. examples with a different value for Y and a different design for component N
B:	no. examples with a different value for Y and the same design for component N
C:	no. examples with the same value for Y and a different design for component N
D:	no. examples with the same value for Y and the same design for component N
E:	total no. examples with a different value for input feature Y
F:	total no. examples with the same value for input feature Y
G:	total no. examples with a different design for component N
H:	total no. examples with the same design for component N

- $P_{ij}$is the probability the variable is in both row i and column j
- $P_{i+}$is the marginal probability the variable is in row i
- $P_{+j}$is the marginal probability the variable is in column j.

Once a Tau value is found for each combination of input feature and solution component, we can order these combinations. In general, we expect the combinations with higher Tau values to indicate a stronger relationship between input feature and solution component (1 indicates perfect association, 0 indicates no association). For a new problem, which components can stay the same and which must be changed? Given this new problem case and the retrieved solution, we have a difference vector between the two. Where an attribute is different, we can now use these Tau values to determine which solution components must be changed in response to this difference in the input attribute.

We applied this technique to carry out adaptation in the footwear domain [12]. In practical application, it was discovered that the approach generally held true, but there were some Tau values that did not agree with an expert's mappings of input features to solution components (Section 4). Given a listing of input feature and solution component combinations ordered by Tau value, there was typically a group of the very highest Tau values that indicated correlations confirmed by domain experts. The bottom of the listing (those combinations with very low Tau values) were combinations that generally were not indicated by domain experts as correlated. There was, however, a group in the middle where some represented valid correlations, while others did not.

Another drawback of this technique is that each input feature is dealt with on an individual basis. Sometimes second-order features (the combination of several features and the compound change in them) are important in determining the related design solution. Artificial neural networks (ANNs) with a hidden layer that allows the formation of these second-order attributes could be employed.

3 Using a Neural Network to Determine What to Change

A backpropagation network could be used to determine which solution components must be changed in the retrieved case to cater for differences in the problem case. Inputs map to the difference vectors between case input descriptions while the outputs map to whether solution components should be changed or stay the same. Each output corresponds to a different case solution component.

Choosing the cases to use in a network is a problem in itself. Hanney presents the following three strategies for case selection [13]:

1. Cases with less than some threshold number of differences are compared.
2. Similar cases are compared. This requires a similarity metric for determining closeness. If a similarity metric is used in retrieval, it could be used again here.
3. Each case is compared with the n most similar cases in the case base.

A fourth strategy that is similar to the third approach above could be more effective. Taking a random sample of the case base m, retrieve the (large value) k most similar cases to the cases in m and of those $k \times m$ pairs again select a random sample. The reason we choose a second random sample rather than reducing the value of k is to allow some of the training patterns to be further apart. If only the closest cases are used to train the neural network, it could considerably cripple the learning of adaptation knowledge as this assumes that the closest cases alone will give a good coverage of the adaptation ability of the cases. As Hanney discovered, choosing only the n most similar cases learnt the least amount of adaptation rules [13].

Consider again the example in Section 1.1 of a case base with two cases whose input features' difference vector is 0 0 0.6 0 1. The neural network for this case base would therefore have five inputs and one of the training samples could have the inputs 0 0 0.6 0 1 for the difference vector above. Using a back-propagation ANN, we have supervised learning and therefore must specify expected outputs for each training example.

Outputs in this network are related to the solution components of the cases. Each separate component of a case would map to a different output neuron. The inputs are problem description difference vectors and the outputs are solution component difference vectors. So, for each training example, the expected outputs would correspond to which components could stay the same and which must be changed because of the differences in the input features. For the above example, there would be three output nodes, one for whether or not the first component needs to be changed, one for the second component, and one for the third component. For this solution difference vector where the component designs for the first case are 1A 2B and 3C, and the designs for the second case are 1A 2B and 3D, the first two components remain the same and the third one is different. Giving *same components* the value 0 and *different components* the value 1, the expected output for the inputs 0 0 0.6 0 1 would be the solution difference vector 0 0 1.

Having trained these ANNs on a set of difference vectors, retrieval consists of presenting the difference vector between the input values of the current problem and a retrieved case. The output nodes that activate (1) indicate which solution components of the retrieved case would have to be changed, and those outputs that do not activate (0) would be able to be retained from the retrieved case's solution in the current case's solution.

This technique was found to be the most effective in determining what has to be changed in case adaptation. Because of the nature of our application domain, footwear design, components of the design are either retained or changed. Within this domain, a component design cannot be changed 'a bit' as whether it changes significantly or in a minor way, the mould (or pattern) for that solution component has to be remade completely. In domains where the case solution's features can be adjusted in partial ways, the outputs from the neural network could be changed to indicate a degree of change in the solution features.

A subset of randomly selected difference vectors was adequate to train the network to achieve very good results. However, it was found that increasing the number of training examples gave better results. There would seem to be a trade-off between the number of examples used for training and the time taken to train the network.

4 Experimental Results from the Footwear Domain

This paper has described techniques for finding a difference vector similar to that between the current problem case and the retrieved case. An exact matching technique was tried and it rarely found an appropriate difference vector. A nearest neighbour technique was more successful but quite time consuming, as was a backpropagation technique to retrieve a specific difference vector. There were significant difficulties in getting these backpropagation networks to train. In most attempts, all the networks outputs converged to 0.

The automated techniques to learn generalised adaptation information from the difference vectors were considerably more successful than the attempts to find similar difference vectors. Therefore, the results of these techniques applied in the footwear domain are described in detail, after a summary of relevant terms within that domain.

- *upper:* part of the shoe that covers the top of the foot. It may consist of numerous components attached to each other. The main two parts of the upper are the vamp (covers the toes and the front half of the foot) and the quarters (covers the back part of the foot).
- *sole:* part of the shoe that covers the underneath of the foot and makes contact with the ground. It is attached and held onto the foot by the upper.
- *last:* approximately foot-shaped form over which the upper is fitted to give the shoe its shape.

Zhou and Dillon's Symmetrical Tau: On application of Zhou and Dillon's Symmetrical Tau to calculate the degree of association between any input problem feature and any solution component, we expect that the higher the Tau value generated for a particular combination, the greater the association between the input feature and the solution component. A strong association allows us to predict which components can be retained from a retrieved design and which ones must be changed. The Tau example presented here used 50 difference

Table 2. Leg length's contribution to quarters design

		Leg Length different		
		Yes	No	
Qtrs different	Yes	21	11	32
	No	0	18	18
		21	29	

vectors from randomly selected cases in the case base. For each input feature, its association with each of the four solution components was calculated, and then the combinations were ordered by Tau value. The components of the solution in a footwear design correspond to the sole, the last, the quarters and the vamp design.

To calculate a Tau value for a input feature/solution component relationship, a contingency table must be developed. For example, for the input feature leg length's association with the solution component quarters design, the number of difference vectors with a different value for leg length was calculated. The number with a different quarters design component was also calculated. Additionally the number with a different value for leg length and also with a different quarters design component was calculated, giving contingency Table 2.

The probability that a vector belongs to any given row and column P_{ij} is:
$P_{11} = 21/50$, $P_{12} = 11/50$, $P_{21} = 0/50$, $P_{22} = 18/50$

Probabilities that a vector falls into a given row P_{i+} or column P_{+j} is:
$P_{1+} = 32/50$, $P_{2+} = 18/50$, $P_{+1} = 21/50$, $P_{+2} = 29/50$

From the equation for the Tau (τ) value in Section 2, the 1st and 2nd terms of the numerator for this combination are:

$$\sum_{j=1}^{J}\sum_{i=1}^{I}\frac{P_{ij}^2}{P_{+j}} = \frac{P_{11}^2}{P_{+1}} + \frac{P_{21}^2}{P_{+1}} + \frac{P_{12}^2}{P_{+2}} + \frac{P_{22}^2}{P_{+2}} = 0.726896$$

$$\sum_{i=1}^{I}\sum_{j=1}^{J}\frac{P_{ij}^2}{P_{i+}} = \frac{P_{11}^2}{P_{+1}} + \frac{P_{12}^2}{P_{+2}} + \frac{P_{21}^2}{P_{+1}} + \frac{P_{22}^2}{P_{+2}} = 0.71125$$

The 3rd term of the numerator (also the 1st term of the denominator) is:

$$\sum_{i=1}^{I} P_{i+}^2 = P_{1+}^2 + P_{2+}^2 = 0.5392$$

The 4th term of the numerator (also the 2nd term of the denominator) is:

$$\sum_{j=1}^{J} P_{+j}^2 = P_{+1}^2 + P_{+2}^2 = 0.5128$$

Table 3. Ordered Tau Values for footwear input feature's contribution to solution components

Input feature	*Sol^n Component*	*Tau Value*	*Input feature*	*Sol^n Component*	*Tau Value*
OpenToe?	Last	5.21485e-05	DegreeofToeShape	Last	0.0885417
OpenToe?	Quarters	0.000197006	SlingBack?	Quarters	0.0915698
DegreeofToeShape	Vamp	0.00354456	SlingBack?	Vamp	0.0915698
ApronFront?	Quarters	0.00456673	DegreeofCutOut	Last	0.0969382
SlingBack?	Sole	0.00569107	HeelHeight	Vamp	0.107143
ToeSeam?	Quarters	0.00606061	LegLength	Sole	0.125418
OpenHeel?	Last	0.00620404	TypeofFastening	Sole	0.166717
OpenHeel?	Sole	0.0214646	LegLength	Last	0.168116
OpenHeel?	Vamp	0.0234375	ToeShape	Vamp	0.174626
OpenHeel?	Quarters	0.0234375	SoleThickness	Vamp	0.184244
OpenToe?	Sole	0.032882	DegreeofCutOut	Quarters	0.202812
OpenToe?	Vamp	0.0359043	TypeofFastening	Last	0.212608
ApronFront?	Last	0.0409207	HeelHeight	Quarters	0.259259
ToeSeam?	Last	0.0409207	ToeShape	Quarters	0.289406
ApronFront?	Sole	0.0447958	DegreeofCutOut	Vamp	0.289773
ToeSeam?	Sole	0.0447958	ToeShape	Sole	0.316005
LegLength	Vamp	0.0467068	HeelHeight	Last	0.336485
ApronFront?	Vamp	0.048913	ToeShape	Last	0.34593
ToeSeam?	Vamp	0.048913	Wedge?	Sole	0.369748
Wedge?	Quarters	0.0580564	LegLength	Quarters	0.407328
Wedge?	Vamp	0.0580564	SoleThickness	Last	0.46819
DegreeofCutOut	Sole	0.0613909	TypeofFastening	Quarters	0.479167
DegreeofToeShape	Quarters	0.0740741	TypeofFastening	Vamp	0.479167
SlingBack?	Last	0.0766074	SoleThickness	Quarters	0.480405
DegreeofToeShape	Sole	0.0808824	HeelHeight	Sole	0.672354
Wedge?	Last	0.0814294	SoleThickness	Sole	0.82969

Hence the Tau value for the association between leg length and quarters is:

$$\frac{0.726896 + 0.71125 - 0.5392 - 0.5128}{2 - 0.5392 - 0.5128} = 0.407327$$

Comparing this with the Tau for leg length's contribution to the sole component (0.125418), leg length has a significantly greater contribution to the quarters design than to the sole design, as expected. From the ordered Tau values in Table 3 for all the attributes' contributions to the different components, the values over 0.5 show the strongest associations and are the most obvious; that is sole thickness and heel height's contribution to sole design. Values between 0.3 and 0.5 are still quite strong associations while values under 0.3 are rather hit and miss. Here, associations known to be strong in the domain are interspersed with weak associations, perhaps influenced by the number of examples (or lack thereof) displaying the difference in the attribute being evaluated. For example, the smallest value of Tau; namely, open toe's contribution to last design, is very small whereas it should be quite a strong association. The number of difference vectors showing a difference in open toe was only three and by chance, in those three examples, the association is not demonstrated. In general, the higher values are better indicators, but there are still hit and miss values up to 0.3. It may be possible to increase the accuracy by increasing the number of difference vectors used to calculate the Tau values. Increasing the number of examples, however, will not change the lack of second-order features and their influence on the various solution components.

In the footwear domain, there are two values, sole thickness and heel height, that are known by domain experts to describe the main requirements of the sole. While the Tau values show that if either or both of these values change then the sole design must change, these values may also influence last design. If one or the other of these values is different then the last must also be changed. If both change the same amount then the same last may be able to be used as there is an implicit second-order attribute of pitch (that is the difference between the heel height and sole thickness). Without this expert knowledge input, the Tau will not take into account pitch and any other second-order attributes.

ANNs for learning adaptation knowledge: Using a backpropagation network for case adaptation was the most successful approach. Inputs were the values that made up the difference vector between two shoe designs' input features, while the outputs were related to each component of the shoe design.

Several different-sized networks were tried, and it was found that a mid range network was the most effective (a network with 15 hidden layer nodes performed better than networks with 10 or 20 hidden layer nodes). The networks generally trained quite quickly, indicating that the relationships between the inputs and outputs were straightforward to determine. The outputs that were not correct showed associations that were generally not present in the training data.

Another newly constructed test set was then tested on a network trained on the original difference vectors. The set was also tested on another ANN with the same configuration but trained on the original difference vectors combined with the original test data. Increasing the number of training vectors from 60 to 75 allowed the network to learn several more associations, as there were several vectors in the first test set that showed associations that did not exist in the original training set. This addition of training vectors increased accuracy from 91.25% to 96.25%.

4.1 Conclusion

Difficulties were found with exact matching, nearest neighbour matching and ANN matching of the difference vector between the problem case and a retrieved case, with stored difference vectors in the case base.

The chance that the difference vector between a current problem and a retrieved case would exactly match an existing difference vector is small. Using the existing difference vectors (or a subset of them) to acquire generalised adaptation knowledge was significantly more successful.

Using these difference vectors with a data mining technique such as Zhou and Dillon's Symmetrical Tau, it was possible to prioritise the associations between input features and solution components to some degree. There were limitations in that strong real-life associations identified by a domain expert were often prioritised lower if there were not many examples in the training set, and also the approach was unable to find and prioritise second-order features.

A backpropagation network for learning adaptation knowledge, where each output mapped to a design component for reuse or change, was the most successful technique. We found that this approach resulted in a high frequency of correct recommendations for solution components to change or reuse, as verified by a domain expert. There were no apparent drawbacks to this approach and it would seem to be a widely applicable and effective automated technique for case adaptation tasks.

References

1. Main, J., Dillon, T., Shiu, S.: A tutorial on case based reasoning. In: Soft Computing in Case Based Reasoning, pp. 1–28. Springer, Heidelberg (2001)
2. Aamodt, A., Plaza, E.: Case-based reasoning: Foundational issues, methodological variations, and system approaches. AI Communications 7(1), 39–59 (1994)
3. Kolodner, J.: An introduction to case-based reasoning. Artificial Intelligence Review 6(1), 3–34 (1992)
4. Leake, D.: Becoming an expert case-based reasoner: Learning to adapt prior cases. In: Proceedings of the Eighth Annual Florida Artificial Intelligence Research Symposium, pp. 112–116 (1995)
5. Anand, S., Patterson, D., Hughes, J., Bell, D.: Discovering case knowledge using data mining. In: Wu, X., Kotagiri, R., Korb, K.B. (eds.) PAKDD 1998. LNCS, vol. 1394, pp. 25–35. Springer, Heidelberg (1998)
6. Corchado, J., Lees, B., Fyfe, C., Rees, N., Aiken, J.: Neuro-adaptation method for a case-based reasoning system. Computing and Information Systems 5(1), 15–20 (1998)
7. Lofty, E., Mohamed, A.: Applying neural networks in case-based reasoning adaptation for cost assessment of steel buildings. International Journal of Computers & Applications 24, 28–38 (2003)
8. Salem, A.B., Nagaty, K., Bagoury, B.E.: A hybrid case-based adaptation model for thyroid cancer diagnosis. In: ICEIS 2003, vol. 2, pp. 58–65 (2003)
9. Swingler, K.: Applying Neural Networks: A Practical Guide. Academic Press Inc., London (1996)
10. Zhou, X., Dillon, T.: A statistical-heuristic feature selection criterion for decision tree induction. IEEE Transactions on Pattern Analysis and Machine Intelligence 13(8), 834–841 (1991)
11. Sestito, S., Dillon, T.: Automated Knowledge Acquisition. Prentice Hall, Englewood Cliffs (1994)
12. Main, J., Dillon, T.: A hybrid case-based reasoner for footwear design. In: Althoff, K.-D., Bergmann, R., Branting, L.K. (eds.) ICCBR 1999. LNCS (LNAI), vol. 1650, pp. 499–509. Springer, Heidelberg (1999)
13. Hanney, K., Keane, M.: The adaptation knowledge bottleneck: How to ease it by learning from cases. In: Leake, D.B., Plaza, E. (eds.) ICCBR 1997. LNCS, vol. 1266, pp. 359–370. Springer, Heidelberg (1997)

Building Classification Models from Microarray Data with Tree-Based Classification Algorithms

Peter J. Tan, David L. Dowe, and Trevor I. Dix

Clayton School of Information Technology, Monash University, Melbourne, Australia

Abstract. Building classification models plays an important role in DNA mircroarray data analyses. An essential feature of DNA microarray data sets is that the number of input variables (genes) is far greater than the number of samples. As such, most classification schemes employ variable selection or feature selection methods to pre-process DNA microarray data. This paper investigates various aspects of building classification models from microarray data with tree-based classification algorithms by using Partial Least-Squares (PLS) regression as a feature selection method. Experimental results show that the Partial Least-Squares (PLS) regression method is an appropriate feature selection method and tree-based ensemble models are capable of delivering high performance classification models for microarray data.

1 Introduction

DNA microarrays measure a large quantity (often in the thousands or even tens of thousands) of gene expressions of several samples simultaneously. The collected data from DNA microarrays are often called microarray data sets. Advancing statistical methods and machine learning techniques have played important roles in analysing microarray data sets. Results from such analyses have been fruitful and have provided powerful tools for studying the mechanism of gene interaction and regulation for oncological and other studies.

Among much bioinformatics research concerned with microarray data, two areas have been extensively studied. One is to design algorithms to select a small subset of genes most relevant to the target concept among a large number of genes for further scrutinising. Another popular research topic is to construct effective predictors which are capable of producing highly accurate predictions based on diagnosis or prognosis data.

However, due to the nature of the collection of microarray data, a microarray data set usually has a very limited number of samples. In a typical gene expression profile, the number of gene expressions (input variables) is substantially larger than the size of samples. Most standard statistical methods and machine learning algorithms are unable to cope with microarray data because these methods and algorithms require the number of instances in a data set to be larger than the number of input variables. Therefore, many machine learning articles have proposed modified statistical methods and machine learning algorithms tailored to microarray analyses. As such, many proposed classification algorithms for microarray data have adopted various hybrid schemes. In these algorithms, the classification process usually has two steps, which we now outline.

In the first step, the original gene expression data is fed into a dimensionality reduction algorithm, which reduces the number of input variables by either filtering out a larger amount of irrelevant input variables or building a small number of linear or nonlinear combinations from the original set of input variables. The former approach is often known as *variable selection* while the latter is often known as *feature selection*. In the second step, classification models are trained on the data set with a reduced number of input attributes (created in the previous step) using an ordinary supervised classification algorithm.

In principle, many dimensionality reduction algorithms for supervised learning can be applied to the classification of gene expression data. Various two-step schemes have been presented and all of them reported improved classification accuracy. However, in practice, end results are dependent on the combination of the dimensionality reduction algorithm and the classification algorithm. There is no conclusion from previous studies so far which confirms superiority of any particular scheme for microarray data classification.

In this study, we attempt to improve predictive accuracy by building a hybrid classification scheme for microarray data sets. In the first step, we implement two different dimensionality reduction schemes: (i) Partial Least-Squares (PLS) regression [1,2] as the dimensionality reduction algorithm, and (ii) an alternative and novel hybrid feature selection scheme which consecutively applies the discretization method from [3] on the original data sets followed by the PLS regression algorithm. Then in the second step, the two sets of filtered data with new features resulting from the two feature selection schemes described in the first step are separately fed into tree-based classification algorithms. We then use these two schemes (in Tables 3 and 4 respectively) to compare the results from four tree-based classification algorithms - C4.5 [4], AdaBoost [5,6], Random Forests [7] and MML Decision Forests [8].

2 Dimensionality Reduction for Microarray Data

As discussed in the introduction, various dimensionality reduction algorithms have been proposed for the task of dimensionality reduction of microarray data. Fayyad and Irani's discretization method [3] discretises continuous-valued attributes by recursively applying an entropy minimisation heuristic. Tan and Gilbert [9] applied this method to filter out irrelevant genes for classifications. They also compared the single decision tree-based classification algorithm C4.5 with ensemble classification algorithms Bagging and AdaBoost, they concluded that ensemble methods often perform better than a single classification algorithm, especially in classifying gene expression data. Similar claims can also be found in [10,11].

Evolutionary algorithms have also been applied in some classification algorithms for microarray data sets. Jirapech-umpai [12] implemented an evolutionary algorithm proposed by Deutsch [13] for multiclass classification. The study intended to investigate the problem of searching the optimal parameters for the evolutionary algorithm which will generate the optimal number of predictive genes among the initial gene pool. The performances of the algorithm are measured by testing on the leukemia and the NCI60 data sets. They concluded that good results can be achieved by tuning up the parameters within the evolutionary algorithms.

Díaz-Uriarte and Alvarez de Andrés used random forests [7] as the dimensionality reduction algorithm as well as the algorithm for classification of microarray data. The proposed scheme trains random forests iteratively. At each iteration, the number of input variables is reduced by discarding those variables with the smallest variable importance. They showed that their new gene selection procedure selects small sets of genes while maintaining high predictive accuracy.

Statistical methods in multivariate analysis such as Partial Least-Squares (PLS) regression [1,2] and Principal Component Analysis (PCA) have also been adopted for feature selection for microarray data. Nguyen and Rocke [14] conducted a numerical simulation study on the PLS and the PCA methods for microarray-based classification. They concluded that when being applied as the dimensionality reduction method for classification algorithms, PLS out-performs PCA with microarray data.

Although feature selection methods do not explicitly select a subset of genes most relevant to the target concept, attempts have been made to interpret the results of feature selection methods. Roden et al. presented a method [15] for identifying subsets of biologically relevant genes by using a combination of principal component analysis and information-theoretic metrics. Connection between PLS dimensionality reduction and gene selection was examined by Boulesteix [10]. The study found that the order of the absolute values of the coefficients for the first PLS component was identical to the order produced by the classical BSS/WSS ratio gene selection scheme.

3 Related Algorithms

3.1 Principal Component Analysis Regression and Partial Least Squares Regression

Principal Component Analysis (PCA) reduces the dimension of the original data space by projecting original data points to a new coordinate system of the same dimensionality, and then restricting this. The principle components (PC) are orthogonal and calculated by running the nonlinear iterative partial least squares (NIPALS) algorithm, which in turn maximizes the variance on each coordinate sequentially. So the i^{th} PC is given by

$$w_i = \underset{w^T w=1}{\operatorname{argmax}} \, var\{w^T x\},$$

subject to $t_i^T t_j = 0$, where $i \neq j, t_k = w_k^T x$. The idea behind PCA is to discover and retain those characteristics which contribute most to its variance. As such, the dimension of the data set can be reduced by keeping the lower order (small i) PCs while omitting the higher order (large i) PCs.

Partial least squares (PLS) regression aims to reduce the data dimensionality with a similar motivation, but differs from PCA by adopting a different objective function to obtain PLS components. Whereas PCA maximises the variance of each coordinate and whereas both PCA and latent factor analysis will not take into account the values of the target (dependent) attribute, the PLS regression model attempts to find a small number of linear combinations of the original independent variables which maximise the covariance between the dependent variable and the PLS components. (PLS uses the entire data set: input and target attributes.) So the i^{th} PLS component is given by

$$w_i = \underset{w^T w=1}{\operatorname{argmax}} \, cov\{w^T x, y\},$$

subject to $t_i^T t_j = 0$, where $i \neq j, t_k = w_k^T x$.

The PLS method can be illustrated by examining the following relations. Assuming X is an n x m matrix representing a data set of n instances with p independent variables, then if the number of PLS components is K, then the matrix X can be written as the summation of K matrices generated by outer products between vector t_i (which is often known as the score vector) and p_i^T (which is often called the load vector). The optimal number of PLS components, K, is usually determined by applying cross-validation methods on training data. The details of choosing the optimal K for this study can be found in sec. 4.2.

$$X = TP^T + E = \sum_{i=1}^{K} t_i p_i^T + E$$

In effect, the relation in the PLS model projects the data vectors X from the original p-dimensional space into a (much lower than p) K-dimensional space. In the same way, when PLS components are used in the regression, the relation between dependent variable y and PLS component t_i can be written as

$$Y = TBQ + F$$

where T is PLS components matrix, B is the coefficients vector so that TB is orthogonal, Q is the regression coefficients matrix, F is the residual matrix and $\|F\|$ is to be minimised.

Partial least squares regression can be regarded as an extension of the multiple linear regression model. It has the advantage of being more robust, and therefore it provides a good alternative to the traditional multiple linear regression and principal component methods. The original PLS method was proposed by Wold in the late 1960s and initially applied in the field of econometrics. Since then the method had been adopted in other research disciplines and been widely applied in many scientific analyses. SIMPLS [16] is an algorithm for partial least squares regression proposed by de Jong [16]. Compared to conventional nonlinear iterative partial least squares (NIPALS)-PLS, SIMPLS runs faster and is easier to interpret. In SIMPLS, the PLS components are calculated directly as linear combinations of the original variables, which avoids the construction of deflated data matrices. An implementation of SIMPLS by Mevik as an add-on package for the R statistical environment was used in this study.

3.2 MML Oblique Trees and Decision Forests

MML oblique tree [17] is a multivariate decision tree classification algorithm. At internal nodes of MML oblique trees, the data is divided into two mutually exclusive sets by employing a linear discriminant function of input variables. An MML coding scheme encodes such a split, with the margin between the data and the separating hyperplane taken into account. The motivation behind such a scheme is to find a linear discriminant function with the optimal trade-off between fitting the data and simplicity. **Decision forests with MML oblique trees** [8] is an ensemble classification algorithm which at least matches and sometimes surpasses the "right"/"wrong" performance of Breiman's random forests [7]. The optimal candidate trees in decision forests (with overall lower MML coding) with high probabilistic prediction accuracy (low log-loss score) and smaller tree size (lower height with fewer leaf nodes) in MML Decision Forests are selected by the MML oblique trees algorithm. Compared to schemes with univariate trees (which cut on only one attribute at a time), using MML (multivariate) oblique trees offers potential to greatly increase the diversity of the inferred forest. A new weighted tree averaging scheme is also proposed. The scheme is based on Bayesian weighted tree averaging but uses a modified, smoothed prior on decision trees.

3.3 C4.5, AdaBoost and Random Forests

C4.5 [4] is a decision tree inference algorithm introduced by Quinlan. Similar to most decision tree learning algorithms, C4.5 adopts the divide-and-conquer approach to construct decision trees and the procedure is recursive in nature. The C4.5 classification tree algorithm runs fast and it is simple to implement. Therefore, C4.5 trees are often used as base learners in ensemble learning schemes like AdaBoost and random forests.

AdaBoost [6] iteratively re-samples the training set with adapted probabilities (or assigns adapted weights) over instances of the training set. In the end, the scheme gives a weighted average of the results returned by running the classification algorithms on the re-sampling sets. It works very well when the data is noise free and the number of training data is large. But when noise is present in the training sets, or the number of training data is limited, AdaBoost tends not to perform as well as Bagging and random forests.

Random forests [7] uses CART [18] as the base learner and employs several methods to generate a diverse ensemble. Each decision tree in a random forest is trained on a distinct and random data set re-sampled from the original training set, using the same procedure as bagging. While selecting a split at each internal node during the tree growing process, a random set of features is formed by either choosing a subset of input variables or constructing a small group of variables formed by linear combinations of input variables. Random forests [7] have achieved "right"/"wrong" predictive accuracy comparable to that of AdaBoost and much better results on noisy data sets. Breiman also claimed and showed that AdaBoost is a form of random forest (algorithm) [7].

4 Experiments

4.1 Data Sets

In this study, we select seven (mainly oncological) microarray data sets - Leukaemia, Breast cancer, Central nervous system (CNS), Colon tumour, Lung cancer, Prostate cancer and Prostate cancer outcome. All seven microarray data sets have binary output attributes and can be freely downloaded from the Gene Expression Datasets Collection. They have properties that are common in microarray data sets and have also been extensively tested in many previous studies. It makes comparisons with other approaches more convenient. Table 1 shows the summary of the data sets, which can also be found on the web site (http://sdmc.lit.org.sg/GEDatasets) and in [9].

Table 1. Summary of Datasets

Dataset	Number of attributes	Number of PLS components	Number of Instances (Training+Test)	(Binary) Class distribution
Leukaemia	7129	9	72	47:25
Breast cancer	24481	10	97	46:51
Central nervous system	7129	8	60	21:39
Colon tumour	7129	8	62	40:22
Lung cancer	12533	14	181	31:150
Prostate cancer	12600	12	136	77:59
Prostate cancer outcome	12600	5	21	8:13

4.2 Methodology

In this study, the dimensionality reduction scheme is implemented as follows. Each column of the training set is normalised, so that each column has a mean of zero and variance of one. The values of the binary target attribute are set to either 0 or 1. Specifying the number of components for the Partial Least Square Regression, then a PLS model for a training data set is built by feeding the original training set into the SIMPLS algorithm. The output scores of the PLS algorithm are regarded as the values of input variables and forms the training set for the classification algorithms. Similarly, the test sets for the classification algorithm were obtained by feeding each instance of the original test data into the PLS model built from the training set.

Determining the optimal number of PLS components. There is only one free parameter in the PLS algorithms - the number of components, m. There are extensive discussions on how to determine the optimal number of components. However, the goal for performing PLS on the training set in this study is not for regression, rather, the PLS method is applied as a procedure for data pre-processing for the decision tree-based classification algorithms. In our scheme, m is the number of input variables of the data sets to train decision tree and various ensemble learning algorithms.

One major advantage of leave-one-out cross-validation is that it retains the maximum number of data as training sets. As the number of samples in a typical microarray data set is small, we use leave-one-out cross-validations to find the optimal number of components m which will result in classification models with highest "right"/"wrong" predictive accuracies. For each pair of data set and learning algorithm, the PLS methods were repeated with various numbers of PLS components m which ranged from 2 to $4\sqrt{N}$. To reduce the computational cost, the number of PLS components is increased by 2 instead of 1 in each iteration. Then the numbers of PLS components leading to classification models with highest predictive accuracies are regarded as the optimal numbers, as shown in table 2.

Ten-fold cross-validation. For each original data set, 100 pairs of training and test data sets are generated by repeating the 10-fold cross-validation method ten times. Then these 100 pairs of data sets are pre-processed by using procedures described at the beginning of this section. Then for each of 100 pairs of training and test sets which resulted from the above process, classification models were built and tested by using the four classification algorithms (C4.5 [4], AdaBoost [5,6], Random Forests [7] and MML Decision Forests [8]) described at the end of the introduction.

Leave-one-out cross-validation. By selecting one instance from a data set as a test set and using the rest of the data as a training set, N pairs of training and test sets were obtained for a data set with N instances by selecting each data instance only once as a test set. Then for each of the N pairs of training and test set, the experiments were conducted as the procedures described in subsection 4.2 immediately above.

4.3 Results and Discussions

Table 3 shows the classification performances of the four classification algorithms on seven microarray data sets, with the lowest classification errors for each data set highlighted, MML oblique forest achieves the lowest classification error in 5 out of 7 data sets while random forest performs best in the other 2 sets. The MML oblique forests, which ensemble oblique trees with optimal probabilistic prediction performance (see e.g., [17, sec. 3.1][8, sec. 4.2]), return excellent predictive accuracy on noisy data such as microarray data. On the other hand, C5 AdaBoost did not perform well on such noisy data sets. In general, all three

Table 2. The optimal number of PLS components

Dataset	Single C4.5	Random Forest	C5.0 AdaBoost	MML Oblique Forest
Leukaemia	2	8	8	2
Breast cancer	14	10	10	10
Central nervous system	4	6	4	8
Colon tumour	2	4	2	4
Lung cancer	2	2	2	4
Prostate cancer	10	24	12	32
Prostate cancer outcome	18	18	6	6

decision tree-based ensemble classification algorithms achieve higher predictive accuracies than the single model based decision tree algorithm C4.5. For all seven data sets, the best performing ensemble learning algorithms have classification errors 12.7% to 70% (e.g.,$\frac{0.6-2.0}{2.0} = -0.7$) lower than those of C4.5. It clearly indicates that ensemble algorithms are better candidates for building classification models for microarray data sets. Such a conclusion can also be found in [9,11].

Table 3. Predictive error (%) of classification algorithms, using PLS dimensionality reduction scheme ((i) from section 1)

Dataset	Single C4.5	Random Forest	C5.0 AdaBoost	MML Oblique Forest
Leukaemia	5.7	3.8	4.3	**3.3**
Breast cancer	34.8	**28.8**	32.1	30.8
Central nervous system	38.8	35.5	36.8	**34.1**
Colon tumour	19.1	15.3	17.3	**11.2**
Lung cancer	2.0	3.8	1.8	**0.6**
Prostate cancer	17.0	9.4	11.9	**8.7**
Prostate cancer outcome	35.0	**30.7**	48.5	46.8

When applying the PLS method directly on the whole gene set from the original data, our tests returned improved classification accuracies on three (Leukaemia, Lung Cancer and Prostate Cancer) data sets. However, the other tests returned lower “right”/“wrong” classification accuracies on other four data sets than those reported in Tan and Gilbert’s paper [9]. In [9], a subset of the original set of genes was selected by using Fayyad and Irani’s discretization method [3]. For each of the four data sets with worse results in our study, only less than 5% of the original gene set was selected and used to build classification models. For each of the three data sets with improved results in this study, at least 14% of the original gene set were retained. The observation suggests a two-stage dimensionality reduction scheme. In the first stage, irrelevant genes are

Table 4. Predictive error (%) of classification algorithms, using a hybrid dimensionality reduction scheme ((ii) from section 1)

Dataset	Single C4.5	Random Forest	C5.0 AdaBoost	MML Oblique Forest
Leukaemia	3.3	**1.9**	3.6	**1.9**
Breast cancer	21.9	18.4	20.4	**17.9**
Central nervous system	25.8	**21.7**	27.6	23.1
Colon tumour	12.3	15.7	15.8	**11.6**
Lung cancer	1.8	**0.1**	1.8	0.2
Prostate cancer	10.9	7.1	8.6	**6.6**
Prostate cancer outcome	32	**20.3**	30.3	28.1

filtered out by using Fayyad and Irani's discretization method [3]. In the second stage, dimension of the data is further reduced by applying the PLS method on the data with reduced numbers of genes. We processed data on each of seven data sets using the above scheme, then we re-ran the experiments. These experimental results show that, in going from the PLS scheme in Table 3 to the hybrid scheme in Table 4, there are significant across the board increases in classification accuracy. In some data set like the lung cancer data set, the predictive accuracies were extremely high (something like 99.9%).

5 Conclusions and Future Research

We conducted an extensive survey in the area of building classification models from microarray data with various tree-based classification algorithms. Experimental results show that in most cases, tree-based ensemble learning algorithms delivered classification accuracies equivalent to or better than those on the same data sets reported by other studies. Combined with the Partial Least-Squares (PLS) regression method, which is proved to be an appropriate feature selection method, tree-based ensemble learning algorithms are capable of building classification models with high predictive accuracies from microarray data. As the study shows that our hybrid feature selection scheme improves classification accuracies, one question immediately arises: will there be better hybrid schemes for the feature selection process for building tree-based classification models? Since the number of instances in the studied microarray data is small and the performances of many classification algorithms are sensitive to the number of training data, another interesting question is raised: when comparing predictive performances of various classification algorithms on microarray data, what is the impact of adopting different methodologies such as ten-fold cross-validation, leave-one-out cross-validation and bootstrap [19]?

This work was funded by a Monash University Faculty of I.T. Small Grant. We thank Robert Jorissen of the Ludwig Institute for Cancer Research for valuable feedback.

References

1. Geladi, P., Kowalski, B.: Partial least-squares regression: a tutorial. Analytical Chimica Acta 185, 1–17 (1986)
2. Höskuldsson, A.: PLS regression methods. Journal of Chemometrics 2, 211–228 (1988)
3. Fayyad, U.M., Irani, K.B.: Multi-interval discretization of continuous-valued attributes for classification learning. In: IJCAI, pp. 1022–1029 (1993)
4. Quinlan, J.R.: C4.5: Programs for Machine Learning. Morgan Kaufmann, San Francisco (1992), The latest version of C5 is available from http://www.rulequest.com
5. Freund, Y.: Boosting a weak learning algorithm by majority. Information and Computation 121(2), 256–285 (1995)
6. Freund, Y., Schapire, R.E.: Experiments with a new boosting algorithm. In: ICML. International Conference on Machine Learning, pp. 148–156 (1996)
7. Breiman, L.: Random forests. Machine Learning 45(1), 5 (2001)
8. Tan, P.J., Dowe, D.L.: Decision forests with oblique decision trees. In: Gelbukh, A., Reyes-Garcia, C.A. (eds.) MICAI 2006. LNCS (LNAI), vol. 4293, pp. 593–603. Springer, Heidelberg (2006)
9. Tan, A.C., Gilbert, D.: Ensemble machine learning on gene expression data for cancer classification. Applied Bioinformatics 2, S75–S83 (2003)
10. Boulesteix, A.L.: PLS dimension reduction for classification with microarray data. Statistical Applications in Genetics and Molecular Biology 3(1) (2004)
11. Díaz-Uriarte, R., de Andrés, S.A.: Gene selection and classification of microarray data using random forest. BMC Bioinformatics 7, 3 (2006)
12. Jirapech-umpai, T.: Classifying Gene Data Expression using an Evolutionary Algorithm. Master thesis, University of Edinburgh (2004)
13. Deutsch, J.M.: Evolutionary algorithms for finding optimal gene sets in microarray prediction. Bioinformatics 19, 45–52 (2003)
14. Nguyen, D.V., Rocke, D.M.: On partial least squares dimension reduction for microarray-based classification: a simulation study. Computational Statistics & Data Analysis 46(3), 407–425 (2004)
15. Roden, J.C., King, B.W., Trout, D., Mortazavi, A., Wold, B.J., Hart, C.E.: Mining gene expression data by interpreting principal components. Bioinformatics 7, 194 (2006)
16. de Jong, S.: SIMPLS: an alternative approach to partial least squares regression. Chemometrics and Intelligent Laboratory Systems 2(4), 251–263 (1993)
17. Tan, P.J., Dowe, D.L.: MML inference of oblique decision trees. In: Webb, G.I., Yu, X. (eds.) AI 2004. LNCS (LNAI), vol. 3339, pp. 1082–1088. Springer, Heidelberg (2004)
18. Breiman, L., Friedman, J.H., Olshen, R.A., Stone, C.J.: Classification And Regression Trees. Wadsworth & Brooks (1984)
19. Efron, B.: Bootstrap methods: Another look at the jackknife. The Annals of Statistics 7(1), 1–26 (1979)

MAUSA: Using Simulated Annealing for Guide Tree Construction in Multiple Sequence Alignment

P.J. Uren, R.M. Cameron-Jones, and A.H.J. Sale

School of Computing, Faculty of Science, Engineering and Technology,
University of Tasmania, Hobart and Launceston, Tasmania, Australia
{Philip.Uren, Michael.CameronJones, Arthur.Sale}@utas.edu.au

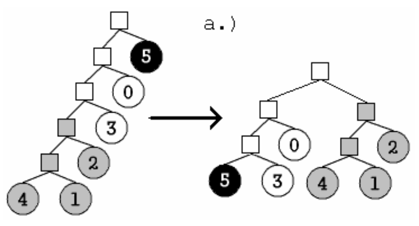
Abstract. Multiple sequence alignment is a crucial technique for many fields of computational biology and remains a difficult task. Combining several different alignment techniques often leads to the best results in practice. Within this paper we present MAUSA (Multiple Alignment Using Simulated Annealing) and show that the conceptually simple approach of simulated annealing, when combined with a recent development in solving the aligning alignments problem, produces results which are competitive and in some cases superior to established methods for sequence alignment. We show that the application of simulated annealing to effective guide tree selection improves the quality of the alignments produced. In addition, we apply a method for the automatic assessment of alignment quality and show that in scenarios where MAUSA is selected as producing the best alignment from a suite of approaches (approximately 10% of test cases), it produces an average 5% ($p = 0.005$, Wilcoxon sign-rank test) improvement in quality.

1 Introduction

Multiple sequence alignment (MSA) is a central technique in bioinformatics. It provides the initial step for procedures as diverse as phylogenic analysis, identification of sequence motifs and higher level structure prediction, amongst others [1]. Most previous work on multiple alignment algorithms has focused on heuristic approaches. Exact methods are, in general, too computationally intense to be practical [2]. An example MSA is presented in Figure 1, part b.

A popular paradigm for multiple alignment involves first generating or assuming a guide tree and then progressively aligning based upon the structure of this tree [3]. For example, adding sequences to an alignment in presentation order corresponds to a maximal depth 'left ladder tree'. Two example trees are shown in Figure 1, part a.

There are drawbacks to the progressive alignment approach. Perhaps the best known of these is the 'once a gap, always a gap' problem. This refers to the greedy nature of the progressive alignment strategy which always picks what appears to be the next best step in adding to the alignment, without considering global optimisation. This step cannot be undone in later alignments. As a result an introduced gap persists even though it may not occur in other progressive alignments.



b.)

```
ACTGATC-GTAACATC
-CTGATCA-TAGCATC
ACAGATCAGT-ACATC
ACTGA--AGTA--ATC
GCTGATC-GGAACATC
AC-GATCAGTAACAT-
```

Fig. 1a. A sample step in the simulated annealing algorithm; the black subtree is swapped with the grey subtree. **b.** A sample multiple sequence alignment.

We present MAUSA (Multiple Alignment Using Simulated Annealing), a program that performs a stochastic search by way of simulated annealing [4] through the space of possible guide trees. This approach has not previously been possible using exact alignment algorithms, such as the Carrillo-Lipman algorithm [5], implemented heuristically in the program MSA [6], due to the time requirements of computing an alignment for each guide tree. By utilising the 'Aligning Alignments Exactly' algorithm (AAE) [7], we can ensure that the alignments produced at internal nodes of the guide tree are exact, while dramatically reducing the time required to achieve this. Thus, MAUSA can explore many guide trees in search of the one which provides the best alignment, as judged by a substitution matrix (e.g. Blosum62), which defines the 'distance' between characters and the sums-of-pairs objective function, which sums the score for each pair of sequences in the alignment. We aim to establish 3 points in this paper. Firstly, that MAUSA's average accuracy is comparable to, or better than the leading alignment programs TCoffee [8], MUSCLE [9] and ClustalW [10]. Secondly, that MAUSA produces enough 'wins' (defined as clearly the best result on a given instance) to make it worth running in conjunction with other approaches. The final point being that the simulated annealing search is warranted and improves results.

Simulated annealing has been used for the purposes of multiple sequence alignment on many occasions. A popular approach is to introduce a set number of gaps into the alignment and use simulated annealing to perturb their position [11, 12]. Gaps at the ends of the alignment act as both a source and sink. Hernandez-Guia et al. [13] also employ an approach which directly applies simulated annealing to the alignment itself. In fact, there are many examples of algorithms designed with such a basis in mind [14] and variants have been applied to similar problems such as local sequence alignment [15]. Work by Koller and Raidl [16] applied an evolutionary algorithm to both the multiple sequence alignment and a graph representation that affects its construction. MAUSA, on the other hand, does not apply a simulated annealing search to the alignment itself, but rather to the guide tree. We rely upon the AAE algorithm to produce high-quality alignments at internal nodes within the tree and apply the simulated annealing search to finding the best tree. In this respect, MAUSA is more closely related to work done by Fleissner et al. [17] on the problem of concurrently producing a phylogenetic tree and a multiple sequence alignment. They employ simulated annealing to (amongst other things) search the space of possible trees. MAUSA however employs a much simpler approach and is not intended as a tool for phylogenetic reconstruction. Although Fleissner et al. provide relatively little data on the success of their approach regards the problem of multiple sequence alignment, what they do show suggests that it performs on-par with ClustalW. We will demonstrate that MAUSA consistently outperforms ClustalW [10] on the BAliBASE reference suite of alignments [18, 19] and is comparable to T-Coffee [8] and MUSCLE [9].

Our purpose is to show that MAUSA is a valuable addition to the current set of tools for multiple sequence alignment. In practice, the use of a number of approaches has been recommended for MSA [9, 20]. MAUSA complements existing approaches, allowing improvements in quality on certain alignments not possible with other tools.

The issue of substitution matrices versus consensus libraries and related techniques (exemplified by ClustalW and T-Coffee respectively) is ignored in this work, although this topic is presented as a particularly promising direction for further study.

Due to the space constraints of the paper we are unable to give the level of explanation for several items that we would wish to. A more detailed discussion of this work can be found via the UTAS ePrints website (http://eprints.utas.edu.au/).

2 Algorithm Description

MAUSA is a progressive multiple alignment algorithm. It constructs an alignment organized around a guide tree. This guide tree is a strict binary tree and initially the internal vertices are empty. Each internal vertex is computed by aligning the alignments that are present at its child vertices. This process continues until the root vertex has been computed; this is the final alignment for the tree.

The process of aligning the alignments at each vertex (where an alignment may consist of one sequence) is performed using AAE [7]. The final alignment is scored based on a substitution matrix using the sum-of-pairs objective function. We refer to this score as the 'cost' of the alignment. It is this cost which is used within the simulated annealing program as a measure of comparison between different trees. In other words, the SA search aims to find the tree which produces the lowest cost alignment.

Due to the fact that alignments within MAUSA are computed using AAE, a cost-based substitution matrix is required. For the purposes of this paper, we explore the use of MAUSA on protein sequences. Hence we compute the Blosum [21] series of cost-based matrices from the standard substitution matrices as defined in Equation 1 [22]. *C(i,j)* is the cost of substituting amino acid *i* with *j* and *S(i,j)* is the score of the same substitution as defined by the standard substitution matrix. We select gap initiation and extension scores following the work of Reese and Pearson [23] and convert them to a cost-based scale by applying Equation 2 and 3 which compute the gap initiation cost *C(γ)* and the gap extension cost C(λ), where '–' is the gap symbol.

$$C(i, j) = -(2 \times S(i, j) - S(i, i) - S(j, j)) \tag{1}$$

$$C(\gamma) = -(2 \times S(\gamma) - S('\text{-}', '\text{-}')) \tag{2}$$

$$C(\lambda) = -(2 \times S(\lambda) - S('\text{-}', '\text{-}')) \tag{3}$$

After scoring an alignment, MAUSA performs a modification to the guide tree (a perturbation, described below). It then repeats the alignment with the new guide tree and compares the cost of this new alignment with that acquired from the old tree. Following the principles of simulated annealing, the new tree is accepted unconditionally if it produces a better alignment than the old tree. If the alignment is worse, the new tree will be accepted with a probability determined by $e^{-dE/T}$ where dE is the change in score and T is the current 'temperature' which is progressively reduced.

The modification to the guide tree is a random swap of two vertices (and the subtree contained below them) with the only restriction being that neither forms a subtree of the other. An example is given in Figure 1, part a. The variation in trees is scaled based on the current 'temperature' – a parameter of the simulated annealing, which is reduced linearly during execution. At high temperature large (sometimes adverse, but exploratory) changes are permitted, but as temperature decreases, the allowed magnitude of changes to the tree is also lowered. To achieve this, we compute an infix traversal string for each tree and use the Levenshtein distance [24] between these two strings to define the magnitude of the change (a 'distance' metric). The simpler measure of total number of leaves in the two subtrees was also tested and although on occasion it outperformed the string based measure, in general it produced poorer results. We explored several other mechanisms for restricting tree modification, but omit their description here. In general, we found the program insensitive to these parameters.

The execution time for MAUSA is based heavily upon that of the AAE algorithm. A single run of MAUSA requires *m(n-1)* executions of AAE, where *n* is the number of sequences in the input and *m* is the number of iterations performed within the SA component of the algorithm. The worst-case runtime of AAE is exponential however using several pruning strategies it is tractable in practice. The details are involved and beyond the scope of this work; the reader is referred to [7] for a thorough explanation. To give an indication of effective execution time, MAUSA completes all instances selected in reference 1 of BALiBASE '99 in less than 1 minute per instance. From reference 2, the minimum time is 1 minute, the maximum is 60 minutes. In reference 3, the minimum time is 30 minutes, the maximum 6 hours. MAUSA was never run for more than 15 hours on any given input used within this research.

3 Experiments and Results

We run MAUSA, TCoffee, ClustalW and MUSCLE on two versions of the standard test suite, BAliBASE - the release from 1999 – BAliBASE '99 [18] and BAliBASE 3.0 [19]. TCoffee has long execution times and is relatively memory intensive. We were unable to execute the full set of alignments from BAliBASE 3.0 for this program. For this reason, we present the results for the first two sets (RV11 and RV12) of version 3.0. In the case of reference 1 from BAliBASE '99, we used only the 'small' subset, but all other reference sets are used in full. We define accuracy for a given input as the percentage of correctly aligned core-residue pairs. Core blocks are defined by the dataset and represent areas of the alignment for which the solution is of high reliability. We use bali_score, a program distributed with both versions of the BALiBASE dataset to calculate these scores.

For our experiments on BAliBASE '99, we wished to compare ClustalW with MAUSA on the same substitution matrices. Hence, we restricted ClustalW to the BLOSUM series (as MAUSA uses only these). For BAliBASE 3.0 however, all programs were run using their default settings. As stated above, within this paper we are concentrating upon objective function optimization and as such, MAUSA makes only very primitive attempts to select appropriate substitution matrices and gap penalties.

To demonstrate the effect of simulated annealing, the results from MAUSA were compared using 3 different search approaches: simulated annealing (SA), random search and no-search. 'No-search' aligns sequences in the order they are presented. Random search is the same as simulated annealing, except *all* modifications are accepted (the search is random, but the best tree found is retained). We also examined using the ClustalW guide tree as a substitute for a search and as a search start point.

In addition, we make use of MUMSA [20] as a practical measure of alignment quality when reference alignments are not available. This is a realistic reflection of a scenario MAUSA would be used in. We attempted to compare MAUSA with Alifritz, an implementation of the work done by Fleissner et al. [17] but found the runtime was prohibitively large on a significant number of alignments within BAliBASE.

All iterative versions of MAUSA (in other words all except no-search) have 20 iterations per temperature and 100 temperature levels. However, the program terminates if no improvement is made within 60 iterations. We also restrict MAUSA to a limited execution time for some experiments. In such cases, the number of iterations is determined by the desired execution time. We never restricted the runtime of any other programs discussed in this paper; they do not produce intermediate solutions.

3.1 The Merit of a SA Search and Choice of an Effective Start Tree

In Table 1 we present the results of running MAUSA with simulated annealing, random search and no-search on BAliBASE ‘99. In slightly more than a quarter of instances within BAliBASE ‘99, the simulated annealing method produces results that are better than either a random search or performing no search at all (using the initial guide tree). Furthermore, in approximately 80% of instances, the simulated annealing search was not inferior to a random search or no search at all. This demonstrates the merit of the simulated annealing search in improving the alignment that would otherwise have been produced. The fact that in several instances either the random search or the simulated annealing was inferior to performing no search indicates a weakness in the objective function used. Since all approaches begin with the same initial guide tree, any measured as better must produce a lower cost alignment as defined by the matrix. However this does not necessarily translate into a more accurate alignment.

We also explored the possibility of starting the search from the guide tree that ClustalW produced. Using the dataset RV11 from BAliBASE 3.0, we observed both a reduction in average accuracy (from 0.47 to 0.44) and winning alignments (from 10 to 8). A win is recorded if the given program produces clearly the best score on an input. A win is not recorded if a program produces the best score on an instance, but that is equalled (to within 0.5%) by any other program. Indeed, only on 3 alignments did this modified version of MAUSA outperform that which started with the ladder tree, whereas on 8 instances the ladder-tree approach performed better. On all other alignments both versions produced the same result. A possible explanation for this is that the ClustalW tree represents a local minimum in the search space, from which the SA has difficulty escaping. More experiments are required to verify this however.

Table 1. Summary of results obtained from running MAUSA with no-search, random search and simulated annealing search on BAliBASE ‘99 (87 Instances). Simulated annealing is better than random with p-value < 0.005 and better than no-search with p-value < 0.0004 (Wilcoxon sign-rank test).

Program	Average	Wins	Draws	Losses
MAUSA SA	0.85	24	45	18
MAUSA random	0.82	5	45	37
MAUSA no-search	0.81	9	31	47

Table 2. Results from RV11 in BAliBASE 3.0 (38 instances) with varied search approaches and start tree. The fourth column shows wins against the other MAUSA approaches presented here, the last column against all other algorithms tested (TCoffee, ClustalW and MUSCLE).

Start Tree	Search Algorithm	Average	Wins (vs. MAUSA)	Wins (vs. all)
Left ladder-tree	SA	0.47	8	10
ClustalW-tree	None	0.41	6	5
ClustalW-tree	SA	0.44	3	8

Further to this, we measured the performance of MAUSA with the ClustalW guide tree where no search was performed on the RV11 dataset from BAliBASE 3.0. In this approach, we also observed a degradation of performance, with a drop in average accuracy from 0.47 to 0.41 and from 10 wins to 5. This also represented a reduction in performance from the version of MAUSA described above, using the ClustalW tree and a simulated annealing search. See Table 2.

Because MAUSA employs a stochastic search algorithm, we also explored the reproducibility of results given multiple trials. MAUSA was run 10 times on BAliBASE '99 (composed of 87 instances) and we observed an average standard deviation of less than 0.5% in the scores for each instance.

We have now shown that SA improves upon the multiple sequence alignment produced by using no search algorithm and that random search is not as effective as simulated annealing. We have also demonstrated that using the ClustalW guide tree to perform the alignment using MAUSA with no search algorithm is outperformed by the simulated annealing search with the same tree as start point (further evidence for the usefulness of the SA). Moreover, both approaches are outperformed by the SA search using the ladder tree as start point. Finally, we have demonstrated that the results obtained by MAUSA are representative and reproducible.

3.2 Comparison of MAUSA with ClustalW, MUSCLE and TCoffee

Table 3 presents the results from running MAUSA, T-Coffee, MUSCLE and ClustalW on BAliBASE ‘99 in terms of the number of wins. We can see from these results that MAUSA produces approximately the same number of wins on BAliBASE ‘99 as TCoffee and MUSCLE. This shows that MAUSA is clearly capable of producing the best result for a given input. We also show in Table 3 the results of the same experiments for BAliBASE 3.0. Once again, the results are not for the whole dataset, but only a subset for which all programs could produce results. MAUSA is comparable to MUSCLE and ClustalW, however TCoffee clearly excels on these inputs. Despite this, it is still obvious that MAUSA is capable of producing the top scoring

Table 3. A comparison of MAUSA, MUSCLE, T-Coffee and ClustalW in terms of wins on BAliBASE '99 (87 instances) and a subset from BAliBASE 3.0 (82 instances)

		BAliBASE '99		
Ref.	MAUSA	MUSCLE	T-Coffee	ClustalW
1	5	6	4	3
2	7	2	8	0
3	3	3	1	0
4	1	3	3	0
5	0	2	1	0
All	16	16	17	3
		BAliBASE 3.0		
Ref.	MAUSA	MUSCLE	T-Coffee	ClustalW
11	9	6	16	4
12	5	6	25	5
All	14	12	41	9

result in a reasonable number of instances. MAUSA was restricted to short run times within these tests.

Table 4 shows the results from all five of the reference sets in BAliBASE '99. MAUSA achieves the highest (or equal highest) result on the first three reference sets. However, performance on reference set 4 is clearly below that of T-Coffee and MUSCLE. To ascertain whether this was due to poor substitution matrix selection the evaluation of reference set 4 was repeated, but with a forced selection of the Blosum80 matrix. This produced an improvement, resulting in an average accuracy of 0.80 (equal to that of T-Coffee and MUSCLE). A similar issue was apparent with the result obtained from reference 5 where, once again, the selection of a more appropriate substitution matrix improves performance to a level which is comparable to T-Coffee and MUSCLE. We note this as evidence of MAUSA's sensitivity to substitution matrix selection. It is not our intention to explore here either mechanisms for selecting substitution matrices or alternatives to their use. We do however note that this is a clear limitation to MAUSA's performance and are addressing it in future work.

The final column of Table 4 shows the average result of each algorithm for the full set of test data. As can be seen, MAUSA, T-Coffee and MUSCLE produce very similar results. It is interesting to note that ClustalW produces relatively poor results when restricted to the Blosum matrices. The poor performance of ClustalW is due in part, no doubt, to the restrictions we have placed upon its matrix selection. However, MAUSA was also operating under a restrictive matrix selection scheme and may have performed better had a more sophisticated approach been used.

Table 4. Summarised results of the accuracy of all four algorithms – MAUSA, T-Coffee, MUSCLE and ClustalW – on the five reference alignment sets provided in BAliBASE. Each number is the average accuracy for all alignments produced by the given algorithm in that set. The variation in average for MAUSA, MUSCLE and T-Coffee is not statistically significant.

Program	Ref. 1	Ref. 2	Ref. 3	Ref. 4	Ref. 5	Average
MAUSA	0.85	0.87	0.82	0.76	0.92	0.84
MUSCLE	0.85	0.86	0.80	0.80	0.96	0.85
T-Coffee	0.82	0.86	0.78	0.80	0.96	0.84
ClustalW	0.80	0.80	0.65	0.68	0.83	0.75

Table 5. As for Table 4, but using a subset of BAliBASE 3.0

Program	Reference 11	Reference 12	Average
MAUSA	0.47	0.83	0.65
MUSCLE	0.47	0.84	0.66
T-Coffee	0.50	0.86	0.68
ClustalW	0.41	0.80	0.61

In a related observation, we observed that T-Coffee produces the most top scoring alignments in reference 2 of BAliBASE ‘99, even though its average (shown in Table 4) is less than that of MAUSA. This is an indication that average performance is not necessarily a direct indication of whether a program will produce the best result on a given input. We would stress once again the importance of using several alignment approaches as no single approach is known to be always best.

We now consider how one would know when to select a MAUSA alignment from a group of results when no reference alignment is available. There exist approaches for the automatic assessment of alignment quality, such as that described by Lassmann and Sonnhammer [20], which consider alignments produced by multiple programs. The authors of the above paper have produced a program (MUMSA) which can be used to rank the results of aligning on a given input by using multiple programs. We applied this to all of the alignments produced on BAliBASE ’99 and the first two reference sets in BAliBASE 3.0 by ClustalW, TCoffee, MUSCLE and MAUSA. For each case, we took the alignment that was predicted as the best by MUMSA. There is an average 5% (p = 0.005, Wilcoxon sign-rank test) increase in accuracy for instances where MAUSA is selected (approximately 10% of instances).

As an example of MAUSA’s ability to produce the top scoring result, we highlight the alignment *1idy* from reference set one of BAliBASE ’99, where MAUSA achieves 0.71 compared to 0.43, 0.32, and 0.29 for TCoffee, MUSCLE and ClustalW.

The results presented for BAliBASE 3.0 were obtained by restricting MAUSA’s runtime. Depending on the size of the input, we allowed a maximum of 15 minutes (for 33 of the alignments), 30 minutes (for 25 alignments), 120 minutes (for 11 alignments), 180 minutes (for 9 alignments) or 240 minutes (for 4 alignments) runtime. In fact, we consider these to be conservatively short execution times, as in practice sequence data often takes tens of person-years to create and as such hours or even days for a multiple sequence alignment is not unreasonable. These results do, however demonstrate MAUSA’s usage in a time-restricted setting. Although MAUSA arguably does worse on BAliBASE 3.0 than ‘99, one can see that it still performs well enough to produce many top scoring alignments (as many as MUSCLE and more than ClustalW) and an average that is comparable to TCoffee and MUSCLE. By way of example, we isolate one instance, BB30003 with 142 sequences of approximately 400 residues each, upon which MAUSA produces a result (0.57 – taking 4 hours) that is better than both MUSCLE (0.53) and ClustalW (0.46) and for which TCoffee is unable to compute a solution.

4 Conclusions and Further Work

In this paper, we have revisited the application of simulated annealing to multiple sequence alignment, incorporated recent alignment developments, and demonstrated both our program's ability to compete with leading approaches and the potential for further research. The program we have developed firstly benefits from its simulated annealing search, and secondly is superior to ClustalW and comparable to both T-Coffee and MUSCLE on the BAliBASE reference suite. Furthermore, we have demonstrated that in addition to being capable of producing average accuracies that are similar to TCoffee and MUSCLE, MAUSA is also able to produce the best result for enough instances to warrant its inclusion in a selection of multiple alignment tools. Finally, we have demonstrated that, with just such a suite of alignment tools and in a situation where the optimal alignment is unknown *a priori*, MAUSA produces a statistically significant improvement in overall quality when it is included.

MAUSA has also demonstrated that, while its execution time is quite high, it can still perform well in a time-restricted setting; our results from BAliBASE 3.0 show this. The runtimes that were used on BAliBASE 3.0 are not largely divergent from those expected with TCoffee and although both programs are unable to complete the full set of alignments, they can both produce wins.

As we have indicated throughout this paper, there are several directions for further development beyond that which we have described here. Most notably, we observed that MAUSA suffers from a reliance on an accurate selection of substitution matrix and gap penalties. Although we provided a simple mechanism for automating this choice, our results indicate that it is insufficient. More sophisticated techniques exist for selecting these parameters, such as are used by ClustalW. Alternatively, the replacement of the substitution matrix as an objective function is also possible. Furthermore, although we tried using the ClustalW tree as a start point for the search, there are other approaches for selecting a start tree that might prove more successful.

The runtime of MAUSA could also be improved by examining mechanisms to determine a more appropriate cut-off point for the search, beyond which little improvement is expected.

Acknowledgments

The authors wish to thank Robert Elliott, Michael Horton and Lloyd Allison for their feedback, suggestions and advice. The anonymous reviewers made many useful suggestions, which we have tried to incorporate. Several, however we could not include because of time or space limitations.

References

1. Notredame, C.: Recent progresses in multiple sequence alignment: a survey: Pharmacogenomics, vol. 3, pp. 131–144 (2002)
2. Wang, L., Jiang, T.: On the complexity of multiple sequence alignment. J. Comput. Biol. 1, 337–348 (1994)
3. Feng, D.F., Doolittle, R.F.: Progressive sequence alignment as a prerequisite to correct phylogenetic trees: J. Mol. Evol. 25, 351–360 (1987)

4. Kirkpatrick, S., Gelatt, C.D., Vecci, M.P.: Optimization by simulated annealing: Science 220, 671–680 (1983)
5. Carrillo, H., Lipman, D.J.: The multiple sequence alignment problem in biology: SIAM journal on applied mathematics 48, 1073–1082 (1988)
6. Lipman, D.J., Altschul, S.F., Kececioglu, J.: A tool for multiple sequence alignment: Proc. Natl. Acad. Sci. 86, 4412–4415 (1989)
7. Kececioglu, J., Starrett, D.: In: RECOMB. Aligning alignments exactly: 8th ACM Conf. on Computational Molecular Biology, pp. 85–96 (2004)
8. Notredame, C., Higgins, D.G., Heringa, J.: T-Coffee: a novel method for fast and accurate multiple sequence alignment: Journal of Molecular Biology 302, 205–217 (2000)
9. Edgar, R.C.: MUSCLE: multiple sequence alignment with high accuracy and high throughput: Nucleic Acids Research 32, 1792–1797 (2004)
10. Thompson, J.D., Higgins, D.G., Gibson, T.J.: CLUSTAL W: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position- specific gap penalties and weight matrix choice: Nucl. Acids Res. 22, 4673–4680 (1994)
11. Kim, J., Pramanik, S., Chung, M.J.: Multiple sequence alignment using simulated annealing: Computer Applications in the Biosciences 10, 419–426 (1994)
12. Ishikawa, M., Toya, T., Hoshida, M., Nitta, K., Ogiwara, A., Kanehisa, M.: Multiple sequence alignment by parallel simulated annealing: Comput. Appl. Biosci. 9, 267–273 (1993)
13. Hernandez-Guia, M., Mulet, R., Rodriguez-Perez, S.: A New Simulated Annealing Algorithm for the Multiple Sequence Alignment Problem: The approach of Polymers in a Random Media: Phys. Rev. E. 72, 031915 (2005)
14. Keith, J.M., Adams, P., Bryant, D., Kroese, D.P., Mitchelson, K.R., Cochran, D.A.E., Lala, G.H.: A simulated annealing algorithm for finding consensus sequences: Bioinformatics 18, 1494–1499 (2002)
15. Frith, M.C., Hansen, U., Spouge, J.L., Weng, Z.: Finding functional sequence elements by multiple local alignment: Nucl. Acids Res. 32, 189–200 (2004)
16. Koller, G., Raidl, G.R.: An evolutionary algorithm for the maximum weight trace formulation of the multiple sequence alignment problem: Parallel Problem Solving from Nature - PPSN VIII, pp. 302–311 (2004)
17. Fleissner, R., Metzler, D., von Haeseler, A.: Simultaneous statistical multiple alignment and phylogeny reconstruction: Systematic Biology 54, 548–561 (2005)
18. Thompson, J.D., Plewniak, F., Poch, O.: BAliBASE: a benchmark alignment database for the evaluation of multiple alignment programs: Bioinformatics 15, 87–88 (1999)
19. Thompson, J.D., Koehl, P., Ripp, R., Poch, O.: BAliBASE 3.0: Latest developments of the multiple sequence alignment benchmark: Proteins: Structure, Function, and Bioinformatics 61, 127–136 (2005)
20. Lassmann, T., Sonnhammer, E.L.L.: Automatic assessment of alignment quality: Nucl. Acids Res. 33, 7120–7128 (2005)
21. Henikoff, S., Henikoff, J.G.: Amino acid substitution matrices from protein blocks: Proc. Natl. Acad. Sci. 89, 10915–10919 (1992)
22. Marsh, T.L., Reich, C.I., Whitelock, R.B., Olsen, G.J.: Transcription factor IID in the Archaea: Sequences in the Thermococcus celer genome would encode a product closely related to the TATA-binding protein of eukaryotes: Proc. Natl. Acad. Sci. USA 91, 4180–4184 (1994)
23. Reese, J.T., Pearson, W.R.: Empirical determination of effective gap penalties for sequence comparison: Bioinformatics 18, 1500–1507 (2002)
24. Levenshtein, V.I.: Binary codes capable of correcting deletions, insertions and reversals: Sov. Phys. Dokl. 6, 707–710 (1966)

A System for Modal and Deontic Defeasible Reasoning

Grigoris Antoniou[1], Nikos Dimaresis[1], and Guido Governatori[2]

[1] Institute of Computer Science, FO.R.T.H., Vassilika Voutwn
P.O. Box 1385, GR 71110, Heraklion, Greece
[2] School of ITEE, The University of Queensland, Australia

1 Introduction

The first source of motivation for our work is the modelling of multi-agent systems. In particular, we follow the approach of [1] that combines two perspectives: (a) a cognitive account of agents that specifies motivational attitudes, using the BDI architecture [2], and (b) modelling of agent societies by means of normative concepts [3].

Commonly, both motivational attitudes and normative aspects are logically captured through the use of modal logics which are, by definition, monotonic. Therefore, they cannot deal properly with inconsistencies, that may easily arise in multi-agent and web environments. As argued in [1], reasoning about intentions and other mental attitudes has *defeasible nature*, and defeasibility is a key aspect for normative reasoning.

In our work, we adopt the well-known defeasible logic [4]. It is a simple, rule-based and computationally cheap approach. The main objective of this paper is to extend defeasible logic with modal and deontic operators, and to report on an implementation.

The second important source for motivation for our work is the *modelling of policies*. Policies play crucial roles in enhancing security, privacy and usability of distributed services and extensive research has been done in this area, including the Semantic Web community. It encompasses the notions of security policies, trust management, action languages and business rules. As for modelling multi-agent systems, our formalism of choice is defeasible reasoning.

The expressive power of formal specification languages required by the business rules community is high and includes deontic notions like obligation, permission and prohibition. Thus, this task is compatible with the first aim to model multi-agent systems. Again, we will rely on deontic extensions of defeasible logic.

The two scenarios outlined above can be combined with the *semantic web* initiative [5], which aims at enhancing the current state of the web through the use of semantic information. Semantic web languages and technologies support the issue of *semantic interoperability*, which is important both for multi-agent systems and for policies. Our language of choice, defeasible logic, is compatible with developments in this area. Now that the layers of metadata (RDF) and ontology (OWL) are stable, an important focus is on rule languages for the semantic web. While initially the focus has been on monotonic rule systems [6,7,8], nonmonotonic rule systems are increasingly gaining attention [9,10,11]. In particular, there are implementations of defeasible logic that interoperate with semantic web standards [10,11].

As already stated, the aim of this paper is to propose modal and deontic extensions of defeasible logic, and to describe the basic characteristics of an implemented system.

We base our implementation on the system DR-Prolog [11], which uses XSB [12] as the underlying logical engine.

2 Defeasible Logic

A *defeasible theory* is a triple $(F, R, >)$, where F is a set of literals (called *facts*), R a finite set of rules, and $>$ a superiority relation on R.

There are three kinds of rules: *Strict rules* are denoted by $A \rightarrow p$, where A is a finite set of literals and p is a literal, and are interpreted in the classical sense: whenever the premises are indisputable (e.g. facts) then so is the conclusion. Defeasible rules are denoted by $A \Rightarrow p$, and can be defeated by contrary evidence. *Defeaters* are denoted by $A \rightsquigarrow p$ and are used to prevent some conclusions. In other words, they are used to defeat some defeasible rules by producing evidence to the contrary.

A superiority relation is an acyclic relation $>$ on R (that is, the transitive closure of $>$ is irreflexive).

A formal definition of the proof theory is found in [4].

3 Extension of Defeasible Logic with Modalities

Recent work [1] shows that Defeasible Logic is a nonmonotonic approach that can be extended with modal and deontic operators. This paper presents a computationally oriented nonmonotonic logical framework that deals with modalities. It combines two independent perspectives about cognitive agents, belief-desire-intention(BDI) agent architecture and agent models that are based on social and normative concepts. This approach has many similarities with the Beliefs-Obligations-Intentions-Desires architecture (BOID) [3]. In BOID conflicts may arise among informational and motivational attitudes. The way these conflicts are resolved determines the type of the agent.

The logical framework deals with the following modalities:

1. *knowledge* - the agent's theory about the world
2. *intention* - policy-based intentions, that is the agent's general policies
3. *agency* - agent's intentional actions
4. *obligation* - obligations from the agent's normative system

We consider an additional deontic modality, permission. This component represents what an agent is permitted to do, according to his normative system. We extend defeasible logic with the permission operator, in order to represent and reason with business rules and policies properly, in Semantic Web applications.

Defeasible Logic is the suitable formalism that can deal with these components. The reason being ease of implementation, flexibility and it is efficient. A rule-based nonmonotonic formalism was developed that extends defeasible logic and represents and reasons with these modal operators. It has as main feature the introduction of the mode for every rule, which determines the modality of rule's conclusion. It also supports modalised literals that can be defined in defeasible theories as facts or as part of the antecedents of rules.

4 Translation into Logic Programs

We use the approach proposed in [13], [14], to perform reasoning over a defeasible theory. According to this, we translate a defeasible theory D into a logic program P(D), and we use a logic metaprogram that simulates the proof theory of the formalism that extents defeasible logic, to reason over the defeasible theory. In the following we describe a sample of the metaclauses used.

Cclauses for definite provability: a literal is strictly (or definitely) provable in knowledge, if it is a fact and strictly provable in other modalities, if the corresponding modal literal is a fact. A modalised literal is represented as prefixed with a modal operator (agency, intention, obligation or permission). An unmodalised literal belongs to the knowledge of the environment. A literal is also strictly provable in a modality, if it is supported by a strict rule, with the same mode and the premises of which are strictly provable. A definite provable literal in intention is defined as

```
strictly(P,intention):-fact(intention(P)).
```

The next clauses define defeasible provability: a literal is defeasibly provable in a modality, either if it is strictly provable in the same modality, or if the literal, for this modality, is consistent, it is supported by a rule, and there is not an undefeated applicable conflicting rule.

```
defeasibly(P,Operator):-strictly(P,Operator).
defeasibly(P,Operator):-consistent(P,Operator),supported(R,Operator,P),
negation(P,P1),not(undefeated_applicable(S,Operator,P1)).
```

A literal is consistent in a modality, if its complementary literal is not strictly provable in the same modality and in any of the attacking modalities. For example, in a strongly independent agent, a literal is consistent in intention, if its complementary is not strictly provable in intention, knowledge and agency:

```
consistent(P,intention):-negation(P,P1),not(strictly(P1,knowledge)),
not(strictly(P1,intention)),not(strictly(P1,agency)).
```

A literal is supported in a modality, if it is supported by a supportive rule with the same mode, the premises of which are defeasibly provable.

```
supported(R,Operator,P):-supportive_rule(R,Operator,P,A),defeasibly(A).
```

A rule is undefeated applicable in a modality, if it is a supportive rule or a defeater in a mode that attacks the modality, the premises of the rule are defeasibly provable, and it is not defeated neither by a supported literal in the modality nor by an applicable rule for the rule 's mode. For example, in a strongly independent type, a rule is undefeated applicable in agency, if it is a rule in knowledge which is not defeated neither by a supported literal in agency nor by an applicable rule in knowledge.

```
undefeated_applicable(S,agency,P):-rule(S,knowledge,P,A),defeasibly(A),
not(defeated_by_supported(S,agency,P)),not(defeated_by_applicable
(S,knowledge,P)).
```

A literal, supported by a rule R, is defeated by a supported literal in a modality, if the latter is complementary, it is supported in the same modality by a rule S and S is superior to R.

```
defeated_by_supported(R,X,P):-negation(P,P1),supported(S,X,P1),
superior(S,R).
```

A literal, supported by a rule R, is defeated by an applicable rule in a modality, if this rule is conflicting to R, it is applicable in the same modality and superior to R.

```
defeated_by_applicable(R,X,P):-negation(P,P1),applicable(S,X,P1),
superior(S,R).
```

A rule is applicable in a modality, if it is a supportive rule or a defeater for an attacking modality and its premises are defeasibly provable; it is also applicable even it has a mode that can be converted to an attacking modality with the feature of rule conversion. For example, a rule is applicable in knowledge, if it is a rule in agency, the premises of which are defeasibly provable:

```
applicable(R,knowledge,P):-rule(R,agency,P,A),defeasibly(A).
```

5 Implementation

Our nonmonotonic rule-based system supports reasoning in defeasible logic, extended with the modalities. It provides automated decision support, when running a specific case with the given logic programs and ontological knowledge to get a correct answer.

An additional functionality that the system supports, is that it has the ability to treat RDF data as facts of the user's defeasible theories, in order to be processed by the organization's rules. The RDF/S documents are retrieved from the Web, and validated by the Semantic & Syntactic Validator, before being loaded to the system. This validator is an embedded version of VRP parser [15]. The system employs the SWI RDF parser to load the valid RDF/S data and translate them into RDF triples. The triples are then translated into Prolog facts and rules, which are passed to the Reasoning Engine.

The Reasoning Engine compiles the metaprogram, corresponding to the agent type we use, and the logic programs, representing the rules and contain the ontological knowledge. Then it evaluates the answers to user's queries expressed in Prolog syntax and applied to the compiled programs. The Reasoning Engine was implemented by using the Java programming library of InterProlog [16]. This is an open-source Java front-end that supports the Prolog engines of YAP [17], SWI and XSB. It provides access to Prolog engines over TCP/IP sockets and launches Prolog processes in the background, outside the Java Virtual Machine. Thus InterProlog provides the interface to pass logic programs and process queries to YAP and XSB, the engines that make the reasoning, and RDF/S data to SWI, which translates them in logic programs.

Finally the system provides a Graphical User Interface based on Java Foundation Classes (Swing). By interacting with the GUI, the user can import logic programs, load RDF/S ontologies and query the system.

6 Conclusion

We argued that defeasible reasoning is a computationally efficient way of dealing with issues related to the modelling of policies and multi-agent systems. We have described how to enhance standard defeasible logic with agency, intention, permission and obligation operators, and briefly outlined an implemented system that is also compatible with semantic web technologies.

In future work, we intend to provide an experimental evaluation, experiment with other logical underlying engine, and develop realistic applications.

References

1. Governatori, G., Rotolo, A.: Defeasible Logic: Agency, Intention and Obligation. In: Lomuscio, A.R., Nute, D. (eds.) DEON 2004. LNCS (LNAI), vol. 3065, pp. 114–128. Springer, Heidelberg (2004)
2. Rao, A.S., Georgeff, M.P.: Modeling rational agents within a BDI-architecture. In: Allen, J., Fikes, R., Sandewall, E. (eds.) KR 1991. Proceedings of the 2nd International Conference on Principles of Knowledge Representation and Reasoning, pp. 473–484. Morgan Kaufmann publishers Inc, San Francisco (1991)
3. Broersen, J., Dastani, M., Hulstijn, J., Huang, Z., van der Torre, L.: The BOID architecture: conflicts between beliefs, obligations, intentions and desires. In: AGENTS 2001. Proceedings of the fifth international conference on Autonomous agents, pp. 9–16. ACM Press, New York (2001)
4. Antoniou, G., Billington, D., Governatori, G., Maher, M.J.: Representation results for defeasible logic. ACM Trans. Comput. Logic 2(2), 255–287 (2001)
5. Berners-Lee, T., Hendler, J., Lassila, O.: The Semantic Web. Scientific American 284(5), 34–44 (2001)
6. Grosof, B.N., Horrocks, I., Volz, R., Decker, S.: Description logic programs: combining logic programs with description logic. In: WWW, pp. 48–57 (2003)
7. Horrocks, I., Patel-Schneider, P.F.: A proposal for an OWL Rules Language. In: WWW 2004. Proceedings of the 13th international conference on World Wide Web, pp. 723–731. ACM Press, New York (2004)
8. Rosati, R.: On the decidability and complexity of integrating ontologies and rules. WSJ 3(1), 41–60 (2005)
9. Eiter, T., Ianni, G., Schindlauer, R., Tompits, H.: dlvhex: A System for Integrating Multiple Semantics in an Answer-Set Programming Framework. In: WLP, pp. 206–210 (2006)
10. Bassiliades, N., Antoniou, G., Vlahavas, I.P.: DR-DEVICE: A Defeasible Logic System for the Semantic Web. In: Ohlbach, H.J., Schaffert, S. (eds.) PPSWR 2004. LNCS, vol. 3208, pp. 134–148. Springer, Heidelberg (2004)
11. Antoniou, G., Bikakis, A.: DR-Prolog: A System for Defeasible Reasoning with Rules and Ontologies on the Semantic Web. IEEE Transactions on Knowledge and Data Engineering 19(2), 233–245 (2007)
12. XSB - Logic Programming and Deductive Database System for Unix and Windows. `http://xsb.sourceforge.net` (2007)
13. Antoniou, G., Billington, D., Governatori, G., Maher, M.J.: Embedding defeasible logic into logic programming. Theory Pract. Log. Program. 6(6), 703–735 (2006)
14. Maher, M.J., Rock, A., Antoniou, G., Billington, D., Miller, T.: Efficient defeasible reasoning systems. International Journal on Artificial Intelligence Tools 10(4), 483–501 (2001)
15. VRP - The ICS-FORTH Validating Rdf Parser. `http://139.91.183.30:9090/RDF/VRP` (2007)
16. InterProlog - a Prolog-Java interface. `http://www.declarativa.com/interprolog` (2007)
17. YAP Prolog. `http://www.ncc.up.pt/vsc/Yap` (2007)

Time-Reversal in Conway's *Life* as SAT

Stuart Bain

Institute of Transport and Logistics Studies
Faculty of Economics and Business
University of Sydney, Sydney, NSW 2006, Australia
`stuartb@itls.usyd.edu.au`

Abstract. This paper describes a translation of the time-reversal problem in *Life* to propositional satisfiability. Two useful features of this translation are: that the encoding is linear (in both variables and clauses) with respect to the number of cells in the original problem; and, it can be used to generate problem instances that are known *a priori* to be satisfiable. The problem is shown to have statistically defined hard regions where instances are on average more difficult than in other regions.

1 Introduction

Conway's *Life* [1] is perhaps the most widely known of all cellular automata. Whilst determining the successor of a *Life* pattern is a polynomial-time procedure, determining the precursor of a pattern is significantly more challenging.

The development of SAT solvers suitable for 'real-world' problems benefits from ready access to structured problem instances. As genuine instances are often in short supply, may be cumbersome to distribute, or subject to commercial priviledge, structured problem generators therefore serve an important function in solver development and testing. A useful property of a candidate problem is whether it can be used to generate instances that are known to be satisfiable in advance. This will be shown to be easily enforceable in *Life* time-reversal.

The focus of this paper is that the *Life* time-reversal problem can be used to generate challenging, structured, propositional satisfiability instances for use in the development and testing of SAT solvers.

2 Problem Description

Life operates using an infinite square grid of cells, where at each time-step, each cell is either alive or dead. The state of a cell depends exclusively on the state of itself and its neighbours in the immediately preceding time-step. Each cell has exactly 8 neighbours, being the surrounding cells in the horizontal, vertical and diagonal directions. The rules in *Life* may be described succintly as follows: A cell will be dead in the following time-step unless it currently has either exactly 3 live neighbours or exactly 2 live neighbours and is presently alive itself. Time-reversal in *Life* can be therefore be specified as a decision problem as follows:

Does the specified pattern of cells possess at least one precursor?

3 Problem Encoding

Although *Life* is generally considered to operate using an infinitely large grid of cells, it is possible to restrict the domain of the problem to a finite grid. Given the specification of the state of the grid cells $(1, 1)$-(x, y) (the inner cells) at time t_1, it is possible to consider their time-reversal by considering not an infinite grid, but only the cells $(0, 0)$-$(x + 1, y + 1)$ (the outer cells). As cells in *Life* can exist in only two possible states, SAT is a natural method for encoding such a problem, since the state of each cell can be represented by a single SAT variable.

The following notation is used to specify the clauses of the SAT encoding. $\bigwedge$ and $\bigvee$ are non-binary, set versions of the conjunction and disjunction operators respectively. Sets are written in bold face. Each rule defines a propositional formula (in conjunctive normal form), for a specified cell x and its corresponding neighbour set $\mathbf{n}$. $\mathcal{P}_i(\mathbf{A})$ is defined to be a variation of the power set axiom, which returns all subsets of $\mathbf{A}$ having a cardinality of i. Overlining denotes a literal or set of literals that is negated in the formula. All literals refer to the state of a cell at time t_0. The overall SAT formula is formed by the conjunction of clauses from rules 1-3 for cells live at t_1, and rules 4-5 for cells dead at t_1.

(1) Loneliness: A cell with fewer than 2 live neighbours (at least 7 dead neighbours) at time t_0 is dead at time t_1, irrespective of its own state at t_0. *(2) Stagnation*: A dead cell with exactly two live neighbours at time t_0 will still be dead at time t_1. *(3) Overcrowding*: A cell with four or more live neighbours at time t_0 will be dead at time t_1 irrespective of its own state at t_0. *(4) Preservation*: A cell that is alive at time t_0 with exactly two live neighbours will remain alive at time t_1. *(5) Life*: A cell with exactly 3 live neighbours at time t_0 will be alive at time t_1, irrespective of its prior state. Formally:

$$\mathit{Loneliness}\,(x, \mathbf{n}) = \bigwedge_{\mathbf{c} \in \mathcal{P}_7(\mathbf{n})} \left(\bigvee \mathbf{c} \right) \tag{1}$$

$$\mathit{Stagnation}\,(x, \mathbf{n}) = \bigwedge_{\mathbf{c} \in \mathcal{P}_2(\mathbf{n})} \left(x \vee \bigvee \overline{\mathbf{c}} \vee \bigvee (\mathbf{n} - \mathbf{c}) \right) \tag{2}$$

$$\mathit{Overcrowding}\,(x, \mathbf{n}) = \bigwedge_{\mathbf{c} \in \mathcal{P}_4(\mathbf{n})} \left(\bigvee \overline{\mathbf{c}} \right) \tag{3}$$

$$\mathit{Preservation}\,(x, \mathbf{n}) = \bigwedge_{\mathbf{c} \in \mathcal{P}_2(\mathbf{n})} \left(\overline{x} \vee \bigvee \overline{\mathbf{c}} \vee \bigvee (\mathbf{n} - \mathbf{c}) \right) \tag{4}$$

$$\mathit{Life}\,(x, \mathbf{n}) = \bigwedge_{\mathbf{c} \in \mathcal{P}_3(\mathbf{n})} \left(\bigvee \overline{\mathbf{c}} \vee \bigvee (\mathbf{n} - \mathbf{c}) \right) \tag{5}$$

4 Instance Generation and Evaluation

Two methods of generating random instances of the $Life$ time-reversal problem were examined, both of which generate a mix of satisfiable and unsatisfiable instances.

The first method involves choosing which cells are live at time t_1 entirely at random. The second method distributes live cells in a balanced way, by making cells live in a regular order and then randomly permuting the rows and columns of the grid to create different instances. This method has also been used to create balanced instances of the quasigroup with holes problem [2].

Problem sizes (inner grid dimensions) ranged from 15 through 25, using a square grid so that $x = y$. The percentage of live cells was varied from 4% to 96% in steps of 4%. 100 different instances were created for each parameter combination. Run-times are for a Sun UltraSPARC-III 900MHz computer.

4.1 Basic Results

The difficulty of the SAT encoding of both the random and balanced instances was empirically examined using two different complete solvers: MiniSAT version 1.14.1 [3] and zChaff version 2004.5.13 [4]. The run-time profiles for each solver were similar (so only the results for the more efficient MiniSAT are presented).

Plots of the mean run-time on instances of both types are shown in Fig. 1. The median run-times do not differ substantially from the mean times so are omitted here, but will be presented for the balanced case in the following section.

These figures show that there are two distinct 'hard' regions, located respectively at 48% and 88% live cells. Whilst obvious in the larger instances, smaller instances do not tend to exhibit these peaks; their peak occurs in the valley between the two peaks instead. Also of interest is that the location of these regions does not depend greatly on the method of assignment of live cells (balanced or random).

What explanation can be given for the observed problem difficulty? In other SAT problems, hardness peaks are often associated with a phase transition

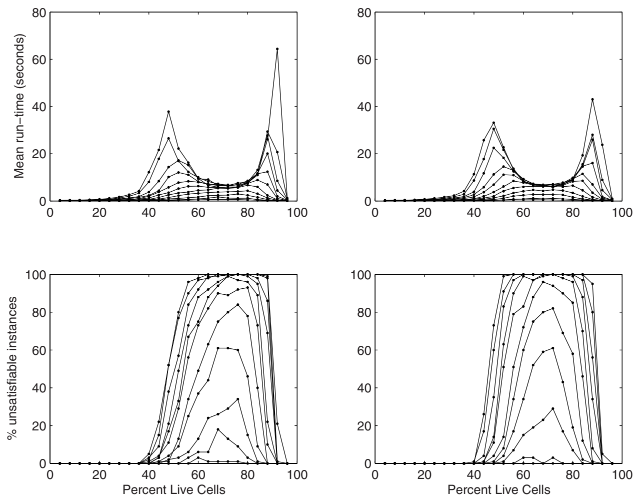


Fig. 1. Average time required to determine the satisfiability of problem instances and percent unsatisfiable instances, by number of live cells

between satisfiable and unsatisfiable instances [5]. The percentage of unsatisfiable instances by percent live cells is also shown in Fig. 1. These figure shows that the hardness peaks in the problem do coincide with the transition from mostly-SAT to mostly-UNSAT instances and *vice versa*. The observed hardness peaks occur approximately at the points where only 50% of generated instances are satisfiable, analogous to results for other SAT-encoded problems [2].

4.2 Known Satisfiable Instances

It is possible to generate instances of this problem that are guaranteed to be satisfiable in advance. This is achieved by generating an $(x-2)$-by-$(y-2)$ pattern (as above) and applying the rules of *Life* in the forward-time direction to determine the state of the x-by-y grid at time t_1. Unlike the instances generated previously, in this case outer cells and any cells external to these are explicitly declared to be off at time t_0 (achieved by adding one literal to represent all external cells and a single unit clause with this literal present negatively).

Instances were generated using the balanced method. It can be seen from Fig. 2 that the previous right-most hardness peak coincident with the UNSAT-SAT transition has been eliminated. But the peak on the left remains, albeit shifted to now exhibit greatest difficulty when between 36% and 40% of cells are originally live, depending on whether mean or median times are considered.

Of particular interest is that the difficulty of the known SAT instances is comparable to that of the mixed SAT/UNSAT instances (at each respective peak). It is reasonable to conclude then, that enforcing satisfiability in this problem does not lead to trivially solved instances as occurs with some problems [6].

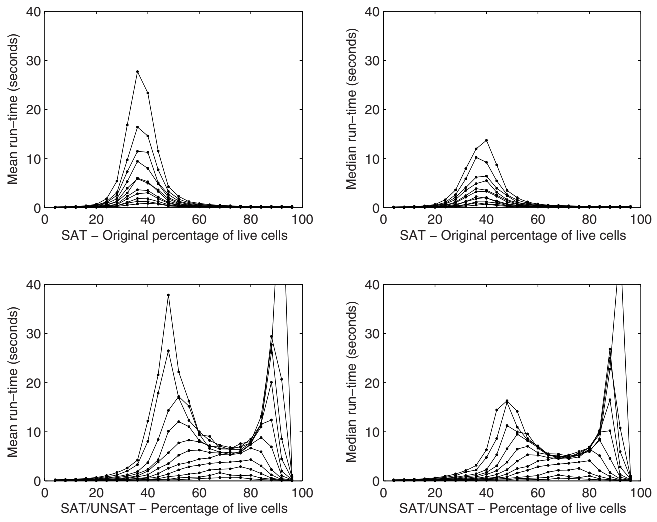


Fig. 2. Comparison of mean/median run-times on SAT vs SAT/UNSAT instances

The run-time distributions for the hardest two parameter settings (those with 36% and 40% of cells originally live) were compared against those of all other settings using the Wilcoxon Rank-Sum test. With the exception of their comparison to each other, the Wilcoxon test confirmed that these two run-time distributions were harder than all other distributions: with strong significance ($\alpha < 0.01$) in all cases for the 36% distribution; and similarly for the 40% distribution but with the single exception of the comparison to the 32% distribution (which was simply significant, $\alpha < 0.05$).

5 Conclusions and Future Work

This paper has presented a method for translating the time-reversal problem of *Life* into SAT. This is a useful structured problem to consider for satisfiability testing, particularly as it can be used to generate known satisfiable instances.

Distributions with both SAT/UNSAT and only SAT instances were examined. In the former case, the most difficult instances were found to be coincident with the phase-transitions from mostly-SAT to mostly-UNSAT instances.

In the latter case however, a hard region still occurs, unrelated to any phase transition phenomena. Perhaps the most pressing questions then are the underlying reasons for the greater difficulty of instances in the hard region of the known SAT instances. Having discounted a phase-transition, a study of backbone size and number of solutions may offer some insight as to their greater difficulty.

The non-parametric Wilicoxon Rank-Sum test was used to confirm that the empirically identified hard distributions were on average more difficult than competing distributions with strong statistical certainty. Since this problem has been shown to have defined hard regions, and the locations of these regions identified, it presents an ideal problem for the testing of SAT solvers.

An extended version of this paper and the instance generator are available from the author's homepage at http://stuart.multics.org

References

1. Gardner, M.: Mathematical Games: The fantastic combinations of John Conway's new solitaire game Life. Scientific American 223, 120–123 (1970)
2. Kautz, H., Ruan, Y., Achlioptas, D., Gomes, C., Selman, B., Stickel, M.: Balance and filtering in structured satisfiable problems. In: IJCAI 2001, pp. 351–358 (2001)
3. Eén, N., Sörensen, N.: An extensible SAT solver. In: Giunchiglia, E., Tacchella, A. (eds.) SAT 2003. LNCS, vol. 2919, pp. 502–518. Springer, Heidelberg (2004)
4. Moskewicz, M.W., Madigan, C.F., Zhao, Y., Zhang, L., Malik, S.: Chaff: Engineering an efficient SAT solver. In: Proc. of the 38th Design Automation Conference (2001)
5. Achlioptas, D., Naor, A., Peres, Y.: Rigorous location of phase transitions in hard optimization problems. Nature 435, 759–764 (2005)
6. Achlioptas, D., Gomes, C.P., Kautz, H.A., Selman, B.: Generating satisfiable problem instances. In: AAAI 2000, pp. 256–261 (2000)

A Knowledge-Based Approach to Named Entity Disambiguation in News Articles

Hien T. Nguyen[1] and Tru H. Cao[2]

[1] Ho Chi Minh City University of Industry, Vietnam
[2] Ho Chi Minh City University of Technology, Vietnam
{hiennt, tru}@cse.hcmut.edu.vn

Abstract. Named entity disambiguation has been one of the main challenges to research in Information Extraction and development of Semantic Web. Therefore, it has attracted much research effort, with various methods introduced for different domains, scopes, and purposes. In this paper, we propose a new approach that is not limited to some entity classes and does not require well-structured texts. The novelty is that it exploits relations between co-occurring entities in a text as defined in a knowledge base for disambiguation. Combined with class weighting and coreference resolution, our knowledge-based method outperforms KIM system in this problem. Implemented algorithms and conducted experiments for the method are presented and discussed.

Keywords: Name disambiguation, coreference resolution, ontology, knowledge base, semantic web.

1 Introduction

In Information Extraction and Natural Language Processing areas, named entities (NE) are people, organizations, locations, and others that are referred to by names ([3]). Having been raised from research in those areas, named entities have also become a key issue in development of Semantic Web ([11]). That is because, in many domains, in particular news articles, the information and semantics of the articles' texts center around the named entities and their relations mentioned therein.

One great challenge in dealing with named entities is that one entity may have different names and one name may refer to different entities. The former raises the NE coreference problem, while the latter the NE ambiguity problem, which this paper addresses. For example, the name "*Jim Clark*" in one news article may refer to *Jim Clark* who is a *Formula One* world champion, whereas in another news article it may refer to *Jim Clark* who is the founder of *Netscape*.

Such identity uncertainty ([9]) has attracted much research effort and been tackled in various domains, with different scopes, and purposes, e.g. name disambiguation ([1], [4], [6], [8]), citation matching ([9]), and entity identification ([5], [7], [10]). Name disambiguation is to make clear if two identical names refer to the same entity or not. Several methods, using vector space model or graph model, for instance, have been proposed for name disambiguation of people in social network ([1]), person names in web search ([4]), authors in publications ([6]), and people in newspaper

([8]). Citation matching is to determine if two references are of the same publication. Meanwhile, entity identification aims at locating in a knowledge base (KB) of discourse the entity that a name represents.

Closely related to our work are those of [5], [7] and [10]. The method proposed in [5] relies on affiliation, text proximity, areas of interest, and co-author relationship as clues for disambiguating person names in calls for papers only. Meanwhile, the domain of [10] is that of geographical names in texts. The authors use some patterns to narrow down the candidates of ambiguous geographical names. For instance, "*Paris, France*" more likely refers to the capital of France than a small town in Texas. Then, it ranks the remaining candidate entities based on the weights that are attached to classes of the constructed Geoname ontology. The shortcoming of those methods is to omit relationships between named entities with different classes, such as between person and organization, or organization and location, etc. In [7], some pattern matching rules written in JAPE's grammar ([12]) are applied to resolve simple ambiguous cases in a text, based on the prefix or suffix of a name. For example, "*John Smith, Inc.*" is recognized as an organization instead of a person because of the suffix "*Inc.*" implying an organization. Then, disambiguation uses an entity-ranking algorithm, i.e., priority ordering of entities with the same label based on corpus statistics.

Our work contrasts with the above-mentioned ones in both of the two following aspects, at once. Firstly, the problem that we address is not limited to named entities of a particular class or domain, but for all that may occur in texts of news domain, where texts are not as well-structured as those of publication references or calls for papers, and co-occurrence and relationship of named entities are essential to identify them. Second, we do not only disambiguate a name, but also identify the entity of that name in a KB of discourse. Here a KB is used both for the goal and as the means of a NE disambiguation process.

The intuition and assumption behind our approach is that the identity of the entity represented by an ambiguous name in a news article depends on co-occurring unambiguous named entities in the text. For example, suppose that in our KB there are two entities named "*Jim Clark*", one of which has a relation with the *Formula One* car racing championship and the other with *Netscape*. Then, if in a text where the name appears there are occurrences of *Netscape* or web-related named entities, then it is more likely that the name refers to the one with *Netscape* in the KB. After running NE recognition engine with respect to an ontology and a KB, those entities in the KB that are already mapped to the unambiguous named entities are called *disambiguated entities*. Also, each entity in the KB whose name is identical to an *ambiguous name* appearing in a text is called a *candidate entity* for that name.

2 Proposed Approach

In a news article, co-occurring entities are usually related to each other. Furthermore, the identity of a named entity is inferable from nearby and previously identified named entities in the text. For example, when the name "*Georgia*" occurs with "*Atlanta*" in a text, if *Atlanta* is already recognized as a city in the *United States*, then it is more likely that "*Georgia*" refers to a state of the *U.S.* rather than the country *Georgia*, and vice versa when it occurs with *Tbilisi* as a country capital in another text. From this observation, our method takes into account not only most probable classes of an ontology, but also other co-occurring named identities in a text of discourse. Concretely, the whole disambiguation process involves several iterations of the following steps (sub-sections 2.1, 2.2, and 2.3), where named entities identified in a round are used as a basis to disambiguate remaining ones in the next round.

2.1 Knowledge-Based Ranking of Candidate Entities

At this step, disambiguated entities in the text are exploited to filter the most promising candidate entities for a name occurrence. The idea is that, considering two candidate entities, the entity whose related entities in the KB occur more in the text is the more likely one. For example, there are one street in *Ha Noi* and one street in *Saigon* that have the same name. Then, in a text, if that name occurs with the city *Saigon*, then the name probably refers to the street in *Saigon*.

Let C be the set of candidate entities, E be the set of disambiguated entities, and f be the function expressing if there exists a relation between a candidate entity and a disambiguated entity. That is f: $C \times E \rightarrow \{0, 1\}$, where $f(c, e) = 1$ ($c \in C$, $e \in E$), if and only if there exists a relation between c and e; $f(c, e) = 0$, otherwise. Candidate entities are then ranked by the number of disambiguated entities that have relations with each entity as given in the KB; the more relations, the higher rank.

Particularly, we calculate confidence score for candidate entities. The confidence score of each candidate is increased one every time there is a relation encountered between that entity and any disambiguated entity so far. The set of the candidate entities with the highest score will be chosen. If two or more of the candidate entities have the same maximal confidence score, then the one that has the highest score and with close-proximity to the disambiguated entities in the same window context is selected. A window context could be a sentence, a paragraph, or a sequence of words containing the name under consideration.

2.2 Class-Based Ranking of Remaining Candidate Entities

This ranking is based on the assumption that entities of one class may be more common than those of another, whence the former class is given a greater weight ([10]). For example, when "*Athens*" appears in a text, it is more likely that it mentions the capital of *Greece* rather than a small town in the state *Georgia* of the *US*, because country capitals are more often encountered than small towns, as in news for instance.

Particularly, the remaining candidate entities after the knowledge-based ranking step are going to continue to be filtered. Each the candidate entity is assigned a score that reflects its preference in terms of its class over others. Then, the candidate entities with the highest score are chosen. We note that the classes of considered candidate entities must be subclasses of the same class *Person*, *Location*, or *Organization*. After this step, if the final set contains only one named entity, the ambiguous name is successfully resolved. The chosen candidate entity is then added to the set of disambiguated entities to be used for further disambiguation.

2.3 Named Entity Coreference Resolution

This step performs coreference resolution within a text to create coreference chains, which help to identify an entity after recognizing some others in the same chain. It can also be observed that a news article often use a full name (i.e., main alias) in its headline or first paragraph, before using other variants such as abbreviation or acronyms. For example, "*George W. Bush*" may be written first and then "*George Bush*", "*Mr. Bush*" or "*Bush*" used as anaphora. If "*George W. Bush*", "*Mr. Bush*" and "*Bush*" are found to be coreferent, then it is likely that "*Mr. Bush*" and "*Bush*" refers to the president of *United States* instead of *Barbara Bush*, *Laura Bush*, or *Samuel D. Bush*, for instance. We employ the rules in [2] for coreference resolution.

3 Experiments and Evaluation

3.1 Corpora and Experiments

For experiments, we employ KIM ontology, KB, and NE recognition module to produce annotated web pages. Currently, KIM ontology contains 250 named entity classes, 100 relations and attributes, while KIM KB is populated with over 200,000 entities. However, since KIM also performs NE disambiguation, its annotated texts do not contain ambiguous cases. Therefore, we have to re-embed all possible annotations for each name in a text.

Since our goal is to evaluate ambiguity resolution, we are interested in ambiguous names in news articles. The corpora are built from English news of CNN, BBC, New York Times, Washington Post, and Business Week. More specifically, two entities are selected for the disambiguation experiments, namely, ("*Georgia*", *Location*) and ("*Smith*", *Person*). There are totally 3267 named entities in the *Georgia* corpus, and 1467 named entities in the *Smith* corpus. Table 1 shows the occurrence times of each type of ("*Georgia*", *Location*) in the *Georgia* corpus. Table 2 shows the occurrence times of each person in the Smith corpus. For later testing, all the entities referred to in those corpora are manually disambiguated with respect to KIM KB, by two persons for the quality of the corpora. For the *Georgia* corpus in the experiments, we assume the class *Country* having a greater weight than *Province*.

Table 1. Occurrene times of ("*Georgia*", *Location*) in the *Georgia* corpus

Class for *Georgia*	Number of articles	Occurrence times
Province	30	116
Country	17	213
Total	47	329

Table 2. Occurrene times of ("*Smith*", *Person*) in the *Smith* corpus

Person for *Smith*	Position	Number of articles	Occurrence times
Jason Smith	COO	1	2
Richard A. Smith	Chairman	5	27
Rick Smith	CEO	2	3
Jason Smith	Finance. Dir	2	15
Richard J. Smith	CFO	2	2
Total		12	49

3.2 Performance Evaluation

For evaluation, we run KIM, KIM enhanced with class weighting, and our method on the same corpora. The results are matched against the manually disambiguated corpora. Table 3 and Table 4 summarize the results when the methods are tested on the *Georgia* corpus and the *Smith* corpus, respectively. The obtained precision and recall measures say that our method, which exploits NE relations in a KB, outperforms both KIM and its enhancement with class weighting.

Table 3. Testing on the *Georgia* corpus

Method	Correct disambiguation	Disambiguated names	Ambiguous names	Precision	Recall
KIM	194	312	329	62.17%	58.96%
KIM + Class weighting	207	315	329	65.71%	62.91%
Our method	306	315	329	97.14%	93.00%

Table 4. Testing on the *Smith* corpus

Method	Correct disambiguation	Disambiguated names	Ambiguous names	Precision	Recall
KIM	39	47	49	82.97%	79.59%
KIM + Class weighting	39	47	49	82.97%	79.59%
Our method	46	47	49	97.87%	93.87%

4 Conclusion

We have proposed a new approach to name disambiguation that exploits knowledge-based relations between named entities in a text under consideration. Firstly, it is quite natural and similar to the way humans do, relying on well-identified entities to resolve other ambiguous names in the context. Secondly, it is robust to free texts without well-defined structures or templates. The experiments have shown that our method achieves higher precision and recall performance measures than KIM even enhanced with class weighting.

Regarding class weighting, since in the experimental corpus there are only two classes, weight assignment is simple. For a complete ontology with several classes, automatic class weight assignment is desirable. Also, there is still contextual and knowledge-based information other than entity relationship that can be used for name disambiguation in a larger scale. We are currently investigating along those lines of research.

References

[1] Bekkerman, R., McCallum, A.: Disambiguating Web Appearances of People in a Social Network. In: Proc. of the 14th International World Wide Web Conference, Chiba, Japan, pp. 463–470 (2005)

[2] Bontcheva, K., Dimitrov, M., Maynard, D., Tablan, V., Cunningham, H.: Shallow Methods for Named Entity Coreference Resolution. In: Proc. of TALN 2002 Workshop, Nancy, France (2002)

[3] Chinchor, N., Robinson, P.: MUC-7 Named Entity Task Definition. In: Proc. of MUC-7 (1998)

[4] Guha, R., Garg, A.: Disambigusting People in Search. In: Proc. of the 13th International World Wide Web Conference, New York, USA (2004)

[5] Hassell, J., Aleman-Meza, B., Arpinar, I.B.: Ontology-Driven Automatic Entity Disambiguation in Unstructured Text. In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L. (eds.) ISWC 2006. LNCS, vol. 4273, pp. 44–57. Springer, Heidelberg (2006)

[6] Han, H., Giles, L., Zha, H., Li, C., Tsioutsiouliklis, K.: Two Supervised Learning Approaches for Name Disambiguation in Author Citations. In: Proc. of ACM/IEEE Joint Conference on Digital Libraries, Tucson, Arizona (2004)

[7] Kiryakov, A., Popov, B., Terziev, I., Manov, D., Ognyanoff, D.: Semantic Annotation, Indexing, and Retrieval. Journal of Web Semantics 2(1) (2005)

[8] Mann, G., Yarowsky, D.: Unsupervised Personal Name Disambiguation. In: Proc. of the 17th Conference on Natural Language Learning, Edmonton, Canada, pp. 33–40 (2003)

[9] Pasula, H., Marthi, B., Milch, B., Russell, S.J., Shpitser, I.: Identity Uncertainty and Citation Matching. Advances in Neural Information Processing Systems 15 (2002)

[10] Raphael, V., Joachim, K., Wolfgang, M.: Towards Ontology-based Disambiguation of Geographical Identifiers. In: Proc. of the 16th International WWW Conference, Banff, Canada (2007)

[11] Shadbolt, N., Hall, W., Berners-Lee, T.: The Semantic Web Revisited. IEEE Intelligent Systems 21(3), 96–101 (2006)

[12] Cunningham, H., Maynard, D., Tablan, V.: JAPE: A Java Annotation Patterns, Technical report CS–00–10, University of Sheffield, Department of Computer Science (2000)

Real-Time Scheduling for Non-crossing Stacking Cranes in an Automated Container Terminal

Ri Choe, Taejin Park, Seung Min Ok, and Kwang Ryel Ryu

Department of Computer Engineering, Pusan National University, Jangjeon-Dong San 30, Kumjeong-Ku, Busan 609-735, Korea
{choilee, parktj, seungmin, krryu}@pusan.ac.kr

Abstract. This paper proposes a local-search-based real-time scheduling method for non-crossing stacking cranes in an automated container terminal. Considering the dynamic property of the yard crane operation and real-time constraints, the method builds a new crane schedule for a fixed-length look-ahead horizon whenever a new crane job is requested. One difficulty in crane scheduling is that sometimes additional crane operations need to be done to complete a requested job, especially when other containers are stacked on top of the requested container. We use a redundant and variable-length representation of a candidate solution for search to accommodate those additional operations. Simulation experiment shows that the local-search-based method outperforms heuristic-based method in real-time situations.

Keywords: optimization, real-time scheduling, stacking crane, automated container terminal.

1 Introduction

An automated container terminal is largely divided into two areas: the quay and the stacking yard. The quay is the area where the containers are loaded to or discharged from the vessels. The stacking yard is the area where the export and import containers are temporarily stored. The stacking yard is divided into several tens of blocks. A block consists of a number of container stacks. The blocks are typically laid out in perpendicular to the quay in automated container terminals. There are handover points (HPs) for automated guided vehicles (AGVs), the equipment transporting containers between the quay and the yard, at the seaside end of a block. There also are HPs for the external trucks (ETs) at the landside end of a block. Automated stacking cranes (ASCs) handle containers at a block. The major categories of ASC operation are vessel and ET operations. A vessel operation is either a discharging or a loading and an ET operation is either a carrying-in or a carrying-out. For discharging, the ASC picks up from an AGV a discharged import container and then stacks it into the block. For loading, the ASC loads an export container stored in the block onto an AGV. For carrying-in job, the ASC stacks an export container into the block from an ET. For carrying-out job, the ASC loads an import container onto an ET from the block.

There are various researches on operations of stacking cranes. Kim et al. [1] and Oh et al. [2] studied the re-marshalling problem of export containers. In particular, Oh et al. proposed a stacking crane scheduling method for intra-block re-marshalling in the stacking yard where non-crossing stacking cranes are used. Kozan et al. [3], Ng [4], and Ng et al. [5] have also studied stacking crane scheduling. Most of these studies, however, are based on simple heuristic techniques except for a few methods using search techniques.

In this paper, we assume that two non-crossing ASCs are operating in each block and propose scheduling methods for such ASCs. The ASC closer to the seaside HP is called the seaside ASC and the other the landside ASC. Since the two ASCs cannot cross each other, interference can occur between them. As an extreme example, if the seaside ASC is to perform an ET operation, the landside ASC must move back out of the block to make room for the seaside ASC. For this reason, we let vessel operations be performed only by the seaside ASC and ET operations only by the landside ASC.

ASC operations are uncertain and dynamic. For example, the requests for ET operations are unpredictable because little information is available about the schedule of ETs. Therefore, an ET operation is requested only after the ET enters the gate of the terminal. Although there is a loading/discharging plan for each vessel, there is uncertainty on the arrival time of AGVs due to the delay in quay crane operations, interference among AGVs, and so on. For this reason, rescheduling is required whenever a new ASC job is requested.

We propose two scheduling methods—a heuristic-based method and a local-search-based method—and compare their performances. Our methods reschedule ASCs whenever a new job is requested. When a request for a new job occurs, the heuristic-based method selects and allocates a job to an appropriate ASC using heuristic rules. This method is readily applicable to real-time situations since it is computationally cheap. However, due to its greedy nature, the method cannot provide a really efficient crane schedule. The local-search-based method utilizes local search algorithms to optimize an ASC schedule for a fixed-length look-ahead horizon. Since it usually takes a few minutes for an ASC to complete a job, the method can exploit this time to conduct a local search.

One difficulty in scheduling ASCs is that sometimes additional crane operations need to be done to complete a requested job. For example, if an ASC should pick up a container on which some containers are stacked, rehandlings to remove these containers must be done before the target container can be lifted. If we regard these rehandlings as a part of the requested job, ASC scheduling can be simplified at the cost of the efficiency. On the other hand, if rehandlings are treated as independent jobs, they can be performed by any idle ASC in advance. This is advantageous for balancing the workload among ASCs. There is another type of operation that can contribute to further improvement of load balancing, that is repositioning. Repositioning refers to the operation of moving an export container to a location closer to the seaside or an import container closer to the landside from its original location. We call both the rehandling and the repositioning the assisting jobs because they are not the jobs requested explicitly but can optionally be performed by an idle ASC. By treating the assisting jobs as independent jobs, the efficiency of the ASC operation can be improved while the scheduling process itself becomes more complex. The number of decision parameters increases when such additional assisting jobs are considered independent.

For example, when a repositioning job is performed, whether or not the container should be repositioned, and where it needs to be moved must be determined. Although the heuristic-based method cannot, the local-search-based method can encompass most of the decision parameters with the appropriate representation and decoding of a candidate solution.

We adopt a redundant and variable-length representation of a candidate solution for the search of a crane schedule. In a candidate solution, both the requested jobs (also called main jobs) and the assisting jobs are arranged in a one-dimensional array according to the order in which they will be performed. Since the number of assisting jobs depends on the stacking configuration of the containers, the length of the candidate solution is variable. For each job in the candidate solution an ASC is assigned, which can be changed during neighbor generation. In addition, the sequence of jobs also changes during neighbor generation. For assisting jobs, we do not encode into the solution representation the intermediate stacking location to which the container is moved. It is determined by a simple greedy heuristic at the time the ASC performs the assisting job. We do not want the solution space to get too big to be searched in real time.

2 Heuristic-Based Crane Dispatching

This section explains the heuristic-based crane dispatching (HCD) method. HCD only considers rehandling jobs as assisting jobs not repositioning jobs. The intermediate location for the rehandled container to be put is simply decided by just selecting the stack that is the nearest to its current location where there are no loading or carrying-out containers.

Whenever an ASC finishes a job, the HCD method first selects candidates among jobs and then chooses and allocates one of them to the ASC. Candidate jobs are selected by candidate-selection heuristics among the jobs able to be completed. A main job is only able to be completed if the AGV or ET for it is waiting at the HP because unless the AGV or ET is ready the ASC should wait until its arrival. In contrast, rehandling jobs are able to be completed any time because they require neither AGVs nor ETs to complete them. Thus, rehandling jobs can be included in the candidates although its main job is not. Therefore, the rehandling of containers can be done much earlier than its main job. A candidate-selection heuristic determines whether or not HCD considers co-operation between ASCs by selecting rehandling jobs. There are three different candidate-selection heuristics. The first heuristic does not consider co-operation. It selects rehandlings for vessel operations for the seaside ASC (but not the landside ASC) and those for ET operations for the landside ASC (but not the seaside ASC). The second and third heuristic consider co-operation. To reduce interference between ASCs, they divide a block into three equal-sized areas. Rehandling jobs in the seaside area are selected only for the seaside ASC, and those in the landside area are selected only for the landside ASC. For the rehandling jobs in the central area the second heuristic selects them for both ASCs. On the other hand, the third heuristic selects them for the landside ASC to make it help the seaside ASC because vessel operations are more important than ET operations.

After candidate selection, a job is selected from the candidates by the job-selection heuristic and allocated to the ASC. For the job-selection heuristic, the earliest deadline first (EDF) and minimum empty travel (MET) heuristic are used. EDF chooses a job with the earliest deadline. MET selects a job whose target container is the closest to the ASC. Combining these three candidate-selection heuristics and two job-selection heuristics, we obtain six HCD versions: EDF1, EDF2, EDF3, MET1, MET2, and MET3.

3 Local-Search-Based Crane Scheduling

This section explains a local-search-based crane scheduling (LCS) method. When an ASC finishes a job, the LCS method starts to build a new crane schedule for a fixed-length look-ahead horizon and the ASC performs the most immediate job that has been allocated to it according to the most recently generated schedule while scheduling. The length of the look-ahead horizon is set to be relatively short[1] to reduce the size of the search space and thus to meet the real-time constraint. The LCS method requires representation of a candidate solution, neighbor generation methods, and an evaluation method for the candidate solutions.

A candidate solution is represented as a one-dimensional array. Each element of the array indicates a job as well as the ASC to perform it. There are four types of main jobs—loading (L), discharging (D), carrying-in (I), and carrying-out (O)—and two types of assisting jobs—rehandling (R) and repositioning (P). The length of a candidate solution is variable since the number of main jobs in the look-ahead horizon is not constant and the number of assisting jobs depends on the stacking configuration of the containers.

An initial solution is generated as follows. First, main jobs are arranged in the order in which they do not incur any unnecessary rehandling. Next, repositioning jobs are generated for each loading and carrying-out job, and then rehandling jobs are generated for the containers stacked on top of the requested container. These assisting jobs are inserted into the location immediately following to the corresponding main job. Because an assisting job that comes after its main job is ignored when decoding the solution, this means that the initial schedule is built without assisting jobs. Finally, each job is allocated to a crane. Vessel operations (L and D) are allocated to the seaside ASC, and ET operations (I and O) to the landside ASC. The assisting jobs (R and P) are randomly allocated to either ASC. When building a new schedule, the initial solution is generated from the previous schedule. Figure 1 (b) shows an example initial solution for the jobs in Figure 1 (a).

The neighbor solutions are generated in two ways: by changing the sequence of the jobs and by changing the ASC to perform it. The first method randomly selects a job from the original solution and moves it to a position that is also selected at random. The second method randomly selects an assisting job and changes the allocated ASC to a different one. The two methods are applied with uniform probability.

Note that there are precedence constraints among the jobs, which are imposed by the stacking configuration of the containers. The arrows in Figure 1 (a) shows, for

[1] In this study, we set the length of the look-ahead horizon to 15 min.

example, that the rehandling of container C_2 (R_2) cannot be done before container C_1 is removed by carrying-out (O_1), repositioning (P_1), or yet another rehandling (R_1). When generating a neighbor by the first method, the precedence constraints can be violated. Therefore, a repair process may be needed after a neighbor solution is generated by the first method. The repair algorithm is shown in Figure 2. The solution in Figure 1 (d) is the repaired solution obtained by repairing the solution in Figure 1 (c) by the algorithm.

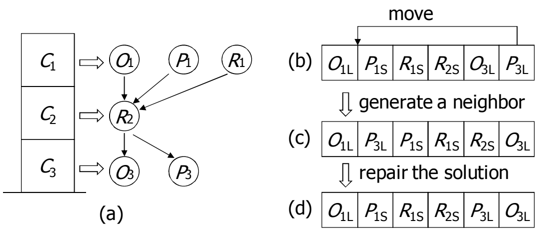

move
(b) O_{1L} P_{1S} R_{1S} R_{2S} O_{3L} P_{3L}
generate a neighbor
(c) O_{1L} P_{3L} P_{1S} R_{1S} R_{2S} O_{3L}
repair the solution
(d) O_{1L} P_{1S} R_{1S} R_{2S} P_{3L} O_{3L}
C_1 C_2 C_3
O_1 P_1 R_1 R_2 O_3 P_3
(a)

Fig. 1. (a) An example of precedence constraints among the jobs. (b) The initial solution. (c) A neighbor generated by changing the job sequence. (d) The solution after repair process. The subscript number of each job is the same as that of its target container. The subscript "L" and "S" denotes whether the landside ASC or seaside ASC is allocated to the job, respectively.

```
For all the jobs for containers in the stack where the container of the moved job is stacked.
Step 1. Put all the jobs with the highest precedence into set $J$.
Step 2. Select the most immediate job $j$ from $J$ according to the job order in the solution and
        then push it into queue $Q$ and remove it from $J$.
Step 3. If the precedence of $j$ is the lowest in $J$, put all the jobs with the next precedence into $J$.
Step 4. Repeat steps 2 and 3 until $J$ is empty.
Step 5. Rearrange the jobs in the solution according to the order of jobs in $Q$.
```

Fig. 2. Repair algorithm for a neighbor solution

To evaluate a candidate solution, a simulation is conducted in which the ASCs operate according to the given solution. If interference occurs, priority is given to the ASC working for the job which precedes in the solution. When decoding a solution, an assisting job put after its main job in the solution is ignored. Now consider the case in which both the assisting jobs P_i and R_i for container C_i precede their main job in the solution. If P_i precedes R_i, R_i is ignored since C_i has already been removed by P_i. If R_i precedes P_i, however, P_i is still performed after R_i. Even though C_i is moved to a new location by R_i, the location is usually close to the original location because the operation is just a rehandling. Therefore, repositioning may still be needed to reduce the operation time of the ASC.

The target location of a container to be rehandled or repositioned is determined among the locations where the container does not cause another rehandling when it is moved. For rehandling, the nearest location from its original location is selected. For repositioning, the location closest to the destined HP is chosen on the condition where the ASC does not interfere with the current operation of the other ASC during repositioning.

The evaluation value of a given candidate solution s after the simulation is calculated by the following equation

$$E(s) = w \times DT_{AGV} + WT_{ET} \tag{1}$$

where DT_{AGV} is the total delay time of AGVs and WT_{ET} is the total waiting time of ETs. The weigh w reflects the relative importance of the vessel and ET operations. Since vessel operations are more important than ET operations for the productivity of a container terminal, the weight w is set to 60.

4 Experimental Results

We conducted simulation experiments to evaluate the performance of the HCD and LCS methods. The terminal to be simulated has three quay cranes at the quay and seven blocks in the stacking yard. A block is 41 TEU (twenty-foot equivalent unit container) bays long and a bay can accommodate 10 rows and 5 tiers of containers. The stacking yard was assumed to be initially occupied to 70% of its capacity.

We experimented with three different scenarios. The scenarios are discriminated by whether the cranes can co-operate or not. In scenario 1, we compared four methods EDF1, MET1, HC, and SA, without crane co-operation. Rehandling jobs are regarded as a part of their main jobs and performed at the time when their main jobs are performed. EDF1 and MET1 are the implementations of the HCD method explained in section 2. HC and SA are the implementations of the LCS method explained in section 3. HC and SA stand for hill-climbing and simulated annealing search algorithms, respectively [6]. In scenario 2, eight methods EDF1, EDF2, EDF3, MET1, MET2, MET3, HC, and SA with prior rehandling only are compared. The numbers 1, 2, and 3 denote the three candidate-selection heuristics applied. Finally, in scenario 3, HC and SA with both the prior rehandling and repositioning are compared. For each scenario, DT_{AGV} and WT_{ET} per job are measured and averaged.

Table 1 shows the result of scenario 1, 2, and 3. In scenario 1 HC and SA perform better than the HCD methods and SA gives the best result. MET1 showed both the longest DT_{AGV} and WT_{ET} because it focuses on reducing the empty travel time of the ASCs rather than meeting the deadlines. Similarly to the results of scenario 1, in scenario 2 SA is the best and the LCS methods outperformed the HCD methods. Although EDF1 and MET1 adopt only prior rehandling without co-operation, they showed significant improvement in comparison to the results of scenario 1. By allowing prior rehandling, ASCs can perform rehandlings before the AGVs or ETs arrive, thereby reducing the loss in efficiency caused by the delayed arrival of AGVs or ETs. Further improvement is obtained when the HCD methods adopt the crane co-operation. Both EDF2 and EDF3 perform better than EDF1. In particular, DT_{AGV} with EDF3 is reduced by 22.38% in comparison to that with EDF1. By adopting repositioning, both DT_{AGV} and WT_{ET} are further improved according to the result of scenario 3. However, the degree of improvement is relatively small because repositioning requires extra pick-up and drop-off operations of the ASCs. Moreover, the container location for repositioning is determined by a simply greedy heuristic. Nevertheless, the performance of HC in scenario 3 improves to the extent that it is better than that of SA in scenario 2, the best one in the scenario.

Table 1. Results of scenario 1, 2, and 3

	Scenario 1		Scenario 2		Scenario 3	
Methods	DT_{AGV}(s)	WT_{ET}(s)	DT_{AGV}(s)	WT_{ET}(s)	DT_{AGV}(s)	WT_{ET}(s)
EDF1	12.10	1106	4.29	383		
EDF2			4.20	258		
EDF3			3.33	325		
MET1	15.45	1867	4.45	476		
MET2			4.10	447		
MET3			3.75	445		
HC	11.20	682	3.21	198	3.12	189
SA	10.08	643	3.15	190	3.12	178

6 Conclusions

We proposed two scheduling methods HCD and LCS for two non-crossing ASCs working in a block. By being able to treat the rehandling and repositioning operations as independent jobs in our solution methods, co-operation between the two ASCs is greatly facilitated. Through co-operation, the workload of the two ASCs can be better balanced and the interference can be avoided more easily, thereby maximizing the performance. According to the results of simulation experiments, the delay time of AGVs and the waiting time of ETs are significantly reduced by adopting crane co-operation. Also, since the number of decision parameters increases when considering co-operation, the LCS method which determines these parameters more elaborately turns out to give better performance than the HCD method.

Acknowledgments. This work was supported by the Korea Research Foundation Grant funded by the Korean Government (MOEHRD) (The Regional Research Universities Program/Research Center for Logistics Information Technology).

References

1. Kim, K.H., Bae, J.W.: Re-marshaling Export Containers. J. Computers and Industrial Engineering 35, 655–658 (1998)
2. Oh, M.S., Kang, J., Ryu, K.R., Kim, K.H.: A Two-step Approach to Scheduling Transfer Cranes for Intra-block Remarshalling. In: Proceeding of Intelligent Manufacturing and Logistics Systems (2005)
3. Kozan, E., Preston, P.: Genetic Algorithms to Schedule Container Transfers at Multimodal Terminals. International Transactions in Operational Research 6, 311–329 (1999)
4. Ng, W.C.: Crane scheduling in container yards with inter-crane interference. J. Operational Research 164, 64–78 (2005)
5. Ng, W.C., Mak, K.L.: An effective heuristic for scheduling a yard crane to handle jobs with different ready times. J. Engineering Optimization 37, 867–877 (2005)
6. Russel, S.J., Norvig, P.: Artificial Intelligence: A Modern Approach. Prentice-Hall, Englewood Cliffs (2003)

The Detrimentality of Crossover

Andrew Czarn[1], Cara MacNish[1], Kaipillil Vijayan[2], and Berwin Turlach[2]

[1] School of Computer Science and Software Engineering,
The University of Western Australia,
Crawley WA 6009,
[2] School of Mathematics and Statistics,
The University of Western Australia,
Crawley WA 6009

Abstract. The traditional concept of a genetic algorithm (GA) is that of selection, crossover and mutation. However, data from the literature has suggested that the niche for the beneficial effect of crossover upon GA performance may be smaller than has been traditionally held. We explored the class of problems for which crossover is detrimental by performing a statistical analysis of two test problem suites, one comprising linear-separable non-rotated functions and the other comprising the same functions rotated by 45 degrees rendering them not-linear-separable.

We find that for the *difficult* rotated functions the crossover operator is detrimental to the performance of the GA. We conjecture that what makes a problem difficult for the GA is complex and involves factors such as the degree of optimization at local minima due to crossover, the bias associated with the mutation operator and the Hamming Distances present in the individual problems due to the encoding.

Finally, we test our GA on a practical landscape minimization problem to see if the results obtained match those from the difficult rotated functions. We find that they match and that the features which make certain of the test functions difficult are also present in the real world problem.

1 Introduction

The detrimentality of crossover in GAs has been the subject of considerable debate. Work by Eshelman and Schaffer [1], entitled Crossover's Niche, suggested that there exists a unique niche for which crossover is advantageous, and that it is smaller than has been traditionally held in the GA community. Saloman suggested that Crossover's Niche is linear-separable problems [2] but that problems in the real world are not linearly-separable but tend to be nonlinear, chaotic and stochastic.

Statistical analyzes of GA performance have added to this debate. Schaffer et al [3] conducted a factorial study using the analysis of variance (ANOVA) to examine the De Jong suite plus an additional five problems. Close examination of the best on-line pools suggested a relative insensitivity to the crossover operator when using Gray encoding.

Our work builds on these findings. We are interested in using statistical analysis to answer three questions:

1. Can the crossover operator be statistically demonstrated to be *detrimental* for a given problem?

2. Is not-linear-separability a sufficient determinant of the detrimentality of crossover?
3. If not, what other factors are involved?

The first question was answered in our earlier work which was a statistical exploratory analysis of GAs [4] of four benchmark functions: De Jong's *F1* [5] known as the SPHERE, De Jong's *F3* [5] known as the STEP function, De Jong's *F2* [5] known as ROSENBROCK'S SADDLE and Schaffer's *F6* [3]. Notably, F1 and F3 are linear-separable[1], whereas F2 and F6 are not. In this study we focus on questions 2 and 3.

2 Methods

For details of our statistical methodology the reader is referred to [4]. To determine whether linear-separability is indeed a determining factor while minimizing other effects, we examined a series of functions of increasing difficulty, while also examining the *same* functions in different orientations We achieved this by rotating the functions by 45 degrees rendering them not-linear-separable. We then tested the algorithm on a newly devised benchmarking problem from the Huygens Suite [6,7]. These functions are detailed below:

1. Test functions *FNn* for n=1 to n=6, which are linear-separable equations, as displayed in Equation 1 below:

$$FNn(x_1, x_2) = \sum_{i=1}^{2} 0.5(1 - \cos(\frac{n\pi x_i}{100})e^{-|\frac{x_i}{1000}|}), -100 \leq x_i \leq 100. \quad (1)$$

2. Test function *FNnR45* (*R45* standing for the original test function *FNn* having been *rotated by 45 degrees* in the solution space), being not-linear-separable, for n=1 to n=6 as displayed in Equation 2 below:

$$FNnR45(x_1, x_2) = 0.5(1 - \cos(\frac{n\pi\frac{x_1+x_2}{\sqrt{2}}}{100})e^{-|\frac{\frac{x_1+x_2}{\sqrt{2}}}{1000}|}) + 0.5(1 - \cos(\frac{n\pi\frac{x_1-x_2}{\sqrt{2}}}{100})e^{-|\frac{\frac{x_1-x_2}{\sqrt{2}}}{1000}|}), -100 \leq x_i \leq 100. \quad (2)$$

3. MacNish has devised a problem series for benchmarking, that based on fractal landscapes, reflect the attributes of highly multimodal problems seen in real world situations [6]. We chose to run our GA on the first landscape in MacNish's 20 series.

3 Results

3.1 Exploratory Analysis of Test Functions *FN1* to *FN6*

Full ANOVA tables and regression analyzes for test functions *FN1* to *FN6* are published in [8]. The analyzes revealed that the optimal value of crossover was 100% for each of the six functions.

[1] We define linear-separable problems as those where the objective function can be written as a sum of univariate functions, which are allowed to be non-linear, where each of the functions can take one component of the input vector as an argument.

3.2 Exploratory Analysis of Test Functions *FN1R45* to *FN6R45*

Again, full ANOVA tables and regression analyzes for test functions *FN1R45* to *FN6R45* are published in [8]. In contrast to the non-rotated forms crossover proved detrimental for test functions *FN2R45*, *FN4R45* and *FN5R45*, with an optimal crossover rate of 0%. Crossover was beneficial, however, for *FN1R45*, *FN3R45* and *FN6R45*. This shows that not-linear-separability alone is not a sufficient indicator.

4 Factors Affecting the Usefulness of Crossover

4.1 Optimization Occurring at Local Minima Due to Crossover

To quantify the degree of optimization at the local minima carried out by crossover we recorded and compared the relative proportion of times crossover and mutation improved the best fitness obtained by the population.

The results were that crossover improved fitness at sites of local minima 82% of the time out of the total number of epochs (with a 99% confidence interval of 80% to 84%) compared to mutation with a value of only 30% (with a confidence interval of 29% to 31%). This suggested that crossover was primarily performing the role of exploitation of local minima while mutation was responsible for exploration. This was confirmed by computer animations of the population.

4.2 Bias Associated with the Mutation Operator

To ascertain in the present work if there was any bias associated with the mutation operator we carried out experiments where many copies of a single chromosome comprised of two-bit strings were mutated and plotted onto a heat map representation of the rotated function *FN2R45*. We observed that the chromosomes landed in a grid-like pattern along the x and y directions illustrating that it is biased in the axial direction. This axial bias is a special case of the more general relationship between the problem encoding and the solution space, discussed below.

4.3 Relationship Between Gray Encoding and the Solution Space

Heat map animations illustrated not only a grid-like pattern with regions of higher density and others of much lower density but also that in general it is much harder to make a “jump” to some areas of the space than others.

An illustrative case for the rotated functions is that of *FN2R45* and *FN3R45*. *FN2R45* was the more difficult of the two functions for the GA. This is evidenced by the fact that the number of epochs taken to reach the threshold was an order of magnitude greater. This is despite the fact that *FN3R45* has the higher modality. To illustrate why this is the case, we can examine the Hamming Distances of the two functions as displayed in Figure 4.

As can be seen, *FN2R45* has the larger Hamming Distance of 12 from any of the local optima to the global optimum for both the x bit string or the y bit string.These Hamming Distances are a direct consequence of the relationship between the encoding and the

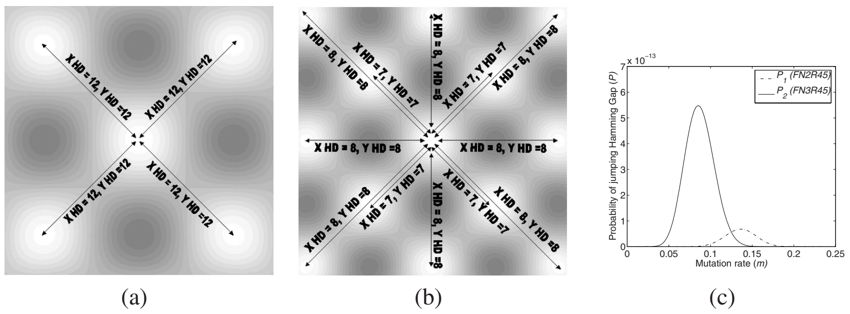


Fig. 1. (a) Hamming Distances for *FN2R45* (b) Hamming Distances for *FN3R45* (c) Probability of jumping Hamming Gap versus Mutation Rate

solution space. It is interesting to note that finding the optimal mutation rate appears to be a case of finding a fixed point that is high enough up the Hamming Distance probability curves for the space while at the same time minimizing the disruptive effect of mutation on convergence.

5 Extending the Results to Difficult Practical Problems

Before leaving this topic, we wanted to see if there was evidence the results would carry over to real-world problems exhibiting the same properties for difficult problems. In order to extend the results to a difficult practical problem, we tested our GA on *Landscape 20_101* [6,7]. Again, the full ANOVA table and the regression analysis are published in [8]. The same behaviour of the GA emerged as for the difficult rotated functions.

We can conjecture that this problem proved difficult for the GA for similar reasons to the problems analyzed earlier. In the first case the random arrangement of the local minima of this problem makes it unlikely that any of the local minima are aligned in the axial directions. Thus, the bias of mutation means that it is less likely that the global minimum will be found by chromosomes moving in the x and y directions.

In reference to crossover, the fact that the surface of *Landscape 20_101* has a great number of local minima means that it is very likely that crossover was enacting optimization at the local minima sites. This is supported by the fact that the optimal mutation rate was high at 18.93% (bit flipping mutation rate of 9.46%), suggesting that a high mutation rate was required to get chromosomes to jump out of regions of local minima where they were “stuck” due to local optimization carried out by crossover.

6 Discussion

The traditional concept of a GA, that of selection, crossover and mutation, is being challenged. We undertook the present research to explore the issue of the detrimentality of crossover using a rigorous statistical methodology.

In the first instance the results from our linear-separable test function series, *FNn*, show that crossover is beneficial for these linear-separable problems. On the other hand, results from the rotated not-linear-separable test function series demonstrated several instances where crossover was statistically proven to be detrimental. This was not true of all problems in this class indicating that not-linear-separability is not a sufficient determinant for the detrimentality of crossover. Rather it occurred for not-linear-separable problems which required the highest mutation rates, which in our experience has been a marker for the difficulty of a problem. These results were backed up by a difficult real world problem. Thus, what makes a not-linear-separable problem hard for a GA to solve is linked to whether crossover will be detrimental to the performance of the GA solving the problem.

We found that three factors were involved in making a not-linear-separable problem hard for the GA to solve. These were optimization carried out by crossover at the sites of local minima, the bias of the mutation operator and the Hamming Distances for the individual problems.

In conclusion, we have been able to demonstrate that crossover is statistically detrimental for the difficult not-linear-separable problems and also for a difficult real world problem with similar attributes.

References

1. Eshelman, L.J., Schaffer, J.D.: Crossover's niche. In: Proc. Fifth International Conference on Genetic Algorithms, pp. 9–14. Morgan Kaufmann, San Francisco (1993)
2. Droste, S., Jansen, T., Wegener, I.: On the analysis of the (1+ 1) evolutionary algorithm. Theoretical Computer Science 276, 51–81 (2002)
3. Schaffer, J.D., Caruana, R.A., Eshelman, L.J., Das, R.: A study of control parameters affecting online performance of genetic algorithms for function optimization. In: Proc.Third International Conference on Genetic algorithms, pp. 51–60. Morgan Kaufmann, San Francisco (1989)
4. Czarn, A., MacNish, C., Vijayan, K., Turlach, B., Gupta, R.: Statistical exploratory analysis of genetic algorithms. IEEE Transactions on Evolutionary Computation 8, 405–421 (2004)
5. De Jong, K.A.: Analysis of the Behavior of a Class of Genetic Adaptive Systems. Phd dissertation, Department of Computer and Communication Sciences, University of Michigan, Ann Arbor, MI (1975)
6. MacNish, C.: Benchmarking evolutionary and hybrid algorithms using randomized self-similar landscapes. In: Wang, T.-D., Li, X., Chen, S.-H., Wang, X., Abbass, H., Iba, H., Chen, G., Yao, X. (eds.) SEAL 2006. LNCS, vol. 4247, pp. 361–368. Springer, Heidelberg (2006)
7. MacNish, C.: Towards unbiased benchmarking of evolutionary and hybrid algorithms for real-valued optimisation (to appear in Connection Science, special issue on Evolutionary Learning and Optimization)
8. Czarn, A., MacNish, C., Vijayan, K., Turlach, B.: Statistical exploratory analysis of genetic algorithms: The detrimentality of crossover. Technical Report UWA-CSSE-07-002, School of Computer Science and Software Engineering, The University of Western Australia Crawley WA 6009 Australia, http://web.csse.uwa.edu.au/research/Technical_Reports (2007)

Automatic Sapstain Detection in Processed Timber

Jeremiah D. Deng and Matthew T. Gleeson

University of Otago, PO Box 56, Dunedin, New Zealand
ddeng@infoscience.otago.ac.nz

Abstract. Research on automatic wood defect inspection has hardly studied sapstain. We propose to use machine learning techniques such as feature selection, visualization and classification, to build an automatic sapstain detection system.

1 Introduction

Sapstain is caused by fungi that can infect and subsequently damage the wood quality. Timber can be infected at any time during manufacturer if the timber becomes moist enough. Sapstain can penetrate deep into the timber, which often means the removing the surface stain will not eliminate the sapstain from the wood. It is a real problem for the timber industry as it can cause the wood to lose its strength over time and generally considered a defect that must not appear in finished products.

The visual discolourization caused by sapstain can vary from completely covering the sapwood to appearing as specks, streaks or patches of various intensities of colour. The colour of the stain can vary depending on the infecting organism, species and moisture condition of the wood. The most common fungal stain is blue-stain, as shown in Figure 1. The colour of blue-stain typically varies from bluish to bluish black and grey to brown, but can also be of shades of yellow, orange, purple and red. Human inspection

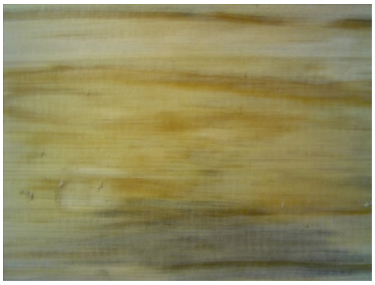


Fig. 1. Blue-stain can be seen at the bottom of the image

of defects in timber is the typical method of classifying defects from clear wood with timber manufacturers, but this can be rather inaccurate. For example, in a recent study [1], human inspection reached an 80% detection rate of knots in pine timber. While prior research of automatic timber inspection systems have focused on the detection of defects such as knots, to our knowledge there are no attempts made in the detection of sapstain. In this paper, we will employ machine learning techniques to analyse the colour and texture features extracted from wood images, and build classification systems that can operate automatically or provide computer-aided sapstain detection.

2 Relevant Works

There is a growing trend in employing machine vision and image analysis techniques in wood processing applications. For inspection of timber, a number of studies have been conducted [1][2][3]. Self-organizing maps (SOM) is a widely used algorithm to visualize and classify wood defects e.g. in [3]. Centiles have been widely used in defect inspection in timber with good results (eg. [3]) and again are computationally light suiting the practical application of colour centiles with automatic wood defect systems.

3 Experiment Settings

120 images were taken from 20 different timber boards using a digital camera. An experienced defect inspection expert then identified the clear and sapstain regions within the 120 images. The regions were extracted from the images and then split into sample pictures of 30×30 pixels. We collected 2500 sample images from each class (either ‘clear’ or ’sapstain’), forming a data set of 5,000 in total. Experiments were then conducted to evaluate various feature schemes and find out the most effective feature combination that can be used for classification. A number of feature ranking techniques are employed to examine a few colour and texture features. The choice of features are then assessed using some classifiers with 10-fold cross validation.

4 Feature Extraction

4.1 Colour Features

We assessed several colour features: a) RGB colour means. As sapstain dominantly appears as a discolouring defect, using colour means seems to be relevant. For each sample image, the average RGB components are computed. b) Colour centiles. Colour centiles are colour histogram features. They have been proven to be consistent despite different colouring shades due to varying illumination effects [1]. We use 30% centiles in this study. c) Colour-Opponent model. Ohta and Wandell are two widely-used colour-opponent models. Assessed by scatter plots of these colour features, the overlaps between samples of two classes seem to be less obvious in colour-opponent models.

4.2 Texture Features

Two texture features are investigated: Gabor filtering and local binary Patterns (LBP). Gabor filters are a set of Gaussian modulated sine defined on a few spatial orientations. Gabor filtering has been widely used in image segmentation, object classification, and image retrieval [4]. We used a 15-dimension feature based on mean of filtering responses. LBPs have been shown to be a simple and effective method of texture discrimination based on a simplification of directional grayscale difference histogram [5]. LBPs provide information about the spatial structure of textures in an image.

5 Visualization

Visualization of timber data is important in a wood inspection system. Self-organizing map (SOM) [6] as a key visualization algorithm has been successfully used in several automatic defect detection systems [1,7]. As the timber is put through the automatic detection system, images are taken and fed into the system. The images are then divided into regions so each region can be classified as sapstain or clear wood. The size of region is dependent upon the resolution of the camera, the required quality of classification and the desired time to classify each region. Using smaller regions results in a larger classification time but allows for a more accurate classification. On the other hand, regions have to be big enough for colour and texture features to be applicable. In this study a region size of 30×30 pixels was used as this gives adequate classification results and fast processing time.

On a trained 2-D feature map, the boundary between sapstain and clear wood was automatically drawn by using the pre-assigned labels of each data sample. Fig. 2 displays the trained feature maps using Wandell colour opponents and LBP as features respectively. All colour features successfully generated with 2-D maps, with the Wandell colour opponents being the best, where distinctive cluster boundary between ‘sapstain’ and ‘clear’ is shown. This can be fairly useful for the inspection of classification results, with a new sample mapped to a best-matching unit that may fall either in the ‘sapstain’

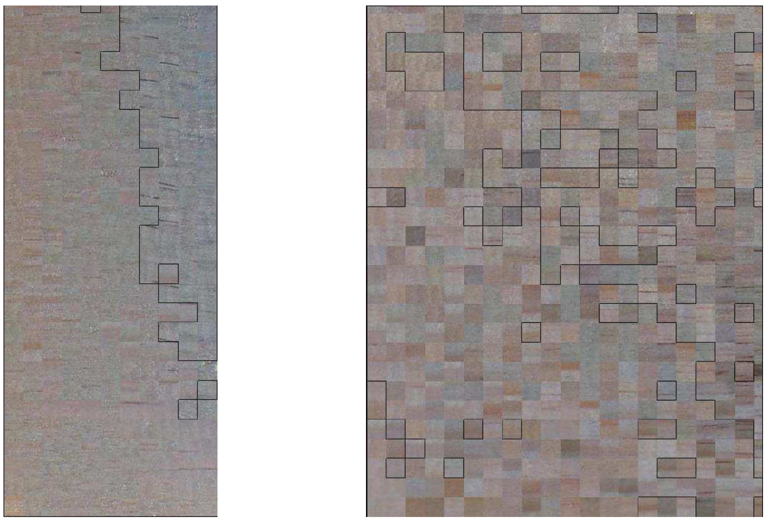

(a) (b)

Fig. 2. Feature maps generated from feature schemes: (a) Wandell colour opponents, and (b) LBP

region or the ‘clear’ region. However, the LBP feature set resulted in a very ‘choppy’ map and there is no continuous boundary between two classes.

6 Feature Selection and Classification

To evaluate the quality of the features, we assessed the performance of the feature schemes, each individually and also in combination, for classification. Four classifiers are used: k-nearest neighbour with $k = 11$ and distance weighting; Naive Bayes with kernel estimation; RBF with 24 or 32 Gaussian clusters; and SVM with complexity value of 20. Results are listed in Table 1.

Table 1. Performance of individual feature schemes and their combinations

Accuracy (%)	k-NN	NB	RBF	SVM	SOM
RGB	89.54	77.32	87.82	88.50	88.00
RGB Centile	89.12	-	-	-	86.30
Wandell	90.38	86.50	88.40	88.60	88.40
Ohta	90.18	86.00	89.22	88.54	-
Gabor	82.26	70.52	75.34	76.36	-
LBP	72.70	65.92	63.94	65.32	65.50
Ohta + Gabor	93.58	80.80	87.58	89.52	-
Wandell + Gabor	93.74	74.78	87.74	89.46	-
Wandell + LBP	92.58	83.48	65.32	88.96	-
ALL	93.84	83.32	89.66	89.36	-

It seems that among the colour features, Wandell colour opponents work the best almost for all classifiers, although the margin to the performance of Ohta colour components is quite small. As with texture, Gabor features clearly outperform the LBP. The best combination of colour and texture features is ‘Wandell + Gabor’, which is as high as 93.74 and comparable to that of using all the features in a k-NN classifier. The performance of SOM is slightly worse than the k-NN. For instance, its classification rate on the Wandell set and LBP set is 88.4% and 65.5% respectively when the first five best-matching SOM prototypes are used in classification, similar to k-NN.

We used a few attribute selectors to find the seven best feature components. Feature ranking and selection methods used include: Information Gain (IG), Symmetric Uncertainty (SU), and SVM. These all identified that ‘Average Red’ and ‘Average Ohta2’ are the most relevant features along with some Wandell features and Gabor features. We also tested the performance of the selected features and compared with the results obtained using the first seven principal components obtained by PCA. It is found that surprisingly all reduced feature sets can achieve very good classification accuracies. Using k-NN, the performance is 90.66%, 93.54% and 94.20% for feature selection based on IG, PCA, and SVM respectively. These are better or almost the same accuracy as that of using the whole feature set.

Another problem that arose which could potentially cause a lot of frustration in practical application was the occurrence of single regions being classified as sapstain when in fact the region was clear wood but had a similar colour to that of light sapstain. In practical application a region classified as sapstain would be cut and removed from the timber board. This could cause valuable lengths of clear board to be lost. To overcome this, the neighbours of each classified region were inspected; if no neighbours within a 1-block distance of the current region were also classified as sapstain, it meant the region was an isolated sapstain region and was thus re-classified as clear wood.

7 Conclusion

Much research has been done on detecting defects such as knots, but sapstain is of a different nature to knots in terms of visual appearance. We found that colour features provide good results for classifying sapstain and clear wood. RGB colour means, colour-opponent combinations and centiles all had an adequate classification rate of greater than 80%. Due to the lack of texture difference between sapstain and clear wood, texture features have much poorer classification rates. However, combining with colour features (such as Wandell colour opponents), better performance can be obtained than using single texture or colour feature. Furthermore, by conducting feature analysis, it is found that texture features (Gabor filtering features) are among the best seven feature for classification, which explained why the inclusion of texture features actually help to improve performance.

As for the choice of classifiers, it is found that k-NN can give efficient but comparable (sometimes better) performance as SVM. As a good choice for visualization purpose, SOM does not perform the best in classification, yet its performance is very close to that of k-NN classifiers.

Automatically detecting sapstain would be of a great value to automatic defect inspections systems in the timber industry. The results of this study could be incorporated into exiting automatic detection systems using SOM. Combined with other classification algorithms such as k-NN and SVM it is hopeful that a satisfactory accuracy can be achieved in a real-world system. On the other hand, practical implications such as speed and the size of SOM output map need to be investigated in a future work.

References

1. Lampinen, J., Smolander, S.: Wood surface inspection system based on generic visual features. In: Proc. of ICANN 1995 (1995)
2. Silven, O., Kauppinen, H.: Color vision based methodology for grading lumber. In: Proceedings of the 12th ICPR, pp. 787–790 (1994)
3. Niskanen, M., Silven, O., Kauppinen, H.: Color and texture based wood inspection with non-supervised clustering. In: Proc. 12th Scandinavian Conference on Image Analysis, pp. 336–342 (2001)
4. Manjunath, B., Ohm, J., Vasudevan, V., Yamada, A.: Colour and texture descriptors. IEEE Trans on Circuits and Systems for Video Technology 11(6), 703–715 (2001)
5. Ojala, T., Pietikainen, M., Harwood, D.: A comparative study of texture measures with classification based on feature distributions. Pattern Recognition 29, 51–59 (1996)
6. Kohonen, T.: Self-organizing maps. Springer, Heidelberg (1997)
7. Silven, O., Niskanen, M., Kauppinen, H.: Wood inspection with non-supervised clustering. Machine Vision and Applications, 275–285 (2003)

Structure-Sensitive Learning of Text Types

Peter Geibel[1], Ulf Krumnack[1], Olga Pustylnikov[2], Alexander Mehler[2], Helmar Gust[1], and Kai-Uwe Kühnberger[1]

[1] University of Osnabrück, Institute of Cognitive Science, AI Group, Germany
{pgeibel,krumnack,hgust,kkuehnbe}@uos.de
[2] University of Bielefeld, Text Technology Group, Germany
{Alexander.Mehler,Olga.Pustylnikov}@uni-bielefeld.de

Abstract. In this paper, we discuss the structure based classification of documents based on their logical document structure, i.e., their DOM trees. We describe a method using predefined structural features and also four tree kernels suitable for such structures. We evaluate the methods experimentally on a corpus containing the DOM trees of newspaper articles, and on the well-known SUSANNE corpus. We will demonstrate that, for the two corpora, many text types can be learned based on structural features only.

1 Introduction

In the emerging new field of so-called web corpora one is interested in classifying documents according to their genre or type [1]. For determining the type of a text, not only the occurring words play a role, but it is to a large extend determined by its visual and organizational structure, the latter of which can, in the case of XML and HTML documents, be captured by their DOM trees (Document Object Model), or, more generally speaking, by a tree- or graph-like representation of the logical document structure (LDS).

In this paper, we restrict consideration to LDSs represented by trees. In order to solve the corresponding classification task, we would like to apply the support vector machine (SVM) in order to achieve good results. We will therefore outline a method for describing the trees using predefined features suitable for the structure-based classification of texts. This results in a flat representation that can be combined with standard kernels, e.g., polynomial or RBF. This methods will be called Quantitative Structure Analysis (QSA).

A related approach consists in using so-called tree kernels that take complex substructures into account. An example is the *parse tree kernel* [2], which is applicable to parse trees of sentences. The parse tree kernel is an instance of the class of so-called convolution kernels. In this paper we will describe variants of the parse tree kernel that are applicable to the DOM trees of XML/HTML documents or to a tree representation of the LDS of a document.

The rest of this paper is structured as follows. QSA is defined in Section 2. Our tree kernels are defined in section 3, followed by an experimental evaluation in section 4. The conclusions can be found in section 5.

2 Quantitative Structure Analysis (QSA)

The general idea behind QSA is to utilize quantitative indicators of text structure in order to thematically discriminate input texts. This approach is based on the correlation hypothesis of Biber [3] who argues that situation types vary by their linguistic instances. In the experiments described in section 4, we will test how far we can reach when identifying the genre of input texts by looking only at their expression plane.

Given a corpus of texts for which structural information is available in terms of their *logical document structure*[1], we represent each text as a vector of structure features (e.g., by the their mean sentence length) and apply SVM-based supervised learning in order to classify the texts subsequently.[2] Thus, we represent any input text as a *bag-of-features* where each vector is composed of features of various levels. **Structure Level:** S is the set of constituent types of logical document structure as, e.g., sentence, paragraph, phrase. Every structure type $s_i \in S$ is then represented in terms of quantitative characteristics of its tree structure [5]. **Feature:** Each input structure level $s_i \in S$ is described with respect to a set of features F_j. F_j may represent, for example, the *complexity* (i.e., the number of immediate daughter elements) of or the *length* (i.e., the number of leafs dominated by) a corresponding instance of s_i. For any input text x and some feature F_j, this gives a separate vector whose coefficients represent the instances of s_i ordered according to their presence in input text x. Next, these vectors are mapped onto single numbers per feature F_j. **Characteristic:** This is done by means of some parameters of location or statistical spread. In order to compute these parameters the input vectors are conceived as feature value distributions. For each input text and a given set of features $\{F_j\}$ this gives a separate feature value vector which is finally input to text categorization — cf. Mehler et al. [1] for a detailed description of this approach.

3 Tree Kernels

Collins and Duffy [2] defined a tree kernel for parse trees of natural language sentences, in which non-leaf nodes are labeled with the non-terminal of the node, and leaves with single words. The production applied to a non-leaf node determines the number, type, and ordering of the child nodes. Note that this is not the case for DOM trees, in which children might haven been modified, permuted, inserted, or deleted.

Collins and Duffy showed that, for two trees T and T', the kernel value $k(T, T')$ can be computed efficiently by determining the number of possible

[1] In this study we use a 10 years newspaper corpus of *Süddeutsche Zeitung* (SZ) with logical document structure assigned in a preliminary step using the TextMiner system [4].

[2] For calculating the input features, we assume that a DOM-tree is the underlying text representation model which consists of nested paragraphs, sentences, phrases etc.

mappings of isomorphic partial (incomplete) parse trees, $\Delta(v, v')$, rooted in nodes v and v' (excluding such consisting of a single node only) resulting in $k(T, T') = \sum_{v \in V, v' \in V'} \Delta(v, v')$.

The Δ-function can be computed recursively by setting $\Delta(v, v') = 0$ for *any* words and if the productions applied in v and v' are different. If the productions in v and v' are identical and both nodes are pre-terminals, Collins and Duffy set $\Delta(v, v') = 1$. For other non-terminals with identical productions, they use the recursive definition $\Delta(v, v') = \prod_{i=1}^{n(v)} (1 + \Delta(v_i, v'_i))$, where v_i is the i-th child of v, and v'_i is the i-th child of v'. $n(v)$ denotes the number of children of v (here corresponding to that of v'). It is possible to weight deeper trees using a factor $\lambda \geq 0$.

It can be shown that it is possible to just multiply Δ with $k^{\Sigma}(\alpha(v), \alpha(v'))$, where k^{Σ} is a kernel operating on node labels, and α and α' are the labeling functions of the trees.

The Left-Aligned Tree Kernel (LeftTK) is a relatively straightforward extension of the parse tree kernel, based on isomorphic subtrees, in which the right-most children of a node are allowed to be missing. Its basic idea is thus to compare just as many children as possible using the given order $\leq$ on the child nodes if the number of children differs. This is achived by redefining the recursive part of the Δ-function by

$$\Delta(v, v') = \lambda \cdot k^{\Sigma}(\alpha(v), \alpha(v')) \Big(1 + \sum_{k=1}^{\min(n(v), n'(v'))} \prod_{i=1}^{k} \Delta(v_i, v'_i) \Big) . \tag{1}$$

The complexity of computing the kernel is $O(|V| \cdot |V'| \cdot b)$, where V and V' are the node sets of the two trees, and b is the maximum branching factor. The remaining kernel to be defined in the following all have a higher complexity of $O(|V| \cdot |V'| \cdot b^2)$, with an additional factor for the string tree kernel.

The LeftTK does not allow the child trees of v and v' to be permuted without a high loss in similarity as measured by the kernel value $k(T, T')$. This behavior can be improved, however, by considering the child tree sequences as *sets* and applying a so-called set kernel to them, which is also an instance of the convolution kernel. The corresponding definition of Δ is obtained by replacing the sum-product with $\sum_{i=1}^{n(v)} \sum_{i'=1}^{n'(v')} \Delta(v_i, v'_{i'})$, i.e., all possible pairwise combinations of child trees are considered.

Since this pairwise comparison is ignorant of the order of the child trees in the child sequences of v and v', we cannot distinguish the trees that differ only in the ordering of the children. This can, however, be achieved by multiply the SetTK with a factor $k_{\gamma}(\mu(v), \mu'(v'))$, where μ (and μ', resp.) give the position in the child sequence (defined as 1 for the roots), and k_{γ} is a kernel operating on positions, e.g., $k_{\gamma}(x, y) = e^{-\gamma(x-y)^2}$ for node positions x and y. The maximum value is attained for $x = y$. γ is a parameter to be set by the user. It determines how different the positions are allowed to be. This kernel will be called soft tree kernel (SoftTK).

The string tree kernel (StringTK) results from applying a string kernel to the sequence of child trees. In contrast to the work of Kashima and Koyanagi [6] and Alessandro Moschitti [7], we allow gaps, complex node labels and use a different, more suitable recursive computation.

4 Experiments

We applied our techniques to the corpus described in Section 2. QSA could be applied to the whole dataset. For applying the tree kernels, however, we had to downsample the dataset originally containing about 35000 examples, resulting in a set with 6250 examples. Still, we were not able to apply the StringTK, and SetTK and SoftTK yielded bad results. Among the tree kernels, the LeftTK performed best.

Fig. 1 presents the results of the experiment for QSA and LeftTK. Every category is identified by an abbreviation representing a category (e.g. `woch2` = *'Wochenchronik'* 'chronicle of the weak' etc.). The corresponding *F*-Score values demonstrate the separability of most of the categories. Although LeftTK performs better for a few classes, it is often outperformed by QSA.

The SUSANNE Corpus [8] consists of 64 files which are annotated version of texts from the Brown corpus, each of which contains more than 2000 words. Sixteen texts are drawn from each of the following genre categories: A (press reportage), G (belles lettres, biography, memoirs), J (learned, mainly scientific and technical, writing), N (adventure and Western fiction). Because we are inter-

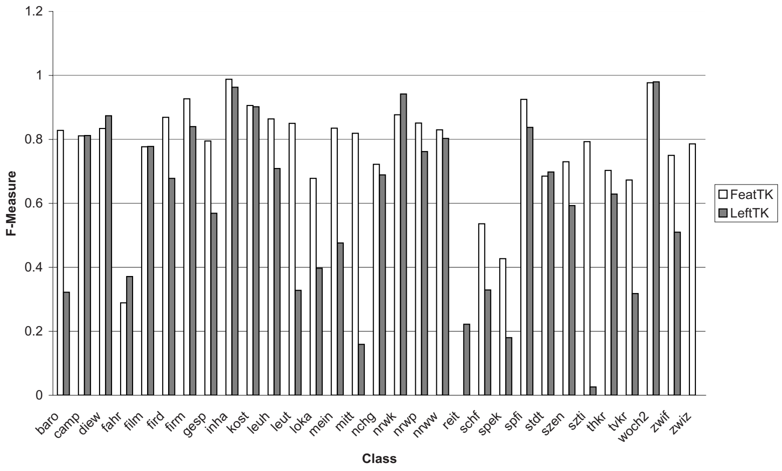


Fig. 1. Categorization experiment of 31 categories of SZ ordered alphabetically: optimal F-measures (cross validation) for the binary classification problems (class vs. rest)

Table 1. Results for Susanne Corpus: F-Measures

Cat.	LeftTK	-N	SetTK	-N	SoftTK	-N	StringTK	-N	QSA
A	0.538	0.4	0.97	0.968	1.0	1.0	1.0	1.0	1.0
G	0.405	0.367	0.733	0.89	0.733	0.89	0.687	0.727	0.86
J	0.688	NAN	0.903	0.903	0.903	0.953	0.8	0.903	0.97
N	0.4	NAN	0.97	1.0	0.97	1.0	0.967	0.967	1.0

ested in the structure based classification, we removed all information on specific words and kept only a simplified version of the parsed text and sentence structure, with tags like N (noun phrase), V (verb phrase), etc., in which the specific tag Y was used to denote the occurrence of **some** word. Interestingly enough, we could still obtain very good classification results, see Table 1.

In Table 1, the estimated optimal F-measure for the four categories are shown (using the leave-one-out method). For every type of kernel (see above), we considered the original definition and the normalized version given. Note that for large values of γ, SoftTK shows the same results like SetTK.

It can be seen, that the classes A, J, N can be learned quite well with any of the approaches, except LeftTK. For class G, however, normalized SetTK and (normalized) SoftTK perform best. QSA performs best for class J.

5 Conclusions

The most important result in terms of computational linguistics is that, seemingly, the type of a text can often be determined based on structural features only. The evaluation moreover showed that complexity is a critical issue, particularly for large corpora containing complex trees.

References

1. Mehler, A., Geibel, P., Pustylnikov, O., Herold, S.: Structural classifiers of text types. LDV Forum (to appear, 2007)
2. Collins, M., Duffy, N.: Convolution kernels for natural language. In: NIPS, pp. 625–632 (2001)
3. Biber, D.: Dimensions of Register Variation. A Cross-Linguistic Comparison. Cambridge University Press, Cambridge (1995)
4. Mehler, A.: Hierarchical orderings of textual units. In: COLING 2002. Proc. of the 19th International Conference on Computational Linguistics, pp. 646–652. Morgan Kaufmann, San Francisco (2002)
5. Köhler, R.: Syntactic Structures: Properties and Interrelations. Journal of Quantitative Linguistics, 46–47 (1999)
6. Kashima, H., Koyanagi, T.: Kernels for semi-structured data. In: ICML, pp. 291–298 (2002)
7. Moschitti, A.: Efficient convolution kernels for dependency and constituent syntactic trees. In: Fürnkranz, J., Scheffer, T., Spiliopoulou, M. (eds.) ECML 2006. LNCS (LNAI), vol. 4212, pp. 318–329. Springer, Heidelberg (2006)
8. Sampson, G.: English for the Computer: The Susanne Corpus and Analytic Scheme: SUSANNE Corpus and Analytic Scheme. Clarendon Press, Oxford (1995)

A Comparison of Neural-Based Techniques Investigating Rotational Invariance for Upright People Detection in Low Resolution Imagery

Steve Green and Michael Blumenstein

Griffith University, Gold Coast, Queensland 4222, Australia
{s.green, m.blumenstein}@griffith.edu.au

Abstract. This paper describes a neural-based technique for detecting upright persons in low-resolution beach imagery in order to predict trends of tourist activities at beach sites. The proposed system uses a structural feature extraction technique to represent objects of interest for training a selection of neural classifiers. A number of neural-based classifiers are compared in this study and a direction-based feature extraction technique is investigated in conjunction with a rotationally invariant procedure for the purpose of beach object classification. Encouraging results are presented for person detection using video imagery collected from a beach site on the coast of Australia.

1 Introduction

The study of local beach behaviour can provide valuable information for local authorities to estimate the number of persons using a beach on a particular day, and could also notify lifeguards when a swimmer is at risk in a hazardous beach environment [1]. A number of World Wide Web cameras (also called web cams) have been established to provide local beach information on surf and current weather conditions at many beach locations around Australia. Coastalwatch[1] Pty. Ltd. provides one such service. The streamed images from Coastalwatch web cams are of a low resolution. Therefore, aside from the inherent difficulties in dealing with variable outdoor imagery, processing low quality images provides quite a challenge for any automated human detection system.

2 Overview of Existing Techniques

The automated detection of persons and their behaviour in beach scenes is a novel application in the field of video surveillance. However, a number of techniques and systems have been proposed for automated analysis of humans in other indoor and outdoor situations. Some of these are reviewed and detailed in the paragraphs that follow.

[1] http://www.coastalwatch.com

Iketani *et al.* [2] propose a system for real-time detection of intruders (persons) in difficult, outdoor and cluttered scenes using information from video imagery over space and time. Sacchi *et al.* [3] present advanced image processing tools for remote monitoring of a tourist site involving the counting of persons in outdoor scenes. Bartolini *et al.* [4] propose a system for automatically counting persons getting on and off a bus using image sequence analysis for allocating appropriate resources on bus lines. Pai *et al.* [5] present a system for pedestrian tracking using vision-based techniques to prevent traffic accidents.

3 System Overview

This research describes a model for the automatic detection of people on beaches using a hybrid exhaustive search strategy. The proposed system uses a classification based approach, which searches grey-scale images of beach scenes for potential objects of interest. The sub-images extracted are then processed to determine whether a person object has been found.

3.1 Object Detection and Segmentation

The first important stage for segmenting objects out of a complex scene is to detect the object boundaries in the image. Boundaries encompass an object and are regions where there is a high change in luminance values. Beach scenes mainly contain large areas of sand and sea. From one point of view, this is of great benefit when detecting people, as people in beach scenes contrast quite significantly with the sand and therefore are easier to detect, than say, in a forested scene. However, conversely, the presence of the ocean, constant wave motion and shadowing effects can make the process a challenging one.

Two different approaches to object segmentation have been carried out; the first approach used a background extraction technique to segment foreground objects, and the second approach utilized the benefits of the first technique but subsequently applied an exhaustive search to regions of interest (ROI) [6].

3.2 Modified Direction Feature (MDF)

MDF has been detailed elsewhere [7] and will only be briefly described here. MDF feature vector creation is based on the Location of Transitions (LTs) from background to foreground pixels in the vertical and horizontal directions of an object represented by the object's boundary edges These LT values are coupled with Direction Transitions (DTs) calculated based on the shape of the object's boundary.

3.3 Rotationally Invariant Experiments with MDF

Currently, MDF examines both foreground transitions and direction values to build a feature vector. Foreground transitions are calculated by searching from all four sides of an image. Experiments were conducted to examine whether MDF could be improved (in the domain of low resolution beach imagery) by extending it to be rotationally invariant. As a first step towards conducting experiments with a rotationally invariant version of MDF, images were rotated individually before being presented to MDF. To make MDF rotationally invariant, foreground and direction transition values could be calculated from a maximum distance on the object boundary to the object's centroid. As there can be several maximum points, a method would be required to find the same maximum point on an object regardless of the object's rotation. With the low quality beach data used in this research, beach object boundaries are often incomplete and have holes within the boundary.

Therefore it was decided that the best initial approach was to conduct experiments on rotating beach objects vertically based on their principal axis and presenting the image to MDF to obtain a feature vector created by employing operations to simulate rotational invariance. In this research, four classifiers were tested for classifying objects into person and non-person categories.

3.4 Neural-Based Classification

The classifiers were chosen to give a good cross-section of popular classification techniques. The four classifiers tested were: the Multilayer Perceptron (MLP), a Support Vector Machine (SVM) [9], a Radial-Basis Function (RBF) network, and a Bayesian Network. The Weka 3 data mining software written in Java was used to conduct all experiments on the above classifiers [8]. All classifiers were trained with images processed by the MDF extraction technique.

4 Results

4.1 Image Database

To test the four different classifiers, and the person detection system on beach scenes using the MDF feature extraction technique, a database was created including people and non-people objects. To facilitate accurate classification, it was necessary to create a beach image database that contained images of people and non-people in all possible situations to obtain a representative data set. The image database consisted of 2576 images of varying size. These images were extracted predominately from recordings taken from a Coastalwatch web camera at Surfers Paradise beach. Frames were extracted from the beach recordings, and image regions were classified by a human operator into two categories; "person" and "non-person". The image database was then broken into three sets to facilitate three-fold cross validation, whereby each set was composed of 2000 training images and 576 test images.

4.2 Classifier Settings

As previously mentioned, the image database was broken into separate sets to perform three-fold cross validation. MLP training was conducted using 8, 12, 16, 20, 24, and 28 hidden units, and 500 epochs for training. The SVM training incorporated both the polynomial and RBF kernels, varying the exponent and gamma settings respectively. The RBF Network was trained using 2 clusters (basis functions), and a minimum standard deviation of 0.1 for the clusters. The Bayesian network was trained using the basic search algorithm and simple estimator configuration [8]. Table 1 shows the results for all three sets, and the average of the three sets for each classifier.

4.3 Classification Results

In this section, results for the classification of person objects are presented in tabular form. Table 1 presents the results obtained using the four classifiers for processing the test image sets. The MLP and SVM parameters for training were modified to optimise the training result. The test set classification rate was calculated based upon the number of successfully recognised person objects (True Positives (TP)), and the number of "non-person" items incorrectly labelled as person objects (False Positives (FP)). The results in table 1 show the best overall results for both TP and FP for each classifier.

Table 1. Top Classification Rates Using the MDF Extraction Technique (True Positive TP, False Positive FP)

	Set01		Set02		Set03		Average	
	Person TP	Other FP	Person TP	Other FP	Person TP	Other FP	Person TP	Other FP
MLP								
24 Hidden Units	93.06	13.19	88.89	8.33	86.46	11.11	89.47	10.88
RBF Network	90.97	22.57	89.93	17.01	87.50	17.71	89.47	19.10
SVM Polynomial Kernel								
Exponent = 2	90.63	10.76	89.93	6.94	92.71	11.11	91.09	9.60
SVM RBF Kernel								
Gamma = 0.300	94.10	12.15	89.93	6.94	91.32	6.94	91.78	8.68
Bayesian Network	88.89	16.32	89.58	16.67	90.28	17.36	89.58	16.78

4.4 Classification Result Discussion

Table 1 shows that the MLP and SVM gave the best results overall, with the MLP's overall best result achieving 93.06 whilst the SVM using an RBF kernel produced an overall best result of 91.32 and 6.94 kernel gave better results for FP classification. The test results for all classifiers show a definite correlation between achieving a high person TP classification rate and a higher FP rate for other beach objects'. Further observation of the data found that of those people who were not identified correctly, the errors were mainly due to: 1) the person was very small, sometimes only 20 pixels tall, 2) the person was obscured by the surf or 3) a shadowing effect produced an incorrect extraction of the person object's boundary. In the case of false positives, objects that were classified as

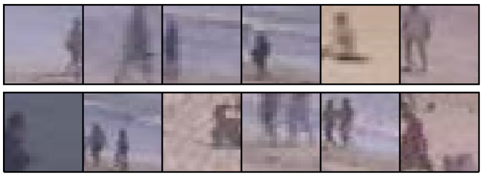


Fig. 1. Person detection results showing true negative (top row), and false positives (bottom row); both sets of images are not to scale

Table 2. Results Using Rotation (True Positive TP, False Positive FP)

	Set01		Set02		Set03		Average	
	Person TP	Other FP	Person TP	Other FP	Person TP	Other FP	Person TP	Other FP
Test 1 MLP								
Hidden Units								
16	93.06	13.54	88.54	9.38	86.81	12.15	89.47	11.69
20	91.32	12.5	89.58	9.03	89.93	12.15	90.28	11.23
24	93.06	13.19	88.89	8.33	86.46	11.11	89.47	10.88
Test 2 MLP-R								
Hidden Units								
16	72.57	16.32	80.56	25.35	80.21	26.04	77.78	22.57
20	68.75	15.97	78.47	23.61	74.65	27.78	73.96	22.45
24	71.53	15.97	78.13	25.69	81.25	19.1	76.97	20.25
Test 3 MLP-R (People Only)								
Hidden Units								
16	88.89	10.42	92.71	7.99	93.06	12.50	91.55	10.30
20	88.54	10.42	95.94	9.03	91.32	13.54	91.93	11.00
24	88.54	9.03	95.49	7.64	92.36	11.11	92.13	9.26

a person when the object was a non-person, tended to have an outline that was upright (a common 'person-type' shape), causing an incorrect classification (see Figure 1).

4.5 Results with Rotation

Table 2 shows the results for the rotational invariant experiments, where images were rotated so that the object's principal axis was vertical before extracting the feature vector with MDF (see Figure 2).

The table shows three test results; test 1 shows a result with no rotation, test 2 shows results using a rotation operation, and test 3 shows the results with only person objects being rotated. Test 2 gave very poor classification results compared to test 1 where images are not rotated. This would seem to be because non-person objects (mainly noise) when rotated vertically confused the neural network follwing training, giving a lower classification result for both TP and

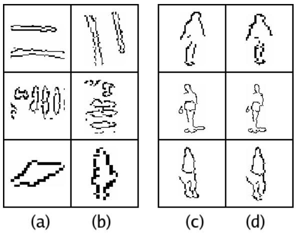


Fig. 2. Rotated beach objects where (a) and (c) are the original images, and (b) and (d) are after rotation. In column (a) the top image shows wave noise, the centre image shows a group of people, and the bottom image displays noise created by shadowing on the sand (not to scale).

FP results. In Figure 2 the last images in columns (a) and (b) highlight the problem of beach noise, that when rotated can look very similar to a person boundary, and therefore makes it difficult for the neural network to classify person and non-person objects correctly. It is clear, that in identifying upright people in low-resolution imagery where ROI are only a few pixels tall, that the main feature the classifier uses to differentiate between upright person and non-person objects is their height to width ratio. From this observation in test 3, only person objects were rotated in the training and test sets, and the result showed some improvement for both true positive and false positive classification. Again, the reason for this would seem to be a combination of 1) the low-resolution images and the way that MDF acquires feature values, and 2) non-person objects, when rotated, can look very similar to person object boundaries. For the first point, low-resolution images contain beach objects that in most cases are only 20 to 80 pixels in height. This means that once the boundary of the object is acquired there is very little to distinguish between person and other objects. Empirical studies would strongly suggest that person and other beach objects are classified mainly based on their rotation i.e. vertical or horizontal. Person objects are predominately vertical and other objects are predominately horizontal. The results in Table 2 seem to support this theory, as the classification results in test 3 demonstrate an improvement when person objects are rotated vertically.

5 Conclusions and Future Research

In this paper, an upright person detection system for quantifying people in beach scenes has been described. The system incorporated a neural-based classification system for the detection of person and non-person objects. Encouraging results were presented for automated person detection and quantification, which can be applied to beach scene analysis for surveillance and coastal management applications. A number of classifiers were investigated and compared for the purpose of beach object classification. The best results were obtained when using an MLP trained with the backpropagation algorithm and an SVM-based classifier. The features described in this paper were initially processed using a rotational operation to simulate rotational invariance. The results obtained are encouraging when person objects are rotated vertically on their principal axis. Future experiments will address the issue of horizontal rotation and incorporation of rotational invariance in MDF. Further work will also examine different object segmentation techniques. Finally, frames retrieved from beach media are at present processed individually. Future research will look at generating some confidence values for a person object based on temporal information from past frames.

Acknowledgments

The authors would like to thank Chris Lane from Coastalwatch and Professor Rodger Tomlinson from the Griffith Centre for Coastal Management for their support on this project.

References

1. Browne, M., Blumenstein, M., Tomlinson, R.: LaneC.: An intelligent system for remote monitoring and prediction of beach conditions. In: Proceedings of the International Conference on Artificial Intelligence and Applications, pp. 533–537. Innsbruck (2005)
2. Iketani, A., Nagai, A., Kuno, Y., Shirai, Y.: Real-Time Surveillance System Detecting Persons in Complex Scenes. Real-Time Imaging 7, 433–446 (2001)
3. Sacchi, C., Gera, G., Marcenaro, L., Regazzoni, C.S.: Advanced image-processing tools for counting people in tourist site-monitoring applications. Processing 81, 1017–1040 (2001)
4. Bartolini, F., Cappellini, V., Mecocci, A.: Counting people getting in and out of a bus by real-time imagesequence processing. Image and Vision Computing 12, 36–41 (1994)
5. Pai, C-J., Tyan, H-R., Liang, Y-M., Liao, H-Y.M, Chen, S-W.: Pedestrian detection and tracking at crossroads. Pattern Recognition 37, 1025–1034 (2004)
6. Green, S., Blumenstein, M.: An exhaustive search strategy for detecting persons in beach scenes using digital video imagery and neural network-based classification. In: IEEE World congress on computational intelligence, Vancouver, Canada, pp. 2439–2444 (2006)
7. Blumenstein, M., Liu, X.Y., Verma, B.: A modified direction feature for cursive character recognition. In: IJCNN 2004. Proceedings of the International Joint Conference on Neural Networks, Budapest, Hungary, pp. 2983–2987 (2004)
8. Witten, I.H., Frank, E., Woods, R.E.: Data mining: practical machine learning tools and techniques, 2nd edn. Morgan Kaufmann, San Francisco (2005)
9. Platt, J.: Sequential minimal optimization: A fast algorithm for training support vector machines. Technical Report Microsoft Research Redmond Washington, pp.98–14 (1998)

Multilevel Thresholding Method for Image Segmentation Based on an Adaptive Particle Swarm Optimization Algorithm

Chonghui Guo and Hong Li

Institute of Systems Engineering, Dalian University of Technology
116024 Dalian, P.R. China
guochonghui@tsinghua.org.cn, snowerlh@yahoo.com.cn

Abstract. The multilevel thresholding method with maximum entropy is one of the most important image segmentation methods in image processing. However, its time-consuming computation is often an obstacle in real time application systems. Particle swarm optimization (PSO) algorithm is a class of heuristic global optimization algorithms which appeared recently. In this paper, the maximum entropy is obtained through an adaptive particle swarm optimization (APSO) algorithm. The APSO algorithm is shown to obtain the maximum entropy of multilevel thresholding effectively on experiments of image segmentation.

1 Introduction

Thresholding is a popular tool for image segmentation which is essentially a pixels classification problem which can be classified into two groups: global and local [5]. Global thresholding methods select one threshold value for the entire image based on different criteria, such as minimum error thresholding [6]and Otsu's method [7]. Entropy is a useful criterion in communications [8]. It was firstly introduced into thresholding by Pun [9] and then many methods based on entropy are developed. Local thresholding methods select different threshold values for different regions, or even for each pixel [10]. To segment complex images, a multilevel thresholding method is required. However, its time-consuming computation is often an obstacle in real time application systems.

The particle swarm optimization algorithm (PSO) has been proven to be a powerful competitor to other heuristic algorithms, such as genetic algorithm, tabu search and simulated annealing algorithm for global optimization problems [11, 12]. Clerc improved the standard PSO algorithm by introducing a constriction factor (CPSO) [13]. Du et al. segmented infrared image with 2-D maximum entropy based on PSO [14].

In this paper, considering the complexity of multilevel thresholding method with maximum entropy, we take use of an adaptive particle swarm optimization (APSO) algorithm [15], which introduces two adaptive acceleration factors and a new weight function in terms of the convergence speed and global search capability of the PSO algorithm, to search the maximum entropy of image segmentation.

2 Maximum Entropy Thresholding

In image segmentation, the most commonly method is to segment the image into two parts by one threshold. For an image with n gray-levels, let $p_1, p_2, \cdots, p_n$ be the probability distribution of the levels. From the distribution we derive two probability distributions given a threshold value t, one for the object A_1 and another for the background A_2. The probability distributions of the object A_1 and background A_2 are given by

$$A_1: \frac{p_1}{P_{A_1}}, \frac{p_2}{P_{A_1}}, \cdots, \frac{p_t}{P_{A_1}}, \qquad A_2: \frac{p_{t+1}}{P_{A_2}}, \frac{p_{t+2}}{P_{A_2}}, \cdots, \frac{p_n}{P_{A_2}}, \tag{1}$$

where $P_{A_1} = \sum_{i=1}^{t} p_i$, $P_{A_2} = \sum_{i=t+1}^{n} p_i$.

The Shannon entropy for each distribution is defined as

$$H(A_1) = -\sum_{i=1}^{t} \frac{p_i}{P_{A_1}} \log \frac{p_i}{P_{A_1}}, \quad H(A_2) = -\sum_{i=t+1}^{n} \frac{p_i}{P_{A_2}} \log \frac{p_i}{P_{A_2}}. \tag{2}$$

Let $\varphi(t) = H(A_1) + H(A_2)$. When $\varphi(t)$ is maximized the luminance level t is considered to be the optimal threshold value. This can be achieved with a cheap computational effort.

$$t^* = \arg\max_t (\varphi(t)). \tag{3}$$

It is not difficult to extend this principle to multilevel thresholding. Let

$$\varphi(t_1, t_2, \cdots, t_k) = H(A_1) + H(A_2) + \cdots + H(A_k), \tag{4}$$

where k is the number of the thresholds which segment the image into $k+1$ parts, and $A_j = \sum_{i=t_{j-1}+1}^{t_j} p_i$ .Then the optimal threshold vector can be described as

$$\left(t_1^*, t_2^*, \cdots, t_k^*\right) = \arg \max_{(t_1, t_2, \cdots, t_k)} (\varphi(t_1, t_2, \cdots, t_k)). \tag{5}$$

In this paper, we use the multilevel thresholding to segment images.

3 APSO Algorithm

Let i indicate a particle index in the swarm that includes m particles. Each particle flies through an n-dimensional search space. Let $X_i = (x_{i1}, x_{i2}, \cdots, x_{in})$ and

$V_i = (v_{i1}, v_{i2}, \cdots, v_{in})$ represent the current position and velocity of particle *i* respectively. They are dynamically adjusted according to the particle's previous best position $P_i = (p_{i1}, p_{i2}, \cdots, p_{in})$ and the best position of the entire swarm $P_g = (p_{g1}, p_{g2}, \cdots, p_{gn})$. In APSO [15], the evolving equations of particles can be described as follows:

$$v_{ij}(t+1) = w(t)v_{ij}(t) + w_1 r_{1j} A_{i1}(p_{ij}(t) - x_{ij}(t)) + w_2 r_{2j} A_{i2}(p_{gj}(t) - x_{ij}(t)) \ , \tag{6}$$

$$x_{ij}(t+1) = x_{ij}(t) + v_{ij}(t+1) \qquad i = 1, 2, \cdots, m; \ j = 1, 2, \cdots, n \ , \tag{7}$$

where r_{1j}, $r_{2j} \in [0,1]$ are random numbers. The weight function *w*(*t*) ~ *U*(*w*$_0$, 2*w*$_0$), where w_0 is a constant parameter, the suggested range for w_0 is [0.3, 0.6], and *U* presents a uniform probability distribution. w_1 and w_2 are as shown in (8), and the two adaptive acceleration factors A_{i1} and A_{i2} are as shown in (9) respectively.

$$w_1 = c_1^2 / (c_1 + c_2) \ , \ w_2 = c_2^2 / (c_1 + c_2) \tag{8}$$

where c_1 and c_2 are the same as in the standard PSO.

$$A_{i1} = \ln\left(\left|\frac{f(P_i(t)) - f(X_i(t))}{\overline{PF}}\right| + e - 1\right), \ A_{i2} = \ln\left(\left|\frac{f(P_g(t)) - f(X_i(t))}{\overline{GF}}\right| + e - 1\right) \tag{9}$$

where *f(x)* is the fitness function, *e* is the base of the natural logarithm, i.e. *ln(e)*=1, $\overline{PF} = \frac{1}{m}\sum_{i=1}^{m}(f(P_i(t)) - f(X_i(t)))$, and $\overline{GF} = \frac{1}{m}\sum_{i=1}^{m}(f(P_g(t)) - f(X_i(t)))$.

4 Image Segmentation Based on APSO Algorithm

Two images are segmented in the experiments to illustrate the validation of APSO algorithm. The original images Lena and Fruit contain 256×256 pixels. After getting the gray histogram of two images, we set the number of threshold *k*=3 for each image. The optimal threshold vector of Lena is (61, 109, 159) with the maximum Shannon entropy value 15.6632, and the optimal threshold vector of Fruit is (57, 111, 166) with the maximum Shannon entropy value 15.9421. That can be obtained by exhaustive search which needs to do $256 \times 255 \times 254 = 16{,}581{,}120$ times computation of entropy function.

In both experiments, standard PSO, CPSO and APSO algorithm are used to search the best thresholding. For the standard PSO algorithm, a linearly decreasing weight function is used which starts at 0.9 and ends at 0.4 with c_1=c_2=2.0. For CPSO and APSO algorithm, we set c_1=2.8, c_2=1.3 under the suggestion of Clerc [13]. For all algorithms we set the initial particles m=30 and the maximum generation $G_{\max} = 10000$. The final result takes from the mean value of 50 runs for each algorithm. All the algorithms are written in C and executed in Visual C++ 6.0 on PC with a CPU of Pentium Ⅳ and 256M RAM. Table 1 shows the results of experiments.

Table 1. The comparison of efficiency between Exhaustive search, PSO, CPSO and APSO

		Algorithm			
		Exhaustive search	PSO	CPSO	APSO
Lena	Time-consuming (ms)	406,609	12,055	6,232	1,103
	Calls for entropy function	16,581,120	45,472	22,168	921
	Success rate	—	100%	100%	100%
Fruit	Time-consuming (ms)	656,296	13,918	6,548	1,067
	Calls for entropy function	16,581,120	62,736	29,673	731
	Success rate	—	100%	100%	100%

From Table 1, we can see it clearly that APSO algorithm performances much better than standard PSO, CPSO algorithm and exhaustive search. For example, the time-consuming of PSO for Lena is 12,055ms, comparing with exhaustive search method 606,609ms, the efficiency is improved by $606,609/12,055 \approx 50.32$ times; The time-consuming of CPSO for Lena is 6,232ms, comparing with exhaustive search method 606,609ms, the efficiency is improved by $606,609/6,232 \approx 97.34$ times; and the time-consuming of APSO for Lena is 1,103ms, comparing with exhaustive search method 606,609ms, the efficiency is improved by $606,609/1,103 \approx 549.96$ times. Moreover, it is shown that PSO, CPSO and APSO can all reach the global optimal solution 100% within a smaller $G_{\max}$.

5 Conclusion

Multilevel thresholding with maximum entropy is a good method to do image segmentation except its complex computation. And PSO algorithm is a heuristic global optimization algorithm which appeared recently. PSO has been widely concerned by researcher because of its feasibility and effectiveness. In this paper, with the help of APSO algorithm, the maximum entropy vector can be obtained easily with high efficiency. The multilevel thresholding method for image segmentation based on APSO algorithm is a simple and effective method to do image segmentation. Especially, it is full of importance in the system where real time processing is needed.

Acknowledgements

This work was supported by Natural Science Foundation of China under grant No. 10571018.

References

1. Punam, K.S., Jayaram, K.U.: Optimum image thresholding via class uncertainty and region homogeneity. IEEE Trans. Pattern Anal. Mach. Intell. 23(7), 689–706 (2001)
2. Ma, W.Y., Manjunath, B.S.: Edge flow: A framework of boundary detection and image segmentation. In: IEEE Conference on Computer Vision and Pattern Recognition, Puerto Rico, pp. 744–749 (1997)
3. Xu, C.Y., Jerry, L.P.: Snakes, shapes, and gradient vector flow. IEEE Trans. Image Process 7(3), 359–369 (1998)
4. Deng, Y.N., Manjunath, B.S.: Unsupervised segmentation of color–texture regions in images and video. IEEE Trans. Pattern Anal. Mach. Intell. 23(8), 800–810 (2001)
5. Sahoo, P.K., Soltani, S., Wong, A.K.C.: A survey of thresholding techniques. Comput. Vis. Graph. Image Process 41(2), 233–260 (1988)
6. Kittle, J., Illingworth, J.: Minimum error thresholding. Pattern Recognit. 19(1), 41–47 (1986)
7. Otsu, N.: A threshold selection method from gray level histogram. IEEE Trans. Syst. Man, Cybern. 8(1), 62–66 (1979)
8. Shannon, C.E., Weaver, W.: The mathematical theory of communication. Univ. of Illinois Press, Urbana (1949)
9. Pun, T.: A new method for gray level picture thresholding using the entropy of the histogram. Signal Process 2(3), 223–237 (1980)
10. Yang, Y., Yan, H.: An adaptive logical method for binarization of degraded document images. Pattern Recognit. 33(5), 787–807 (2000)
11. Kennedy, J., Eberhart, R.C.: Particle swarm optimization. In: Proceedings of the 1995 IEEE International Conference on the Neural Networks, vol. 4, pp. 1942–1948. IEEE Service Center, Piscataway, NJ (1995)
12. Schutte, J.F., Groenwold, A.A.: A study of global optimization using particle swarms. Journal of Global Optimization 31(1), 93–108 (2005)
13. Clerc, M.: The swarm and the queen: Towards a deterministic and adaptive particle swarm optimization. In: The Congress of Evolutionary Computation, Washington DC, USA, pp. 1951–1957 (1999)
14. Du, F., Shi, W., Chen, L., Deng, Y., Zhu, Z.: Infrared image segmentation with 2-D maximum entropy method based on pratilcle swarm optimization. Pattern Recognition Letters 26(5), 597–603 (2005)
15. Guo, C., Li, H.: An adaptive particle swarm algorithm for global optimization. In: Proceedings of the 6th International Conference on Management, Wuhan, China, pp. 8–12. Science Press (2007)

Twin Kernel Embedding with Relaxed Constraints on Dimensionality Reduction for Structured Data

Yi Guo[1], Junbin Gao[2,⋆], and Paul W. Kwan[1]

[1] School of Science and Technology,
University of New England, Armidale, NSW 2351, Australia
{yguo4,kwan}@turing.une.edu.au

[2] School of Acc & Computer Science,
Charles Sturt University, Bathurst, NSW 2795, Australia
jbgao@csu.edu.au

Abstract. This paper proposes a new nonlinear dimensionality reduction algorithm called RCTKE for highly structured data. It is built on the original TKE by incorporating a mapping function into the objective functional of TKE as regularization terms where the mapping function can be learned from training data and be used for novel samples. The experimental results on highly structured data is used to verify the effectiveness of the algorithm.

1 Introduction

The target of dimensionality reduction is mainly to find the corresponding counterparts (or embeddings) of the input data with dimension D in a much lower dimensional space (usually Euclidean with dimension d and $d \ll D$) without incurring significant information loss. A number of nonlinear dimensionality reduction (NLDR) algorithms have been proposed recently. Twin Kernel Embedding (TKE) [1] is a powerful algorithm for highly structured data. However, it is unclear about how to handle novel samples in TKE. In this paper, we will address this problem. We first define a mapping function $f : \mathcal{Y} \rightarrow \mathcal{X}$ from the input space $\mathcal{Y}$ to the latent space $\mathcal{X}$ which is integrated into TKE as regularization terms. Then the mapping function will be learned by training the revised TKE. For new samples, it is convenient to use f to predict their embeddings in latent space. We call the mapping function relaxed constraints. To facilitate the explanation, we will give a brief introduction to TKE in next section followed by the details on incorporating the relaxed constraints into TKE. The experiments on texts and proteins which are recognized as highly structured data will be used to verify the effectiveness of this method and finally we conclude this paper in the last section.

⋆ The author to whom all the correspondences should be addressed.

2 Twin Kernel Embedding

Let $\{y_i\}_{i=1}^N$ be the structured input data in $\mathcal{Y}$ and $\{x_i\}_{i=1}^N$ their corresponding embeddings in $\mathcal{X}$. In Twin Kernel Embedding (TKE), the pairwise similarity between two input data is evaluated by a kernel function $k_y(\mathbf{y}_i, \mathbf{y}_j)$ where $i, j = 1 \ldots, N$. For the corresponding embeddings, another kernel $k_x(\mathbf{x}_i, \mathbf{x}_j)$ is adopted to measure the similarity. These two measures have been matched under the following objective functional to be minimized

$$L = -\sum_{ij} k_x(\mathbf{x}_i, \mathbf{x}_j) k_y(\mathbf{y}_i, \mathbf{y}_j) + \lambda_k \sum_{ij} k_x^2(\mathbf{x}_i, \mathbf{x}_j) + \frac{1}{2}\lambda_x \sum_i \mathbf{x}_i \mathbf{x}_i^\top \tag{1}$$

where λ_k and λ_x are tunable parameters to control the strength of the regularizations. The first term performs the matching mainly. It conveys that the $k_x(\mathbf{x}_i, \mathbf{x}_j)$ and $k_y(\mathbf{y}_i, \mathbf{y}_j)$ should be as similar as possible. Regularization terms make the optimization well-defined which will constrain the kernel values and the norm of the embeddings respectively. Minimizing (1) with respect to $\mathbf{x}_i$'s will lead to the optimal embeddings.

Equation (1) has no closed form solution basically. A gradient descent based optimization is applied to compute the optimal $\mathbf{X}$, the matrix formed by stacking all $\mathbf{x}_i$'s in rows. Normally, RBF kernel is selected for its simple derivative form (with respect to $\mathbf{X}$ and the kenrel hyperparameters as well) and its ability to capture nonlinearity.

3 TKE with Relaxed Constraints

It is clear from observing (1) that TKE can only find the optimal embeddings for currently presented data. To enable it to handle novel samples, we define a mapping function

$$f_j(\mathbf{y}_i) = \sum_m \alpha_{mj} k_y(\mathbf{y}_i, \mathbf{y}_m) \tag{2}$$

where α_{mj}'s are the coefficients to be determined. Let $x_{ij} = f_j(\mathbf{y}_i)$ and x_{ij} is the jth component of $\mathbf{x}_i$. We then revise the objective functional of TKE as

$$L = -\sum_{i,j} k_y(\mathbf{y}_i, \mathbf{y}_j) k_x(\mathbf{x}_i, \mathbf{x}_j) + \lambda_k \sum_{ij} k_x(\mathbf{x}_i, \mathbf{x}_j)^2 + \frac{1}{2}\sum_i \|\mathbf{x}_i - \mathbf{f}(\mathbf{y}_i)\|^2 + \frac{\lambda_\alpha}{2}\sum_{ij} \alpha_{ij}^2 \tag{3}$$

where $\mathbf{f}(\mathbf{y}_i) = [f_1(\mathbf{y}_i), \ldots, f_d(\mathbf{y}_i)]^\top$ and the coefficient λ_α governs the relative importance of the regularization term on α_{ij}'s. Let $\mathbf{A}$ be the matrix of all α_{ij}'s. Rewrite the objective function L in matrix form as

$$L = -\operatorname{tr}[\mathbf{K_x K_y}] + \lambda_k \|\mathbf{K_x}\|_F^2 + \frac{1}{2}\|\mathbf{X} - \mathbf{K_y A}\|_F^2 + \frac{\lambda_\alpha}{2}\|\mathbf{A}\|_F^2 \tag{4}$$

where $\| \cdot \|_F$ denotes the Frobenius norm of a matrix. To minimize (4) with respect to X and A with a gradient based algorithm, we need the derivatives as follows,

$$\frac{\partial L}{\partial \mathbf{X}} = \frac{\partial L}{\partial \mathbf{K_x}} \frac{\partial \mathbf{K_x}}{\partial X} + X, \text{ and, } \frac{\partial L}{\partial \mathbf{K_x}} = 2\lambda_k \mathbf{K_x} - \mathbf{K_y}. \tag{5}$$

$$\frac{\partial L}{\partial \mathbf{A}} = -\mathbf{K_y}\mathbf{X} + \mathbf{K_y}^2\mathbf{A} + \lambda_a \mathbf{A} \tag{6}$$

To start the optimization, the $\mathbf{X}$ will be initialized by KPCA or KLE [2]. After $\mathbf{X}$ is given, we use $\frac{\partial L}{\partial \mathbf{A}} = 0$ to get $\mathbf{A}$. So $\mathbf{A} = (\mathbf{K_y}^2 + \lambda_\alpha \mathbf{I})^{-1}\mathbf{K_y}\mathbf{X}^\top$. Because the relaxed constraints is integrated in TKE, we call this algorithm RCTKE.

4 Experimental Results

To demonstrate the effectiveness of the proposed RCTKE algorithms, the experiments of visualizing non-vectorial data (the target latent space is a normal plane and hence $d = 2$) were conducted on Reuters-21578 Text Categorization Test Collection[1] and SCOP (Structural Classification Of Protein) database[2] which are recognized as highly structured data.

4.1 Parameters Setting

Both RCTKE and TKE have parameters to be determined beforehand. Through performing a batch of experiments on the same data set varying only the parameters, we found these algorithms are not sensitive to the choice of the parameters. For TKE, we set $\lambda_k = 0.05$, $\lambda_x = 0.01$ and $k = 10$ in k nearest neighboring; for RCTKE, $\lambda_k = 0.01$ and $\lambda_\alpha = 1000$. The minimization will stop after 1000 iteration or when consecutive update of the objective is less than 10^{-7}. $k_x(\cdot,\cdot)$ is RBF kernel and initialization is done by KPCA for both RCTKE and TKE.

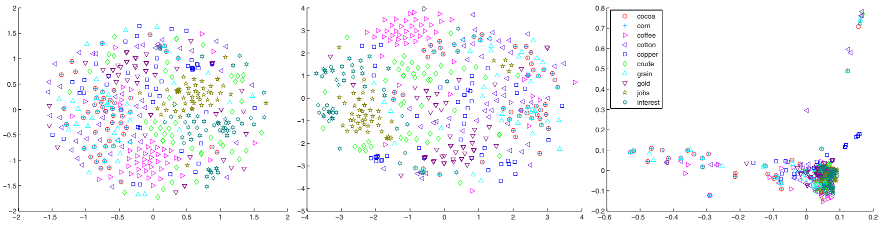
4.2 Reuters Texts

The data set consists of 10 arbitrarily selected topics and 500 pieces of news (50 pieces in each topic) extracted randomly from TOPICS category in Reuters-21578. The texts from the same topic are expected to be close in the 2D plane. The kernel used for input texts is the string subsequence kernel (SSK) [3] which is suitable for string objects (texts, DNA sequences etc). The results are presented in Figure 1 with the topics displayed as legend in (c).

The result of RCTKE and TKE reveal clear cluster structure complying with our expectation while that of KPCA is indistinguishable. An interesting observation is that the results of the texts from topics *cocoa* and *corn* are almost the same because the corresponding kernel values are very close or equal to 1. In other words, the underlying SSK does not significantly distinguish topics cocoa and corn. This also indicates the similarity preserving ability of RCTKE and TKE, namely, if two objects are similar in the sense of a given kernel, they

[1] Distribution 1.0 test collection which is available from http://www.daviddlewis.com/resources/testcollections/reuters21578

[2] SCOP data is available at http://scop.mrc-lmb.cam.ac.uk/scop/



(a) RCTKE **(b)** TKE **(c)** KPCA

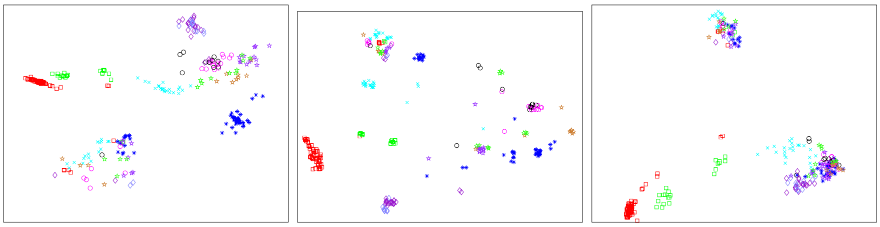
Fig. 1. The results on Reuters texts

will stay close together in the latent space. Compare the result of RCTKE with TKE, we observe that they are quite similar with clear clusters structure while the mapping function obtained in RCTKE which is ready for novel samples is absent in original TKE.

4.3 Proteins

Another promising application is in bioinformatics. Experiments were conducted on SCOP database. This database provides a detailed and comprehensive description of the structural and evolutionary relationships of the proteins of known structure. 292 proteins from different superfamilies and families are extracted for the test. The kernel for proteins is MAMOTH kernel which is from the family of the so-called alignment kernels whose thorough analysis can be found in [4]. Visualizing proteins on the 2D plane is of great importance to facilitate researchers to understand the biological meaning. The representation of proteins on the 2D plane should reflect the relational structure among proteins, that is, proteins having similar structures should be close while those with different structures should be far away.

The results are plotted in Figure 2. The result of KPCA is also presented for comparison. Each point represents a protein. The same shapes with the same colors are the proteins from same families while the same shapes with different colors represent the proteins from different families but from the same superfamilies. Both RCTKE and TKE reveal the fact that proteins from the



(a) RCTKE **(b)** TKE **(c)** KPCA

Fig. 2. The result of TKE and KPCA with MAMOTH kernel

same families congregate together as clusters while KPCA fails to reveal it. For example, almost all of the proteins from the globin family (black squares) gathered at certain area. Interestingly, they also reveal the truth that the proteins from the same superfamily but different families are similar in structure to some extent, which is reflected in the plane that the corresponding groups (families) are close if they are in the same superfamily (same shape). For instance, note that the proteins from ferritin and ribonucleotide reductase-like families (black diamonds and gray diamond respectively in the figures), they are close in the group level. This experiment also demonstrates that the RCTKE has comparable effect to original TKE. An attractive virtue is the mapping function learned from training provides a device for new samples.

5 Conclusion

In this paper, we incorporated a mapping function defined by kernel feature mapping into the objective functional of TKE as regularization terms and thus generated a new algorithm called RCTKE. Interestingly, though there are more variables to be optimized in RCTKE than those in TKE, it does not cost additional computation complexity. However, the mapping function embedded in the algorithm enables it to handle novel samples. Thus applications such as classification, pattern recognition would be possible based on RCTKE.

References

1. Guo, Y., Gao, J., Kwan, P.W.: Visualization of non-vectorial data using twin kernel embedding. In: International Workshop on Integrating AI and Data Mining, pp. 11–17 (2006)
2. Guo, Y., Gao, J., Kwan, P.W.: Kernel Laplacian Eigenmaps for Visualization of Non-vectorial Data. In: Sattar, A., Kang, B.H. (eds.) AI 2006. LNCS (LNAI), vol. 4304, pp. 1179–1183. Springer, Heidelberg (2006)
3. Lodhi, H., Saunders, C., Shawe-Taylor, J., Cristianini, N., Watkins, C.J.C.H.: Text classification using string kernels. Journal of Machine Learning Research 2, 419–444 (2002)
4. Qiu, J., Hue, M., Ben-Hur, A., Vert, J.P., Noble, W.S.: An alignment kernel for protein structures. Bioinformatics (to appear, 2007)

Evaluation of Connectives Acquisition in a Humanoid Robot Using Direct Physical Feedback

Dai Hasegawa, Rafal Rzepka, and Kenji Araki

Graduate School of Information Science and Technology,
Hokkaido University, Sapporo, Japan
{hasegawadai, kabura, araki}@media.eng.hokudai.ac.jp

Abstract. In this paper, we propose a method where humanoid robot acquires meanings of grammatical connectives using direct physical feedback from human. Our system acquired 70% connectives of all connectives taught by subjects. It can be also said that robot partially learned the concept of time.

1 Introduction

Humanoid robots should interact with humans using natural language and work adaptively in the same environment as human's. However, it is too difficult to suppose every environmint preliminarily. Therefore, algorithms where a robot will adaptively learn concepts of actions represented by natural languages are required.

Thus if we assume that infants and robots having less knowledge were analogical, we can also assume that Inductive Learning[1][2] should be efficient for the progress of robot's language acquisition, because infants learn inductively[3]. Furthermore, according to studies of so called "Embodied Intelligence"[4], embodied systems can easily acquire intelligence by mutual interaction with their environment[5]. Therefore we assumed that direct physical feedback (see Chapter 3) is also effecient for this problem.

As the first step, we start to acquire connectives. In this reserch, our target language is Japanease, and we will use *italic* when giving a Japanese examples.

2 Element of Action and Connectives

In this paper, we define Element of Action (EoA) as follows. EoA contains only one verb. EoA shows final physical position of action. We also call a command that meant EoA an **EoA command**. The final position of EoA is decided by average of direct physical feedback, by robot's users.

We define connectives as grammatical expressions which join two EoA commands. We also call a command that contains connectives a **connective command**. According to our definition of connectives, the connectives can also contain verbs to connect two EoAs.

3 Way of Teaching(Direct Physical Feedback)

A humanoid robot learns by being taught by human how to move its body. This becomes an universal and natural method which allows us to teach within an allowance of a humanoid robot's body structure. This approach helps such robot to reproduce movements with very high certainty.

4 System

4.1 Humanoid Robot

For our experiments, we use a humanoid robot KHR2-HV[6]. The robot is equipped with 17 motors, and it sends signals describing its states of motors only. It does not have any other sensors.

We use 7 motors of all motors which is upper half of the robot's body. The particular motors are abbreviated as follows. [H]:Head, [LS]:Left Shoulder, [LA]:Left Arm, [LH]:Left Hand, [RS]:Right Shoulder, [RA]:Right Arm, [RH]:Right Hand. All motors' 180 degrees movements were divided into 10 ranks, 18 degrees each.

4.2 System Flow

Our system algorithm is outlined below.

1. A user inputs a command in natural language (Japanese) from a keyboard.
2. The robot applies rules or uses examples that have been previously acquired, and performs an action.
3. If the robot performs a wrong action or does not perform any action, the user teaches it a proper action by direct physical feedback. And then the system adds a pair of command and action to the example database.
4. Then, the system generates abstracted rules from the example database by Inductive Learning process.

4.3 Representation of Action

An action is represented as a matrix shown in Figure 1. While a user moves the robot's body, it measures degrees of all the angles of motors and digitalize them every second. If a motor does not move, degree of its angle is described as "x". We do not use motor speed information yet and we plan to use it when acquiring adverbs.

4.4 Example

An example consists of two parts. One is a natural language command morphologicaly analyzed by MeCab[7], and another is an action matrix. We use morphologic analysis to make learning more effective. The analyzer absorbs endings of verbs which change variously depending on conjugation.

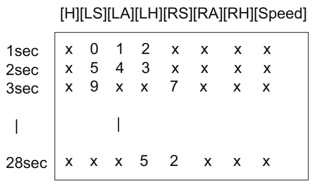


Fig. 1. Action Matrix

4.5 Generation of Rules by Inductive Learning

The system acquires meanings of connectives by generation of rules using Inductive Learning (Figure 2). The rule is consisted of natural language part and action matrix part. First, the system brings out three examples (example 1, 2 and 3 in Figure 2) from example database. Then, from natural language parts, the system realizes that example 1, 2 are EoA commands and example 3 is connective command. And the system parameterizes example 3 (@1 after that @2). Next, in action matrix parts, the system finds final positions of EoA commands (xxxx8x5, x9xxxxx) and parameterizes the motor positions of example 3. Finally, each rule has an adaptation value which is the number of identical rules.

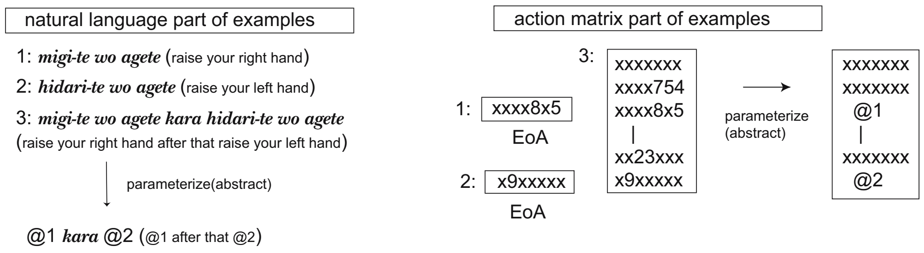

natural language part of examples
1: migi-te wo agete (raise your right hand)
2: hidari-te wo agete (raise your left hand)
3: migi-te wo agete kara hidari-te wo agete
(raise your right hand after that raise your left hand)
parameterize(abstract)
@1 kara @2 (@1 after that @2)
action matrix part of examples
1: xxxx8x5
EoA
2: x9xxxxx
EoA
3: xxxxxxx xxxx754 xxxx8x5 xx23xxx x9xxxxx
parameterize (abstract)
xxxxxxx xxxxxxx @1 xxxxxxx @2

Fig. 2. Generation of Rules by Inductive Learning

5 Experiment

We implemented the system and experimented on learning connectives. We made 4 subjects (age 20-30, all were male students of graduate school majoring math and science) input commands and teach actions to the robot. We got 157 connective commands (all 157 commands differed) and 25 connectives (Table 1). We restricted the number of EoA commands to 10 for performing the experiment.

The subjects evaluate the system responses using following evaluation marks for every system output: Correct Response, Semi-correct Response, Erroneous Response, No Response. We define Correct Response and Semi-correct Response as proper responses. In case of Erroneous Response and No Response, the subjects have to teach the correct action to the robot.

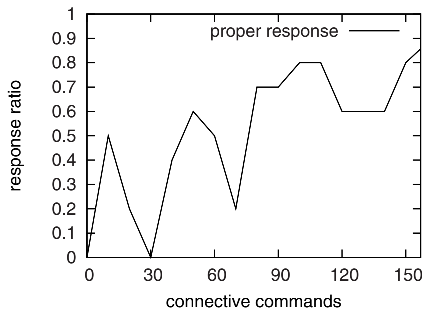


Fig. 3. Shift of the system response (connective commands)

Figure 3 shows changes in proper response ratio per 10 commands of all responses to the 157 connectives commands. This shows that the effective response line becomes stable from the third person (from the 75 connective command). Though the system did not have enough knowledge to reply with effective responses in the beginning, it managed to correct its knowledge enough to start reacting properly.

Table 1. connectives from subjects

kara (and then) **	*nagara* (and at the same time) **
mae-ni (before) **	*ato* (after) **
nai-de (without doing) **	*None* (None) **
to-doji-ni (while) **	*kara-yukkuri* (before slowly) *
sugu (do just before) *	*to-mise-kakete* (pretend to) *
ato-de (after that)	*sukoshi-matu-kara* (wait a minute)
ato-de-takusan-matsu (wait some minutes)	*tsutsu* (and)
shibaraku-shite-kara (and wait a minute)	*sonomama* (keep on)
ato-de (after that)	*soshite* (after)
soshite-takusan-matsu-kara (wait couple of minutes)	*toki* (when)
kara-sugu (soon before)	*sukoshi-matsu-kara* (wait a second)
takusan-matsu-kara (wait a couple of seconds)	

We evaluated rules applied 3 or more times in learning experiment. There were 10 such rules obtained (rules marked ** or * in Table 1). Other 4 subjects evaluated actions which the system outputs applying the rules, and 7 of the 10 connectives (rules marked **) were evaluated as "successful" or "almost successful" by all subjects. And 3 connectives (rules marked *) were evaluated as "erroneous" by at least one subject.

As a result, we can say that the system can acquire connectives which signify simple relations of order. For example, "*nagara* (and at the same time)" or "*mae-ni* (before)" or "*naide* (without doing)". However, the system could not acquire 3 connectives. Firstly, "*sugu* (do just before)" and "*kara-yukkuri* (before slowly)" had problems with acquiring proper timing. This can be acquired by increasing feedback. Secondly, "*to-mise-kakete* (pretend to do)" was incorrect due to difficult verb existence inside the connective. The system still can not understand all verbs correctly.

6 Conclusions

We proposed a method for grammatical connectives acquisition. We implemented the system, and performed learning and evaluation experiments.

From the results, we understood that learning connectives by teaching with direct physical feedback gives precision of connectives acquisition 0.7. It can be also said that our system was able to acquire a partial concept of time by embodying it. We also discovered that our system could not acquire some connectives due to the lack of understanding of verbs meaning. We try to develop mechanism of understanding verbs in the next step.

References

1. Araki, K., Tochinai, K.: Effectiveness of Natural Language Processing Method Using Inductive Learning. In: IASTED International Conference on Artificial Intelligence and Soft Computing, pp. 295–300 (2001)
2. Fukui, H., Araki, K., Tochinai, K.: The Method of Acquisition of Corresponding Relation between Language and Act Using Inductive Learning with Genetic Algorithm (in Japanease). Journal of Infomation Processing Society of Japan 45(10), 2428–2437 (2004)
3. Fernald, A., Hurtado, N.: Name in frames: infants interpret words in sentence frames faster than words in isolation. Developmental Science 9(3), 33–44 (2006)
4. Brooks, R.A.: Intelligence without representation. Artificial Intelligence 47, 139–159 (1991)
5. Kanda, T., Ishiguro, H., Ono, T., Imai, M., Nakatsu, R.: An evaluation on interaction between humans and an autonomous robot Robovie (in Japanease). Journal of Robotics Society of Japan 20(3), 315–323 (2002)
6. Kondo Kagaku Co. Ltd. `http://www.kondo-robot.com/`
7. MeCab: Yet Another Part-of-Speech and Morphological Analyzer: `http://mecab.sourceforge.jp/`

To Better Handle Concept Change and Noise: A Cellular Automata Approach to Data Stream Classification

Sattar Hashemi[1], Ying Yang[1], Majid Pourkashani[2], and Mohammadreza Kangavari[2]

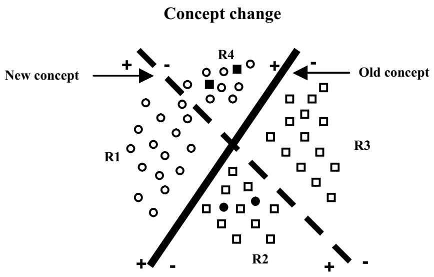
[1] Clayton School of Information Technology
Monash University, Australia
{Sattar.Hashemi,Ying.Yang}@infotech.monash.edu.au
[2] Computer Engineering Department, Iran University
Of Science and Technology, Tehran, Iran
{mpkashani,kangavari}@iust.ac.ir

Abstract. A key challenge in data stream classification is to detect changes of the concept underlying the data, and accurately and efficiently adapt classifiers to each concept change. Most existing methods for handling concept changes take a windowing approach, where only recent instances are used to update classifiers while old instances are discarded indiscriminately. However this approach can often be undesirably aggressive because many old instances may not be affected by the concept change and hence can contribute to training the classifier, for instance, reducing the classification variance error caused by insufficient training data. Accordingly this paper proposes a cellular automata (CA) approach that feeds classifiers with most *relevant* instead of most *recent* instances. The strength of CA is that it breaks a complicated process down into smaller adaptation tasks, for each a single automaton is responsible. Using neighborhood rules embedded in each automaton and emerging time of instances, this approach assigns a relevance weight to each instance. Instances with high enough weights are selected to update classifiers. Theoretical analyses and experimental results suggest that a good choice of local rules for CA can help considerably speed up updating classifiers corresponding to concept changes, increase classifiers' robustness to noise, and thus offer faster and better classifications for data streams.

1 Introduction

In data streams, the concept underlying the data may change over time, which can cause the accuracy of current classifiers to decrease. Meanwhile, real-world data are seldom perfect and often suffer from significant amount of noise, which may affect the accuracy of induced classifiers. Dealing with concept changes and differentiating them from noise has become an interesting and challenging task in the machine learning and data mining community [5,7,10]. The traditional approaches proposed for mining data streams, incremental and ensemble classifiers, are mainly based on windowing [5,8,9]. In both cases the implicit idea underlying the method is that *the more recent the data, the more relevant they are to test/train the current learner.*

This paper suggests that the arrival time of an instance is not always the best relevance criterion in changing environments because it decreases the number of instances on which the classifier is induced and hence, affects the overall performance of the classifier. To be more illustrative, let's consider a generic concept change scenario as depicted in Figure 1. The feature space is a 2-d plane with two linearly separable classes, positive and negative, whose instances are shown by circles and squares respectively. The solid line represents the "old" concept before the concept changes, where regions R1 and R4 are positive, and regions R2 and R3 are negative. Instances of the old concept are shown by "empty" shapes. The dashed line represents the "new" concept after the concept changes, where regions R1 and R2 are positive, and R3 and R4 are negative. Instances of the new concept are shown by "filled" shapes, in contrast to the older "empty" ones. Now assume that the old concept is in effect (the solid line) and the algorithm detects concept change when new instances (filled shapes) fall into regions R2 or R4 with unexpected class labels. Once the concept change is detected, the windowing approach will indiscriminately remove instances from every region as long as they are not most recent regardless of whether they are still valid for the new concept. As a result, the windowing approach will unwisely reduce the amount of training data, undermine the learner's ability to approximate the new concept and hence increase the classification error [5, 6].



Regions	R1	R2	R3	R4
Labels in old concept	+	-	-	+
Labels in new concept	+	+	-	-

Fig. 1. Regions R1 and R3 are not affected by the concept change. Their instances are still valid for the new concept and hence should be utilized to update classifiers.

To handle the above problems, the proposed method in this paper uses a cellular automata (CA) approach. Using simple neighborhood rules, The CA approach will identify those negative instances (empty squares) in R2 and those positive instances (empty circles) in R4 are no longer relevant to the learning task. Those instances will be removed while instances in regions R1 and R3 will be retained for updating classifiers. This property provides the leaner with more relevant training instance, which can reduce its classification variance. Meanwhile, noisy instances can be discarded even when they are most recent data. The strength of CA is that it breaks a complicated process down into smaller adaptation tasks, for each a single automaton is responsible. Each individual automaton is only involved with the simple rules which are applied locally to update the overall automata's states. The interested reader can refer to our last research for further information [5].

The rest of the paper is organized as follows. Section 2 offers a formal representation of the CA approach to suppress noise and handle concept change in data streams. Section 3 presents experimental results and analyses. Section 4 gives concluding remarks and suggestions for future work.

2 Methodology

A cellular automata is a discrete dynamical system and includes a regular spatial lattice (gird) of points [1,2,3]. Each point, called a cell or an automaton, is a finite state machine. The inputs to a cell are the states of all cells in its neighborhood. The states of the cells in the lattice are updated according to a local rule. That is, the state of a cell at a given time depends only on its own state and its nearby neighbors' states one time step before. Suppose the instance s_i of the data stream S can be identified by the *d*-tuple of attribute values $(\alpha_1, \alpha_2, ..., \alpha_d)$. This *d*-tuple represents a point in a *d*-dimensional problem space. Let $G = \{c_k = (D, t, w, c) \mid D = (\alpha_1, \alpha_2, ..., \alpha_d)\}$ be a d-dimensional grid (lattice) of cellular automata, which has a one-to-one relation with the instances in the problem space, and c_k represent the *k*th cell or automaton in sequential indexing of the cellular automata of concern. Each automaton is assigned three parameters that describe the state of its corresponding data point: *t* as *timetag*, *w* as *weight*, and *c* as *class*. *Timetag* is a monotonically decreasing value that shows the recency of the data. *Weight* is a parameter that accounts for the relevance of the automaton to the current concept. *Class* is the true label of the instance that is used afterwards to validate the learnt concept. We use the super-index notation to refer to parameters of a cell. Thus, c_k^t means the timetag of the cell c_k, c_k^D stands for its corresponding instance, and c_k^w for its weight. Whenever a new instance streams in, the corresponding cell is activated and its timetag and weight are initialized to a pre-defined constant *T* and *W* respectively. These parameters for other active cells are decreased by a pre-defined forgetting factor λ :

$$(c_k^t \leftarrow c_k^t - \lambda \ and \ \ c_k^w \leftarrow c_k^w - \lambda) \ \forall k \ s.t. \ c_k \ is \ \ active \ \ cell \quad (1)$$

As soon as the timetag and weight of an active cell reach zero, the cell is deactivated. It is worth mentioning that the weight of each cell can also be altered by the local rules of the cells in its neighborhood. Local rules are simple heuristics that aim at feeding classifiers with more relevant instances by taking into account the local situations of the instances in the current concept with respect to each other. By adding local rules, our CA based approach is more appropriate than the naïve recency-based windowing approach. Different rules can be defined depending on the purpose. The only restriction on these rules is the one that has been set by the CA theory: locality. We have adopted a generalized Moore neighborhood definition [4], which includes cells in a hyper-sphere that is centered at the base cell and with a radius of *n* cells.

To deal with noise, our approach simply checks for local class disagreements. For each instance, its true class is checked against the class of its neighbors. If most of the active neighbors have a different class, this instance is considered as noisy. Its weight is suppressed to an amount below the selection threshold. As a consequence, this instance is moved out of the selection basket of learning. This strategy is based on the heuristic "A positive sample will rarely emerge in-between many negative samples, and vice versa".

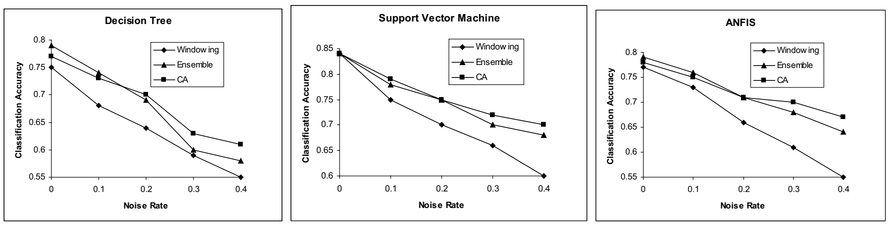
To deal with concept change, an important problem to solve is to differentiate concept change from noise because both affect classifiers' performance but in different ways. Effect of the concept change in a particular neighborhood is consistent while it is not the case for noise. Our proposed method adopts this local heuristic to detect concept change: "A newly misclassified instance, whose class label is different from most neighboring instances but is supported by coming instances from the stream, indicates a concept change". To distinguish between concept change and noise, a two-tailed binomial sign test is applied on misclassification sequence of the classifier. If the result is less than the critical level of 0.05, the drop in accuracy is the effect of concept change, otherwise it is due to noise. Whenever a concept change is detected, the misclassified new instances are considered as representatives of the new concept, while their nearby instances with a different class are considered as representatives of the outdated concept and thereafter suppressed.

3 Experiments

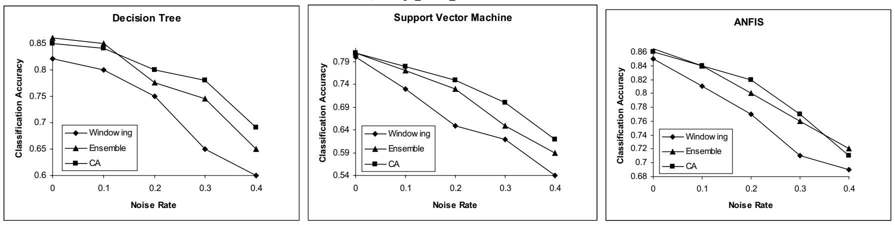
Experiments are conducted to compare the cellular automata approach with the windowing and the ensemble methods for dealing with noise and handling concept change in both synthetic and real-world data streams. Evaluations involve decision trees (DT), support vector machines (SVM) and adaptive neuro-fuzzy system (ANFIS), and also, two public benchmark datasets Hyperplane and Car dataset are used for evaluation [5]. They are relatively large, have concept changes, and are commonly used in published research of data stream classification.

To observe the role that local rules play in dealing with noise, we add different levels of noise (10%, 20%, 30% and 40%) into the data [5]. The presented results demonstrate alternative methods' behavior in face of both concept changes and different levels of noise. The CA removes instances that it has identified as noise and feed other instances to classifiers. The predictive performance of each classification algorithm (DT, SVM and ANFIS) under the windowing, the ensemble and the CA strategy respectively are compared in Figure 2. As the graphs show, the proposed CA approach yields better classification accuracy, which suggests that the local rule heuristic is more robust to noise. Although the ensemble approach performs better at first, but, it is outperformed by the CA approach in noisy environment what approves our early discussion that relying on the most recent instances is not always the best strategy in noisy environments.

To compare the performance of different approaches on handling concept change in data streams, we use the Hyperplane dataset [8] because we can manipulate different amount of concept changes. In contrast, the amount of concept changes in real-word data such as the Car dataset can not be changed. In particular, the amount of concept changes in Hyperplane relates to the number of dimensions *k* whose weights are changing over time. Using four different values of *k*, we produce several



a) Hyperplane dataset



b) Car dataset

Fig. 2. Classification accuracy of different classification algorithms under the windowing, the ensemble and the CA approaches as a function of noise level

Table 1. Experiment results in learning drifting concepts

Learner	*k*	Windowing Approach			Ensemble Approach			CA Approach		
		acc	*ct*	$N_\hbar$	*acc*	*ct*	$N_\hbar$	*acc*	*ct*	$N_\hbar$
Decision Tree	1	75.0%	43	70	79%	29	85	77.0%	31	58
	2	71.2%	47	79	73%	34	92	73.3%	37	67
	3	60.8%	72	95	65%	61	111	64.8%	57	74
	4	56.0%	75	104	60%	67	123	61.3%	60	86
SVM	1	84.0%	47	66	84%	34	80	84.0%	35	54
	2	73.0%	70	80	78%	47	89	77.7%	45	62
	3	69.7%	77	85	74%	50	100	76.0%	48	66
	4	58.0%	100	96	67%	67	119	68.3%	63	73
ANFIS	1	77.0%	33	69	79%	15	72	78.1%	20	56
	2	71.3%	40	77	74.5%	23	80	75.0%	23	73
	3	70.9%	41	86	72.9%	26	98	73.4%	25	78
	4	63.8%	57	102	65%	44	113	66.0%	40	83

Hyperplane datasets to evaluate the performance of our approach from different points of view. Three measures of the performance are calculated for each experiment: 1) convergence time, *ct,* which is the average time the system remains unstable during updating to a concept change, measured by the number of instances; 2) classification accuracy, *acc*, which is the percentage of the classifier's correct predictions, and 3) re-learning requirement, $N_\hbar$, as the total number of times the classifier is re-learned from scratch. Results are shown in Table 1. It is observed that almost in every case the CA approach outperforms the windowing and ensemble approaches: it achieves higher classification accuracy; spends less time on updating classifiers upon concept changes; and require less frequently building classifiers from scratch. Lower *ct* and $N_{\hbar}$ are desirable because data streams demand fast online classification. In other words, whenever the other approaches present almost equal classification accuracy, it is at the cost of using more time resources.

4 Conclusion and Future Work

A cellular automata (CA) approach is proposed for data stream classifications. Its advantage is to use most relevant instances instead of most recent instances to update classifiers in face of concept changes. By using neighboring instances and simple local rules, the CA approach can be more robust to noise, achieve higher classification accuracy, adapt faster to changed concepts, and less frequently require building classifiers from scratch.

Using cellular automata for data stream classification is a new topic and there are various interesting issues to further investigate. For example, as cellular automata are distributed along a grid with just interactions between neighboring cells, they can be consider as a means for parallel stream mining. For another example, cellular automata have a discrete nature. Hence, an effective online method for discretizing continuous attributes in data streams may be considered for future work.

References

1. Farmer, J.D., Toffoli, T., Wolfram, S. (eds.) Cellular Automata.Proceedings of an Interdisciplinary Workshop, Los Alamos, New Mexico (March 7-11, 1983)
2. Wolfram, S. (ed.): Theory and Applications of Cellular Automata. World Scientific, Singapore (1986) Collection of papers on CA's
3. Folioni, G., Pizzuti, C., Spezzano, G.: A Cellular Genetic Programming Approach to Classification. In: Proceedings of the Genetic and Evolutionary Computation Conference, vol. 2, pp. 1015–1020 (1999)
4. Maji, P., Shaw, C., Ganguly, N., Sikdar, B.K., Chaudhuri, P.P.: Theory and Application of Cellular Automata for data mining. Fundamenta Informaticae (58), 321–354 (2003)
5. Hashemi, S., Yang, Y., Pourkashani, M., Kangavari, M.: To Better Handle Concept Change and Noise: A Cellular Automata Approach to Data Stream Classification. Technical Report, Monash University (2007)
6. Widmer, G., Kubat, M.: Learning is presence of concept drift and hidden contexts. Machine Learning 23(1), 69–101 (1996)
7. Klinkenberg, R.: Learning drifting concepts: example selection vs. example weighting, Intelligent Data Analysis. Special Issue on Incremental Learning Systems Capable of Dealing with Concept Drift 8(3) (2004)
8. Hulten, G., Spencer, L., Domingos, P.: Mining time-changing data streams. In: SIGKDD. Proceedings of the 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 97–106 (2001)
9. Wang, H., Fan, W., Yu, P.S., Han, J.: Mining concept drifting data streams using ensemble classifiers. In: SIGKDD. Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 226–235 (2003)
10. Zhu, X., Wu, X., Yang, Y.: Effective Classification of Noisy Data Streams with Attribute-Oriented Dynamic Classifier Selection. Knowledge and Information Systems An International Journal (KAIS) 9(3), 339–363 (2006)

Categorizing Software Engineering Knowledge Using a Combination of SWEBOK and Text Categorization

Jianying He, Haihua Yan, Maozhong Jin, and Chao Liu

School of Computer Science & Engineering, Beihang University,
100083 Beijing, P.R. China
durrcyhe@sina.com, {yhh, jmz, liuchao}@buaa.edu.cn

Abstract. In this paper, we utilize a combination of SWEBOK and text categorization to categorize software engineering knowledge. SWEBOK serves as a backbone taxonomy while text categorization provides a collection of algorithms including knowledge representation, feature enrichment and machine learning. Firstly, fundamental knowledge types in software engineering are carefully analyzed as well as their characteristics. Then, incorporated with SWEBOK, we propose a knowledge categorization methodology as well as its implementing algorithms. Finally, we conduct experiments to evaluate the proposed method. The experimental results demonstrate that our methodology can serve as an effective solution for the categorization of software engineering knowledge.

Keywords: Knowledge categorization, Software engineering, SWEBOK, Text Categorization.

1 Introduction

Software development is one of the most knowledge intensive processes that occur in modern organizations. It involves many intellectual assets, including models used, developers' personal skills, technical artifacts, best software development practices, and even the whole process experiences. It is necessary to accumulate and disseminate these valuable assets through the organization. By doing this, developers can facilitate organizational memory to concentrate their creativity on solving technical problems, rather than re-inventing the wheel [1]. Knowledge management is viewed as one of the most effective approach to leverage organizational knowledge during the passed decade [2].

Central to knowledge management is the idea of categorizing, i.e. arranging knowledge in a generalization or inheritance hierarchy [3]. After a broad literature survey, we identify ongoing efforts for knowledge categorization mainly in two directions: text mining approach and ontological approach. The former one either employs text clustering method or adapts text categorization method [4]. While, the later one predefines ontological structures and assigns metadata to the artifacts that conform to these conceptual structures.

Although these state-of-the-art approaches address the problem to some extent and serve as a basis for further investigation, there still remains much work to do. Especially in software engineering area, the following problems should be paid extra attention to:

- Precision and recall

In contrast with Internet documents, scale of software organizations' knowledge assets is far smaller but its categorization performance requires a reasonable higher precision and recall. However, current text mining appears as though a performance plateau has been reached, and improvements are becoming evolutionary [5].

- Automation

Ontological approach can solve the problem of precision and recall. However, currently ontological curation is laborious, being done largely manually with few automatic aids which is not tolerable for the user.

- Backbone taxonomy

Although software organizations' knowledge assets vary tremendously, there must have a backbone taxonomy which can serve as a shared fundamental reference. However, no obvious evidence shows that this issue has been fully exploited.

In order to solve the above problems, a fundamentally new approach is apparently necessary. We argue that knowledge categorization is a little different from text categorization, mainly in threefold. Firstly, knowledge categorization should encompass both explicit and tacit knowledge while text categorization only processes explicit documents. Secondly, knowledge categorization is domain-specific and should treat different knowledge assets according to their characteristics while text categorization is not. Finally, we recognize that knowledge categorization should have high accuracy which can be facilitated by using background knowledge.

In this paper, we tentatively propose a novel solution that capitalizes on a combination of SWEBOK[1] (Software Engineering Body of Knowledge) and fruited text categorization techniques. SWEBOK serves as a backbone taxonomy while text categorization provides a collection of implementation algorithms. Prior to knowledge categorization, we build a knowledge representor dealing with intellectual assets differently according to their analyzed characteristics. In order to improve categorization performance, we enrich text representation with new, more informative features that generated under the direction of ontologies.

2 Knowledge Categorization Methodology

2.1 Three Types of Knowledge

Any knowledge generated and used to reduce rework and improve quality during software development process should be regarded as software engineering knowledge [2]. However, software engineering knowledge can be generally divided into the three types: (1) process experiences; (2) knowledge artifacts; (3) personal skills.

[1] www.swebok.org/

At the core of a software organization's memory, it is process experiences or lessons learned, enabling reuse and sharing of organizational knowledge [2]. However, to compose and organize the process experiences tends to be a difficult work because every software development project is unique in some sense. Characteristics of process experiences mainly include: (1) high dependency on specific project, (2) hard to follow a given process model and reusable between similar projects.

Developer's skills involved are the most important knowledge needed to capture and represent because all activities are carried out by human in the final analysis [6]. Knowledge in developers' mind has to be systematically collected, stored in a corporate memory, and shared across the organization. However, to do so is a rather challenging task. Characteristics of personal skill include: (1) tacit, (2) high value and (3) unstable.

Any explicit knowledge existing in software organization can be akin to knowledge artifacts. An organization's knowledge repository should also contain the knowledge artifacts obtained throughout the projects [7]. The knowledge artifacts could be a list of requirements, a case-tool graphic or some software code. Characteristics of knowledge artifacts include: (1) explicit, (2) large scale and (3) relatively easy to processing.

2.2 Representation of Software Engineering Knowledge

Figure 1 illustrates our overall methodology and corresponding system architecture for knowledge representation.

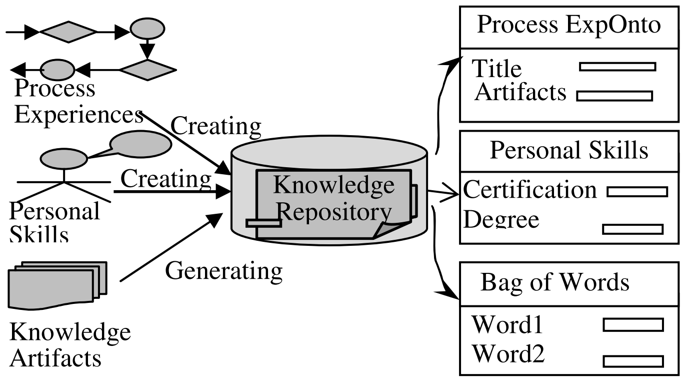


Fig. 1. A sketch of knowledge representation

In order to perform knowledge assets representation tasks once for all, we integrate three open-source toolkits to meet our final requirement: (1) Protégé[2] for ontology management, (2) Rainbow[3] for text representation and (3) OpenCyc4 for next-step

[2] http://protege.stanford.edu/

[3] http://www.cs.umass.edu/~mccallum/bow/rainbow/

[4] http://www.opencyc.org/

feature enrichment. The resulting toolkit is called *knowledge representor* according its true purpose in our scheme.

For process experiences and personal skills, the knowledge representor uses manually built ontologies to represent them and accumulates their ontologies' instance into an organizational knowledge repository.

For knowledge artifacts, the knowledge representor employs its integrated Rainbow system to deal with knowledge intensive documents, completely in a way of text representation which can convert a text document into a vector of weighted words. In order to promote effectiveness and accuracy, we also perform "stop words removal", which aims to take high frequency words out of the text representation dictionary, and "words stemming" which intends to reduce the dimensionality of feature space. Herein, knowledge artifacts are turned into a vector of weighted text features.

We have argued that knowledge categorization should leverage background knowledge. In our approach, we exploit background knowledge about concepts that is explicitly given according to OpenCyc ontology base. For this purpose, we enrich each document vector by new features crawled in OpenCyc. The crawling algorithm is shown in table 1.

Table 1. The new features crawling algorithm

```
Input: document vector consisting of n terms V = {$t_1$ $t_2$ $t_n$}
    OpenCyc consisting a concept set C = $\Phi$
    Temperal_Concept_Set = $\Delta$
    For(I = 1, I <= n, i++) do
        if $t_i$ in $\Phi$
            then find the corresponding concept C
                get all the relevant concept of C
                add them to $\Delta$
      end for
Return $\Delta$ as the enriched features
```

2.3 Categorization of Software Engineering Knowledge

Figure 2 depicts an overall framework of our proposed knowledge categorization methodology. Like knowledge representation, different types of knowledge are treated differently. SWEBOK plays a crucial role during the categorizing process.

Taxonomies have been proved to be a popular way in which to build a domain model to help users to search and navigate [8]. In order to truly leverage its knowledge repository, a software organization must establish a taxonomy to organize the represented knowledge available to the organization members. We leverage SWEBOK guide to build a backbone taxonomy, as well as with flexibility. The backbone taxonomy is four-level hierarchical structure because SWEBOK is organized in the same way.

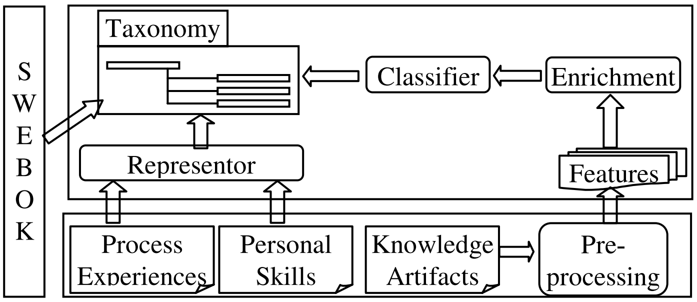


Fig. 2. A sketch of knowledge representation

For personal skills and process experiences, their OWL-format representation can be easily cast to nodes in the SWEBOK taxonomy tree as instances to the respective ontological categories. The reason is that their OWL representation carries structure information which can serves as a direct mapping to the intended categories.

For knowledge artifacts, many different supervised categorization algorithms have been designed and virtually all of them have been used for text categorization tasks, including Naive Bayes, Support Vector Machines and k-Nearest-Neighbours etc [9]. However, prior studies found SVM to have the best performance for text categorization [10]. In our current implementation, we employ SVM text classifier embedded in Rainbow to categorize knowledge artifacts. It compares its input document vectors to all SWEBOK-backboned taxonomy nodes, and returns the desired number of best-matching ones.

3 Experimental Results

3.1 Experimental Design

In the experiments, we use three corpora: WebKB and two manually built dataset. The WebKB[5] dataset (World Wide Knowledge Base) contains WWW-pages collected from computer science departments of various universities by the CMU text learning group. We select four categories (course, faculty, project and student) which are tightly related to computer science as our experimental dataset. These four categories consist of 4,199 documents and we call it WebKB4199.

Also, we manually build another dataset based on the 169 referenced papers list in SWEBOK. Unfortunately, we cannot get all the reference but we make amends to it with other papers crawled from the Internet. Finally, we get a dataset containing 200 papers and call it SWEBOK-Ref. As for tacit knowledge assets, we generate ten instances of process experiences ontology and 50 instances of personal skills ontology.

To evaluate the categorization performance, we use the precision, recall and BEP [9] as our measurement.

[5] http://www.cs.cmu.edu/~webkb/

We plan to test our proposed methodology from two aspects, tacit knowledge aspect and explicit knowledge aspect. Accordingly, experiments are conduct in two stages, tacit knowledge categorizing stage and explicit knowledge categorizing stage. In the first stage, we aim to evaluate performance of our proposed categorization methodology on tacit knowledge using the manually built instances of process experiences and personal skills. In the second stage, classification is performed to evaluate explicit knowledge categorization. Firstly, we split WebKB dataset into three parts and used two parts for training and the remaining third for testing. Then, we conducted the training-test procedure three times and averaged the performance as for the final results using SVM classifier embedded in Rainbow. Finally, we follow the same process on manually acquired 200 papers just as what is conducted on WebKB dataset.

3.2 Results and Analysis

It's obvious that the accuracy of knowledge categorization on process experiences and personal skills must be 100% because their OWL representation carries structure information which can serves as a direct casting to the intended categories.

Table 2 demonstrates the results of categorization of knowledge artifacts using SVM classifier. For both WebKB4199 and SWEBOK-Ref datasets, we consistently observed a little scaling down both in macro-BEP and micro-BEP when the size of datasets grows. This goes in line with our expectations that the scale of dataset will affect categorization performance. However, the average accuracy still reaches a level on about 90% which demonstrates effectiveness of our proposed methodology. Another important phenomena also can be readily seen, micro-BEP on SWEBOK-Ref is improved for notable about 7% than on WebKB4199. This variation in performance is due to differences in data content because, in contrast with WebKB4199, SWEBOK-Ref is more tightly related with software engineering.

Table 2. Evaluation result on WebKB and paper set

	Precision	Recall	micro BEP	macro BEP
WebKB4199				
504 docs	0.92	0.84	0.91	0.84
4199 docs	0.89	0.72	0.89	0.68
SWEBOK-Ref				
20 papers	0.99	0.98	0.98	0.98
200 papers	0.96	0.94	0.95	0.95

4 Conclusion and Future Work

In this paper, we have introduced a fully automatic novel solution based on a combination of SWEBOK and text categorization. In order to improve precision and recall, we rendered machine learning algorithms with a common-sense ontology base. Prior to knowledge categorization, we built a knowledge representor generator translating intellectual assets into machine-readable format once for all. Furthermore, we also constructed a SWEBOK-backboned taxonomy, as well as with high flexibility, which facilitates knowledge sharing and dissemination.

Experimental results definitively confirmed that our proposed methodology can automatically categorize both tacit and explicit software engineering knowledge effectively and efficiently. Also, the validated data clarified the usefulness of standard machine learning algorithms for the knowledge categorization task.

In contrast with text categorization, knowledge categorization is a relatively new domain. We believe that our research only scratches the surface of what can be achieved in the future. Our next step work aims to investigate new algorithms to reduce the computational complexity. Moreover, we conjecture that our proposed knowledge categorization methodology will be very useful for other knowledge management tasks, and we intend to investigate this in the future.

References

1. Birk, A., Surmann, D., Althoff, K.D.: Applications of Knowledge Acquisition in Experimental Software Engineering. In: Fensel, D., Studer, R. (eds.) EKAW 1999. LNCS (LNAI), vol. 1621, pp. 67–84. Springer, Heidelberg (1999)
2. Ligia, M.S.B., Ricardo, A.F.: Managing Software Process Knowledge. In: CSITeA 2002. 2nd International Conference on Computer Science, Software Engineering, Information Technology, e-Business, and Applications, pp. 273–278. ACIS Press, Foz do Iguacu (2002)
3. Ioana, R., Mikael, L., Sachin, S.S.: Knowledge Management in Software Engineering. IEEE Software 19, 26–38 (2002)
4. Stephan, B., Philipp, C., Andreas, H., Steffen, S.: An Ontology-based Framework for Text Mining. LDV-Forum 20, 87–112 (2005)
5. Evgeniy, G., Shaul, M.: Feature Generation for Text Categorization Using World Knowledge. In: 19th International Joint Conference on Artificial Intelligence, pp. 1048–1053. Professional Book Center, Edinburgh (2005)
6. Philip, N.: Ontology-based Retrieval of Software Engineering Experiences. University of Calgary Theses, Calgary (2003)
7. Ceravolo, P., Damiani, E., Marchesi, M., Pinna, S., Zavatarelli, F.: An Ontology-based Process Modeling for XP. In: 10th Asia-Pacific Software Engineering Conference, pp. 236–242. IEEE Press, Chiangmai (2003)
8. Brent, G.: Knowledge Management Systems: Surveying the Landscape. International Journal of Management Review 3, 61–77 (2001)
9. Sebastiani, F.: Machine Learning in Automated Text Categorization. ACM Computing Surveys 34, 1–47 (2002)
10. Yiming, Y., Xin, L.: A Re-examination of Text Categorization Methods. In: 22nd Annual International ACM SIGIR Conference on Research and Development in the Information Retrieval, pp. 42–49. ACM Press, Hong Kong (1999)

An Improved Probability Density Function for Representing Landmark Positions in Bearing-Only SLAM Systems

Henry Huang, Frederic Maire, and Narongdech Keeratipranon

Faculty of Information Technology, Queensland University of Technology
Box 2434, Brisbane Q 4001, Australia
{h5.huang@student, f.maire@, n.keeratipranon@student}qut.edu.au

Abstract. To navigate successfully, a mobile robot must be able to estimate the spatial relationships of the objects of interest in its environment accurately. The main advantage of a bearing-only *Simultaneous Localization and Mapping* (SLAM) system is that it requires only a cheap vision sensor to enable a mobile robot to gain knowledge of its environment and navigate. In this paper, we focus on the representation of the spatial uncertainty of landmarks caused by sensor noise. We follow a principled approach for computing the *Probability Density Functions* (PDFs) of landmark positions when an initial observation is made. We characterize the PDF $p(r, \alpha)$ of a landmark position expressed in polar coordinates when r and α are independent, and the marginal probability $p(r)$ of the PDF is constrained to be uniform.

1 Introduction

There exist well developed solutions to the SLAM problem for the case where the mobile robot is equipped with a sensor that provides both range and bearing measurements to landmarks. However, the sensors these solutions rely on are considered too expensive for domestic robots. Solving the SLAM problem with cheaper sensors is highly desirable. Thanks to recent advances in computer vision and cheaper cameras, vision sensors have become popular for SLAM [1,2,3].

In general, vision sensors are noisy. The uncertainty of an object location caused by sensor noise can be quantified with a *Probability Density Function* (PDF). When a robot is initially placed in an unknown environment without any prior information, the PDF of an object position should be uniform over the whole environment. Once the robot starts to sense the environment, information gathered can be used to update the PDF.

Robustness to sensor noise can be achieved with probabilistic methods such as *Extended Kalman Filtering* (EKF) [1,2,6,10] or *Particle Filtering* (PF) [7,8]. Iterative EKF methods for solving the bearing-only SLAM problems can be found in [1,2]. Some immediate initialization methods to bearing-only SLAM called *Undelayed* methods were introduced in [6,10]. The common approach of the undelayed methods is to initialize each landmark by a mixture of Gaussians

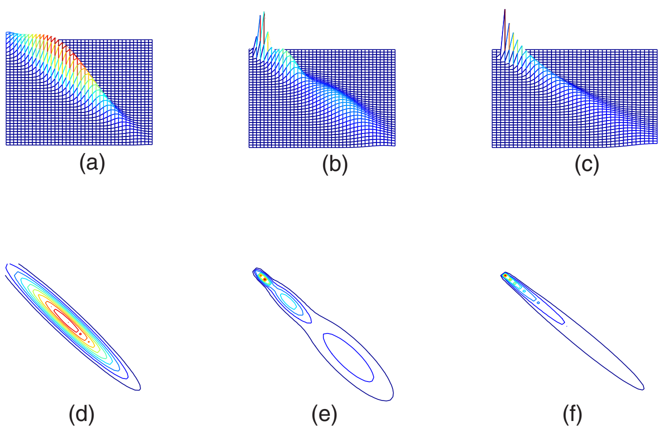


Fig. 1. PDFs of a landmark position represented by different methods: (a) The PDF represented by a single Gaussian. (b) The PDF represented by a mixture of three Gaussians. (c) The PDF we derive. Figures (d), (e) and (f) are the contour plots of figures (a), (b) and (c) respectively.

along the direction of the landmark. PF has become popular for SLAM because PF can approximate multi-modal probability density function. Several solutions to bearing-only SLAM have been proposed by combining PF with vision sensor [3,7,9].

This paper follows a principled approach to determine the spatial uncertainty of an object position when an initial observation is made. Without distance measurement, the probability of an object position should be uniform over the sensor range. We characterize the PDF $p(r, \alpha)$ expressed in polar coordinates that satisfies $p(r|\alpha) = p(r)$ and for which the marginal probability $p(r)$ is uniform. The difference between a Gaussian approximation and our PDF is illustrated in Fig. 1. Existing methods [2,6,10] assume that the probability density distribution of the object position is a Gaussian or a mixture of Gaussians, and do not enforce the range uniformity requirement.

2 A Characterization of the Landmark Probability $p(r, \alpha)$ When r and α Are Independent, and $p(r)$ Is Uniform

Let $p(r, \alpha)$ be the PDF of the landmark position in polar coordinates when only one observation has been made. We characterize $p(r, \alpha)$ when r and α are independent. Let β denote the measured bearing of a landmark. Assume that the vision error for the bearing is smaller than ϵ, the landmark position is contained in the *vision cone* which is formed by two rays $\beta - \epsilon$ and $\beta + \epsilon$ rooted at the observation point (see Fig. 2).

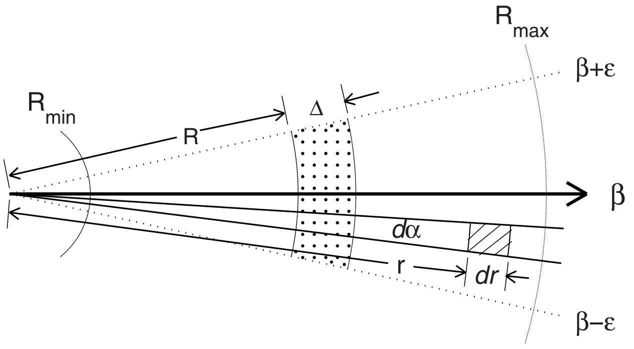


Fig. 2. The vision cone is rooted at the observation point. The surface of the hashed area is approximately $r\,dr\,d\alpha$ for small dr and $d\alpha$.

In Fig. 2, the surface of the hashed area for small dr and $d\alpha$ can be approximated as: $\left[\pi(r+dr)^2 - \pi r^2\right]\frac{d\alpha}{2\pi} = \frac{1}{2}\left[2\,r\,dr + (dr)^2\right]d\alpha \cong r\,dr\,d\alpha$. Because the probability of the landmark being in the vision cone is 1, we have

$$\int_{\beta-\epsilon}^{\beta+\epsilon}\int_{R_{\min}}^{R_{\max}} p(r,\alpha)\,r\,dr\,d\alpha = 1 \tag{1}$$

In (1), $R_{\min}$ and $R_{\max}$ are the bounds of the vision range interval. We define $F(R)$ as the probability of the landmark being in the area $\{(r,\alpha) \mid r \in [R_{\min},\ R],\ \alpha \in [\beta-\epsilon,\ \beta+\epsilon]\}$. This probability $F(R)$ can be written as:

$$F(R) = \int_{\beta-\epsilon}^{\beta+\epsilon}\int_{R_{\min}}^{R} p(r,\alpha)\,r\,dr\,d\alpha \tag{2}$$

We define $\Psi(R,\Delta)$ as the probability of the landmark being in the dotted area in Fig. 2. Since $\Psi(R,\Delta) = F(R+\Delta) - F(R)$, we have

$$\Psi(R,\Delta) = \int_{\beta-\epsilon}^{\beta+\epsilon}\int_{R}^{R+\Delta} p(r,\alpha)\,r\,dr\,d\alpha \tag{3}$$

As the range r and the angle α are independent, $\Psi(R,\Delta)$ is constant with respect to R. That is $\frac{\partial\Psi(R,\Delta)}{\partial R} = 0$, therefore

$$\frac{\partial F(R+\Delta)}{\partial R} = \frac{\partial F(R)}{\partial R} \tag{4}$$

Because of the independence of bearing and range assumption, $p(r,\alpha)$ can be factored as

$$p(r,\alpha) = p(r|\alpha)\,p(\alpha) = p(r)\,p(\alpha) = f(r)\,g(\alpha) \tag{5}$$

Thus we have $\int_{\beta-\epsilon}^{\beta+\epsilon} g(\alpha)\,d\alpha = 1$. After factoring, (2) becomes $F(R) = \int_{R_{\min}}^{R} f(r)\,r\,dr$. Thanks to a well-known property of the derivation of an integral with respect to its bounds, we have

$$\frac{\partial F(R)}{\partial R} = f(R)\,R \tag{6}$$

From (4) and (6), we deduce that $f(R+\Delta)(R+\Delta) = f(R)\,R$
Therefore, $R\left[\frac{f(R+\Delta)-f(R)}{\Delta}\right] + f(R+\Delta) = 0$
By making Δ goes to zero, we obtain $Rf'(R) + f(R) = 0$
The equality $\frac{f'(R)}{f(R)} = \frac{-1}{R}$ can be re-written as $[\log(f(R))]' = -[\log(R)]'$
After integrating both sides, we obtain

$$\log(f(R)) = -\log(R) + c = \log\frac{1}{R} + c$$

Where c is some constant. Setting $\xi = e^c$, we obtain $f(R) = \frac{\xi}{R}$. From (1) and (5), $\xi = \frac{1}{R_{\max}-R_{\min}}$, and thus

$$f(r) = \frac{1}{(R_{\max} - R_{\min})\,r}$$

Hence,

$$p(r,\alpha) = \frac{1}{(R_{\max} - R_{\min})\,r} \times g(\alpha) \tag{7}$$

We call the density function $p(r,\alpha)$ in (7) the *impartial PDF*, because it treats all r equally by making the marginal $p(r)$ uniform. If $g(\alpha)$ is a Gaussian function, the PDF $p(r,\alpha)$ has the shape plotted in Fig. 1c.

3 Applying the Impartial PDF to Bearing-Only SLAM

One fundamental problem facing a robot-based mapping system is how to translate the relative positions of objects with respect to the moving frame of the robot (as well as the position uncertainties) into relative positions with respect to a static frame in the environment. In [4], we introduce a geometric method to solve this problem when the uncertainty regions are polygons. In a technical report [5] (an extended version of this paper), we extend the geometric method to a probabilistic framework. More precisely, we apply the impartial PDF and represent the location uncertainty with a set of particles and adapt the Particle Filtering algorithm to transfer the uncertainty from the moving frame to a static frame. In [5], we describe how the impartial PDF of an object position can be refined with a second observation, and how to further refine the PDF with subsequent observations by Particle Filtering.

4 Conclusion

In this paper, we have followed a principled approach for computing the PDFs of landmark positions when an initial observation is made. We have characterized the PDF $p(r,\alpha)$ of a landmark position expressed in polar coordinates when r and

α are independent, and the marginal probability $p(r)$ of the PDF is constrained to be uniform. Existing methods that approximate an object position with a mixture of Gaussians do not satisfy these natural requirements. In particular, the impartial PDF we derived has the advantage of having far fewer parameters than a mixture of Gaussians.

References

1. Bailey, T.: Constrained initialization for bearing-only SLAM. In: IEEE International Conference on Robotics and Automation (2003)
2. Costa, A., Kantor, G., Choset, H.: Bearing-only Landmark Initialization with Unknown Data Association. In: Proceedings of the 2004 IEEE International Conference on Robotics and Automation, vol. 2, pp. 1764–1770 (2004)
3. Davison, A., Cid, Y., Kita, N.: Real-time 3D SLAM with wide-angle vision. In: IAV2004. 5th IFAC/EURON Symposium on Intelligent Autonomous Vehicles, Lisboa Portugal (July 2004)
4. Huang, H., Maire, F., Keeratipranon, N.: Uncertainty Analysis of a Landmark Initialization Method for Simultaneous Localization and Mapping. In: Proceedings of Australian Conference on Robotics and Automation, Sydney (2005)
5. Huang, H., Maire, F., Keeratipranon, N.: An improved probability density function for representing landmark positions in bearing-only slam systems. Technical report (2007), available at http://eprints.qut.edu.au/archive/00009397/
6. Kwok, N.M., Dissanayake, G.: An Efficient Multiple Hypothesis Filter for Bearing-Only SLAM. In: Proceedings of 2004 IEEERSJ International Conference on Intelligent Robots and Systems (2004)
7. Menegatti, E.P.E., Zoccarato, M., Ishiguro, H.: Image-based Monte-Carlo localisation with omnidirectional images. In: Proceedings of Robotics and Autonomous Systems, vol. 48, pp. 17–30 (2004)
8. Montemerlo, M., Thrun, S., Koller, D., Wegbreit, B.: FastSLAM 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges. In: Proceedings of IJCAI 2003, Morgan Kaufmann, San Francisco (2003)
9. Sim, R., Elinas, P., Griffin, M., Little, J.J.: Vision-based slam using the rao-blackwellised particle filter. In: Proceedings of IJCAI Workshop on Reasoning with Uncertainty in Robotics, Edinburgh, Scotland (2005)
10. Sola, J., Monin, A., Devy, M., Lemaire, T.: Undelayed Initialization in Bearing Only SLAM. In: IROS. Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 2751–2756 (August 2005)

Weight Redistribution for Unweighted MAX-SAT

Abdelraouf Ishtaiwi[1,2], John Thornton[1,2], and Abdul Sattar[1,2]

[1] IIIS, Griffith University, QLD, Australia
[2] DisPRR, National ICT Australia Ltd, QLD, Australia
{a.ishtaiwi,j.thornton,a.sattar}@griffith.edu.au

Abstract. Many real-world problems are over-constrained and require search techniques adapted to optimising cost functions rather than searching for consistency. This makes the MAX-SAT problem an important area of research for the satisfiability (SAT) community. In this study we perform an empirical analysis of several of the best performing SAT local search techniques in the domain of unweighted MAX-SAT. In particular, we test two of the most recently developed SAT clause weight redistribution algorithms, DDFW and DDFW$^+$, against three more well-known techniques (RSAPS, AdaptNovelty$^+$ and PAWS). Based on an empirical study across a range of previously studied problems we conclude that DDFW is the most promising algorithm in terms of robust average performance.

1 Introduction

Since the development of GSAT [1] in 1992, there has been considerable interest and progress in developing stochastic local search (SLS) techniques for solving propositional satisfiability (SAT) problems. While the SAT problem has many important applications, it is the case that many, if not most, real-world problems are *over-constrained.* If we express such problems using the SAT formalism, then the search task is no longer to find whether a satisfying assignment exists, but to find an assignment that maximises the number of true clauses (or equivalently minimises the number of false clauses). This problem is known as the unweighted MAX-SAT problem and has many applications in such areas as scheduling and the processing of Bayesian networks [2].

2 Local Search for Unweighted MAX-SAT

Current state-of-the-art SLS techniques can be divided into two main categories: WalkSAT-based heuristics and dynamic local search (DLS) clause weighting heuristics. Of the WalkSAT techniques, AdaptNovelty$^+$ [3] is recognised as one of the most robust and effective SAT solvers for situations where manual parameter tuning is impractical. By default, the Novelty heuristic [4] will take the best cost flip within a randomly selected false clause, unless this flips the clauses's most recently flipped variable. In that case, Novelty probabilistically chooses between taking the clauses's best and second best cost flip according to a *noise* parameter. The adapt aspect of AdaptNovelty$^+$ automatically controls this noise parameter according to the level of search stagnation, i.e. the more search steps that have elapsed since the last improving move, the greater the noise and hence the probability of selecting the second best flip in a clause. Finally, the $^+$ aspect of AdaptNovelty$^+$ introduces a small probability of taking a randomly selected flip, regardless of cost, to avoid certain deterministic behaviours identified in [5].

Dynamic local search (DLS) SAT algorithms take the approach of adding and manipulating clause weights to provide guidance and avoid stagnation during a search. This means the cost of a flip is calculated by summing the *weights* of the false clauses rather than simply counting their number. Typically, clause weighting involves a two stage process: firstly weight is added to all false clauses when the search becomes trapped in a local minimum (i.e. when it runs out of cost reducing moves). This increases the cost of violating just those clauses that are false at that point in the search. Secondly, clause weights are periodically reduced to avoid the immediate effects of a clause weight increase being swamped by the accumulated weight of past increases. The two older clause weight algorithms used in the current study (PAWS [6] and RSAPS [7]) differ mainly in using an additive weight scheme (PAWS) which increases and decreases weight in single integer units, as opposed to using a multiplicative scheme (RSAPS) that increases and decreases weight according to two real valued multipliers. RSAPS also borrows the adaptive mechanism from AdaptNovelty to adjust its weight reducing multiplier.

2.1 Divide and Distribute Fixed Weight

More recently, a new approach to clause weighting has been proposed, termed divide and distribute fixed weight (DDFW) [8]. DDFW introduces two ideas into the area of clause weighting algorithms for SAT. Firstly, it evenly distributes a fixed quantity of weight across all clauses at the start of the search, and escapes local minima by transferring weight from satisfied to unsatisfied clauses in a single step (rather than using the two phase approach of SAPS and PAWS). Secondly, DDFW exploits neighbourhood relationships between clauses when deciding which pairs of clauses will exchange weight (see [8] for details). For the current MAX-SAT study we additionally changed DDFW so that it keeps track of the current best cost and the current best solution (see lines 3-5 and 9-12 in Algorithm 1).[1]

DDFW comes in two versions: DDFW and DDFW$^+$. DDFW$^+$ uses an adaptive mechanism to alter the total amount of weight distributed according to the degree of stagnation in the search. The original DDFW initialises the weight of each clause to W_{init} (fixed at 8 in the current study), whereas DDFW$^+$ sets this value at 2 and adapts it during the search as follows: if the search executes a consecutive series of i flips without reducing the total number of false clauses,

[1] The same changes were made to all the SAT algorithms appearing in the study.

Algorithm 1. DDFW-MAX-SAT($\mathcal{F}$, $MaxTime$, W_{init}, C_{target}, C_{best}, A_{best})

```
1:  $A \leftarrow$ a randomly generated truth assignment of $\mathcal{F}$
2:  set the weight $w_i$ for each clause $c_i \in \mathcal{F}$ to $W_{init}$
3:  set $C_{best} \leftarrow$ number of false clauses in $A$
4:  set $A_{best} \leftarrow A$
5:  while time $< MaxTime$ and $C_{best} > C_{target}$ do
6:      select list $\mathcal{L}$ of literals causing the greatest reduction in weighted cost $\Delta w$ when flipped
7:      if ($\Delta w < 0$) or ($\Delta w = 0$ and probability $\leq 15\%$) then
8:          update $A$ by flipping a literal randomly selected from $\mathcal{L}$
9:          if (number of false clauses in $A < C_{best}$) then
10:             set $C_{best} \leftarrow$ number of false clauses in $A$
11:             set $A_{best} \leftarrow A$
12:         end if
13:     else
14:         for each false clause $c_f$ do
15:             select a satisfied same sign neighbouring clause $c_k$ with maximum weight $w_k$
16:             if $w_k < W_{init}$ then
17:                 randomly select a clause $c_k$ with weight $w_k \geq W_{init}$
18:             end if
19:             if $w_k > W_{init}$ then
20:                 transfer a weight of 2 from $c_k$ to $c_f$
21:             else
22:                 transfer a weight of 1 from $c_k$ to $c_f$
23:             end if
24:         end for
25:     end if
26: end while
```

where i is equal to the number of literals in the problem, then the amount of weight on each clause is increased by 1 in the first instance. However, if after increasing weights, the search enters another consecutive series of i flips without improvement, then it will reset the weight on each satisfied clause back to 2 and on each false clause back to 3. The search then continues to follow each increase with a reset and each reset with an increase. In this way a long period of stagnation will produce oscillating phases of weight increase and reduction, such that the total weight can never exceed $3\times$ the total number of clauses plus the total number of false clauses (for more detail see [8]).

3 Experimental Study

Initial empirical evaluations of DDFW and DDFW$^+$ on a range of structured and unstructured SAT problems have shown both algorithms can outperform AdaptNovelty$^+$ and RSAPS, and that DDFW$^+$ has a better average performance than DDFW [8]. However, a similar study has yet to be attempted in the MAX-SAT domain. To this end we decided to compare both DDFW variants with AdaptNovelty$^+$, RSAPS and PAWS10 (PAWS10 is a fixed parameter version of PAWS with the Max_{inc} parameter set at 10). Our first criteria was that manual parameter tuning is not allowed. This practical consideration means that the best clause weighting algorithms must be limited to using fixed parameter values (as in PAWS10) or to using an adaptive mechanism (as in RSAPS). We further chose AdaptNovelty$^+$ as our example WalkSAT algorithm because it has performed consistently well in the last three SAT competitions (see http://www.satcompetition.org).

To evaluate the relative performance of these algorithms we followed the strategy used to compare SAPS with iterated local search and GLSSAT2 in [2] and divided our empirical study into three problem categories:

***bor* problem set:** the bor-ku problem set comprises of a range of unweighted unsatisfiable MAX-2-SAT (bor-2) or MAX-3-SAT (bor-3) instances used in [2] and first described by Borcher in [9]. We used all 17 bor-2 and 11 bor-3 unsatisfiable unweighted instances from this set.

***jnh* problem set:** the jnh problems are generated with n variables. Each variable is added to a clause with probability $1/n$ and each literal is negated with probability 1/2 (unit and empty clauses are then deleted). We used 33 unsatisfiable jnh instances taken from the SATLIB library (www.satlib.org) each with 100 variables and between 800 and 900 clauses.

***uuf* problem set:** the uuf problems are unsatisfiable uniform random 3-SAT instances generated with n variables and k clauses with each clause containing 3 literals randomly chosen from the $2n$ possible literals. Tautological and duplicate literals are eliminated. We used the same 49 unsatisfiable instances reported in [2] each containing 200 variables and 1000 clauses.

All experiments were performed on a Dell machine with 3.1GHz CPU and 1GB memory. Cut-offs were set as follows: first RSAPS and DDFW were given a trial on each problem with a flip cut-off of 10,000,000. The best cost reached by either algorithm was then recorded and used as a target cut-off for further runs. All algorithms were allowed 100 trials for each instance and terminated either upon reaching the target cost (C_{target} in Algorithm 1) or after 3 seconds per trial.

***bor* Problem Results:** Taken in combination, the graphs in Figure 1 (a and b) and the statistics in Table 1 show DDFW to have the best overall performance on both the bor-2 and bor-3 problem classes. In the bor-2 graph (Figure 1a) DDFW and RSAPS clearly outperform the other techniques with DDFW having a slight advantage. The bor-2 table results help to separate DDFW and RSAPS further by showing DDFW not only to be twice as fast as RSAPS on average, but also to be consistently better than RSAPS on a problem by problem basis (as shown by the 88.24% Winner value). Interestingly, DDFW$^+$ performed quite poorly on this problem class, indicating there may be something in the two-literal clause structure of the bor-2 problems that causes the heuristic to be counter-productive. The bor-3 graph results in Figure 1b are less conclusive, as all algorithms were able to achieve near 100% success, although DDFW and DDFW$^+$ both exhibit superior performance during the first 0.04 seconds. Again, the table results help to clarify that DDFW is the better algorithm, showing it to have the best performance on 68.68% of instances, with DDFW$^+$ coming a close second in terms of average time.

***jnh* Problem Results:** Both the graph in Figure 1c and the table results show a mixed picture for the jnh problems. In the graph, AdaptNovelty$^+$ and DDFW$^+$

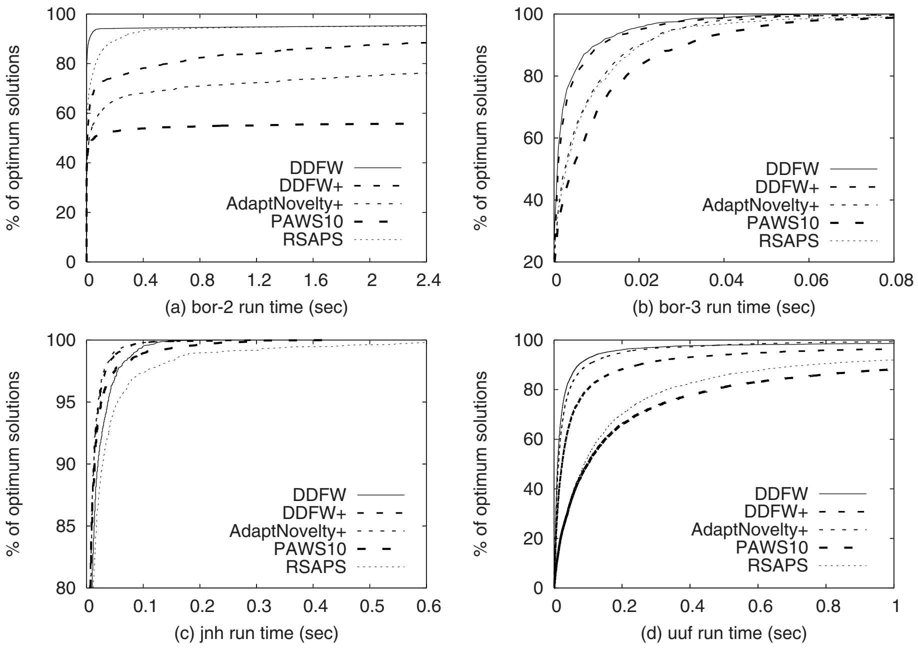


Fig. 1. Graphs of comparative time performance for all problem sets (% optimum refers to the proportion of solutions that reached the target solution cost)

emerge as equal best performers in the first 0.1 seconds of execution but are joined by DDFW after 0.2 seconds, with PAWS10 starting well but trailing off and RSAPS coming last. In the table, both AdaptNovelty$^+$ and DDFW$^+$ achieve similar results, with AdaptNovelty$^+$ having a slight advantage in average time, but DDFW$^+$ doing better on flips and the % Winner value. The surprise is that PAWS10 achieves the best result on 48.48% of jnh instances, despite being beaten in terms of average time. This can be explained by PAWS10 taking a longer than average on the more difficult jnh instances. Overall, therefore, there is little to chose between DDFW$^+$ and AdaptNovelty$^+$ on these problems, although they both do better than DDFW, PAWS10 and RSAPS.

***uuf* Problem Results:** The uuf results are similar to the bor-2 results, only this time DDFW has a very similar curve in Figure 1d to AdaptNovelty$^+$ (rather than RSAPS). In this case DDFW starts well but is caught by AdaptNovelty$^+$ after 0.5 seconds, after which AdaptNovelty$^+$ has a slight advantage. This is confirmed by the slightly better uuf success rate for AdaptNovelty$^+$ in Table 1. However, the table also shows DDFW to have the better average time and to have achieved the best result on 67.35% of the instances. Taking this into consideration, we can conclude that DDFW has the better uuf performance, with AdaptNovelty$^+$ coming a close second, followed by DDFW$^+$.

Table 1. Averaged results for each problem set; the time and flip statistics were measured over runs that reached the target solution cost; % Target measures the proportion of runs that reached the target solution cost; % Winner measures the proportion of problems for which an algorithm obtained the best result

Problem	Method	Mean Flips	Mean Seconds	% Target	% Winner
bor-2	DDFW	**7,719.62**	**0.01777**	**95.41**	**88.24**
	DDFW$^+$	79,308.65	0.15745	88.59	0.00
	AdaptNovelty$^+$	125,590.40	0.19005	77.24	0.00
	RSAPS	28,943.91	0.04536	95.06	11.76
	PAWS10	28,688.04	0.05131	55.82	0.00
bor-3	DDFW	**1,890.02**	**0.00385**	**100.00**	**63.63**
	DDFW$^+$	2,156.29	0.00440	99.91	18.18
	AdaptNovelty$^+$	4,449.40	0.00717	99.73	9.09
	RSAPS	4,798.05	0.00793	99.91	9.09
	PAWS10	6,109.10	0.01093	99.91	0.00
jnh	DDFW	3,721.65	0.00833	100.00	6.06
	DDFW$^+$	**2,376.13**	0.00553	100.00	21.21
	AdaptNovelty$^+$	2,870.73	**0.00546**	100.00	18.18
	RSAPS	7,724.26	0.01489	100.00	6.06
	PAWS10	3,747.89	0.00732	100.00	**48.48**
uuf	DDFW	**15,453.06**	**0.03294**	98.80	**67.35**
	DDFW$^+$	35,161.40	0.07335	96.98	4.08
	AdaptNovelty$^+$	32,441.44	0.04965	**99.92**	28.57
	RSAPS	97,412.31	0.16765	93.80	0.00
	PAWS10	167,275.80	0.30023	95.76	0.00

3.1 Analysis

Overall, the results show that DDFW has the best performance on our chosen problem set. More specifically, DDFW has the better performance on three out of four of the problem classes, emerging as a clear winner on the bor-2 and bor-3 problem sets, slightly better than AdaptNovelty$^+$ on the uuf problems but defeated by AdaptNovelty$^+$ and DDFW$^+$ on the jnh problems. DDFW's closest rivals are AdaptNovelty$^+$ and DDFW$^+$, with AdaptNovelty$^+$ defeating DDFW$^+$ on the uuf problems, equalling it on the jnh problems and being defeated by DDFW$^+$ on the bor-2 and bor-3 problems.

These results partly confirm the initial evaluation of DDFW and DDFW$^+$ in the SAT domain [8]. There, both algorithms were shown to have better performance in comparison to several other state-of-the-art SAT algorithms (including AdaptNovelty$^+$ and RSAPS) on a range of SAT competition benchmark problems. However, it was concluded for SAT that the DDFW$^+$ heuristic was a generally helpful addition to DDFW, causing superior performance in several problem categories while remaining neutral in the others. In contrast, our MAX-SAT study has found the DDFW$^+$ heuristic to have a negative effect on performance on the bor-2, bor-3 and uuf problems, only helping for the jnh problems. This brings the general applicability of the DDFW$^+$ heuristic under question. There is certainly no sign that it is useful for MAX-SAT problems and it could be that the promising results in the SAT domain were produced simply by a fortuitous selection of problems. We leave answering this question as a matter for further research.

4 Conclusions

The main contribution of this paper is to have evaluated and extended the weight redistribution strategy of the DDFW heuristic to the domain of unweighted MAX-SAT problems. In the process we have shown that DDFW outperforms several class-leading local search algorithms on a range of previously studied MAX-SAT benchmark problems. These results strengthen the case for the use of weight redistribution in clause weighting, as opposed to using the two-stage increase/reduce strategies of PAWS and SAPS. As with earlier studies, DDFW's performance has been shown to be more robust than RSAPS or a fixed parameter PAWS, making it the most promising approach for situations where manual parameter tuning is impractical.

Acknowledgements. We thankfully acknowledge the financial support from NICTA and the Queensland government. NICTA is funded by the Australian Government's *Backing Australia's Ability* initiative, and in part through the Australian Research Council.

References

1. Selman, B., Levesque, H., Mitchell, D.: A new method for solving hard satisfiability problems. In: AAAI 1992. Proc. 10th National Conference on AI, pp. 440–446 (1992)
2. Tompkins, D.A., Hoos, H.H.: Scaling and probabilistic smoothing: Dynamic local search for unweighted MAX-SAT. In: AI-2003. Proc. 16th Conference of the Canadian Society for Computational Studies of Intelligence, pp. 145–159 (2003)
3. Hoos, H.H.: An adaptive noise mechanism for WalkSAT. In: AAAI 2002. Proc. 18th National Conference, pp. 635–660 (2002)
4. McAllester, D.A., Selman, B., Kautz, H.A.: Evidence for invariants in local search. In: AAAI 1997. Proc. 14th National Conference, pp. 321–326 (1997)
5. Hoos, H.H.: On the run-time behaviour of stochastic local search algorithms for SAT. In: AAAI 1999. Proc. 16th National Conference, pp. 661–666 (1999)
6. Thornton, J.: Clause weighting local search for SAT. Journal of Automated Reasoning 35(1-3), 97–142 (2005)
7. Hutter, F., Tompkins, D.A.D., Hoos, H.H.: Scaling and probabilistic smoothing: Efficient dynamic local search for SAT. In: Van Hentenryck, P. (ed.) CP 2002. LNCS, vol. 2470, pp. 233–248. Springer, Heidelberg (2002)
8. Ishtaiwi, A., Thornton, J., Anbulagan, S.A., Pham, D.N.: Adaptive clause weight redistribution. In: Benhamou, F. (ed.) CP 2006. LNCS, vol. 4204, pp. 229–243. Springer, Heidelberg (2006)
9. Borchers, B., Furman, J.: A two-phase exact algorithm for MAX-SAT and weighted MAX-SAT problems. Journal of Combinatorial Optimization 2, 299–306 (1999)

A HMM-Based Hierarchical Framework for Long-Term Population Projection of Small Areas

Bin Jiang[1], Huidong Jin[1,2], Nianjun Liu[1,2], Mike Quirk[3], and Ben Searle[3,4]

[1] The Australian National University, Canberra, Australia
bjiang@rsise.anu.edu.au
[2] NICTA, Canberra Lab, Locked Bag 8001, ACT 2601 Australia
Huidong.Jin@nicta.com.au, nianjunl@rsise.anu.edu.au
[3] ACT Planning and Land Authority
Mike.Quirk@act.gov.au, Ben.Searle@ga.gov.au
[4] Geoscience Australia

Abstract. Population Projection is the numerical outcome of a specific set of assumptions about future population changes. It is indispensable to the planning of sites as almost all successive planning activities such as the identification of land and housing supply, the release of land, the planning and construction of social and physical infrastructure are population related. This paper proposes a new hierarchical framework based on Hidden Markov Model (HMM), called HMM-Bin framework, for use in long-term population projection. Analyses of various existing suburbs indicate it outperforms traditional Cohort Component model and simple HMM in terms of less data dependency, output flexibility and long-term projection accuracy.

1 Introduction

Population Projection is the numerical outcome of a specific set of assumptions regarding future population changes [6]. The numerical outcome could be the total population of a specific area and sometimes can be further partitioned into different age and gender cohorts like 0-4 males and 5-9 females. The projection can be made either for 5 to 10 years or for a longer period, such as 20 or more years. The target area could be as large as a country or as small as a suburb.

Population Projection is indispensable because of the following points. Firstly, it plays a key role for government agencies to cost-effectively support development by the timely delivery of infrastructure, facilities and services. For example, in the development plan of a new suburb, the government has to determine whether it is necessary to release land to construct a new shopping centre and, if necessary, when, where and what size of land should be released then. Secondly, according to laws, population projection is compulsory, say, for land planning in ACT [1]. Thirdly, population projection results are also widely used by the private sectors, including retailers, property developers and investors. To predict population, we have to take account of a number of social, economic and political factors. It is almost impossible without certain assumptions [2].

A long-standing approach to conduct population projection is the so-called Cohort-Component Model (also called CC model hereafter) [6]. However, this model requires high quality data, which are often unavailable in many places especially for *small areas* like suburbs. Moreover, if a long-term population projection for twenty or more years is required, it is difficult to make appropriate assumptions, especially for the migration. In addition, this model only produces point values [1]. Such a kind of projection is not expressive enough, especially for decision-making support. Details about this model and our assumptions can be found in [2].

To overcome these limitations, we have developed a new HMM-based hierarchical framework in this paper. To mitigate data dependency such as migration rates, it describes population sizes as observations and takes other factors combined as a hidden variable. It estimates the complicated relationship between hidden states and observation from historical data. It also uses the population projections made by Australian Bureau of Statistics (ABS) for larger areas (LAs) to guide the projection for smaller areas. It can produce point values, as well as intervals or population probability distributions. We will illustrate our model on Canberra suburbs with a 20-year projection. Following ABS conventions, we call Canberra suburbs Statistical Local Areas (SLAs) hereafter.

A HMM is a joint statistical model for an ordered sequence of variables. It is the result of stochastically perturbing the variables in a Markov chain (the original variables are thus "hidden"). The Markov chain has discrete variables which indicate the "state" of the HMM at each step. The perturbed values can be continuous and are the "outputs" of the HMM. HMMs are commonly used in speech recognition [5]. A HMM includes the following elements: N, the number of hidden states in the model; M, the number of observation symbols corresponding to each state; A, the state transition probability distribution; B, the observation symbol probability distribution, so-called the observation probability distribution matrix; and π, the initial state distribution. More specific descriptions can be found in [5].

In the rest of this paper, we describe the proposed framework in Sec. 2 followed by the experiments in Sec. 3 and conclusions in Sec. 4.

2 The Proposed HMM-Based Hierarchical Framework

It is not trivial to applying HMM for population projection, as our preliminary results in Sect. 3 indicate a plain HMM performs quite bad. Actually, we need to overcome several problems including preparing appropriate observation variables for representation and training, the diversity of SLAs, the number of hidden states and constraints for projection. We propose a hierarchical framework based on HMM (termed HMM-Bin hereafter) to overcome these problems. As illustrated in Fig. 1, HMM-Bin has different HMMs for projection for different kinds of SLAs, automatically grouped by clustering. It uses population

[1] a series of specific values.

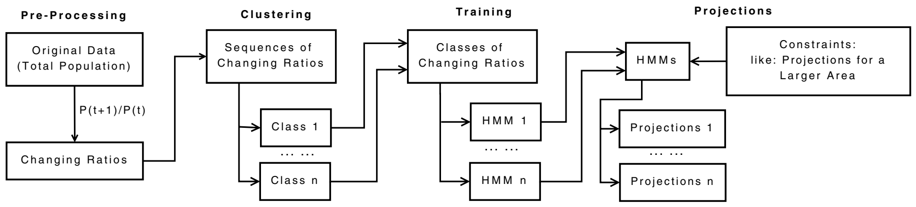

Pre-Processing
Clustering
Training
Projections
Original Data (Total Population)
P(t+1)/P(t)
Changing Ratios
Sequences of Changing Ratios
Class 1
Class n
Classes of Changing Ratios
HMM 1
HMM n
HMMs
Constraints: like: Projections for a Larger Area
Projections 1
Projections n

Fig. 1. A high-level view of our proposed HMM-Bin framework

projections for a larger area to guide projection for smaller areas. We describe how this framework overcomes these problems as below.

Pre-processing. The normally available data is total population by age and gender for a target area or a set of SLAs. However, because different SLA has different geographical size and hence can support different size of people, we use $r_{t+1} = \frac{P(t+1)}{P(t)}$, where $P(t)$ is the total population value at year t instead to avoid the absolute total population size and uncover something in common: *degrees of change.*

Clustering. Given the changing ratios, the degrees of change, some SLAs' population increases, some SLAs' population decreases while some SLAs' population fluctuates greatly. It is acceptable to take all the SLAs' changing ratio sequences as a whole class to train a plain HMM, but obviously it is not natural because differences of population change pattern between different SLAs do exist. So our solution is clustering these changing ratio sequences using clustering method like K-Means clustering algorithm with Euclidean Distance [4]. We have also experimented *WITHOUT* this step and the results (see Sec. 3) show that this step is quite necessary.

Training. After grouping SLAs into different classes, their changing ratios will also be grouped into different classes and each class has a class centroid sequence with the same length. According to the quantity of sequences within each class, we decide whether or not to build a HMM for that class. If the training data is insufficient, we will skip that class since HMM training needs adequate training data. We use Baum-Welch method for training a HMM. To use this method, we have to specify two things first as below.

Type of Observations: Since Cohort-Component method produces very specific point values of population size and its results have no flexibility at all. Therefore, it is necessary to produce more flexible results like intervals or population probability distribution. Provided that any probability distribution could be effectively approximated by a mixture of Gaussian distributions [3], it is arguable to adopt mixtures of Gaussians as our HMM observation type.

Number of Hidden States: The number of hidden states can be determined by examining the histogram of the changing ratios of different classes. Our approach is: (1) visualise the changing ratios with histograms and have an initial guess of how many Gaussian distributions needed to cover them; (2)

then do the training to see, within these initial guess values, which one will generate higher likelihood and less overlap among Gaussian distributions.

Projection. After the training step, the classes having sufficient training data will get their Hidden Markov Models with trained parameter sets. Here are the projection steps: (1) Fit out the relationship between the LA's historical changing ratio sequence and class centroid sequences: centroid sequence = f(LA's historical changing ratio sequence); (2) Get the future class centroid sequences using LA's future changing ratios and the fitted function f; (3) Use the projected centroid sequences as observations to guess the future hidden state sequences for different classes of SLAs in the target future period; (4) Use the projected hidden state sequences, we can get the corresponding future observations, which are then formed to be intervals of 95% confidence level. Since what we have projected are changing ratios, in order to get population sizes, we have to multiply the changing ratios with previous years' population sizes.

3 Experiments and Discussion

We examine HMM-Bin on existing Canberra suburbs and compare it with a plain HMM and the CC model. Our experiments are based on ABS' data for ACT SLAs' historic total population sizes from 1986 to 2000, ACT historic total population sizes from 1994 to 2000 and ACT population projections 2001 to 2020. Our target period is 20 years from 2001 to 2020. We use K-Means clustering algorithm with Euclidean Distance and Kevin Murphy's HMM MATLAB toolbox. The larger area (LA) discussed in the projection step is ACT in this case and its population projections made by ABS are used as constraints.

We only conduct projections for two classes [2](Class 1 and Class 2) that have enough training data. The ABS population estimations for these SLAs for 2001 to 2004 have been used as ground-true values. There are 92.50% out of Class 1 and 95.65% out of Class 2 SLAs have been covered by HMM-Bin projections at the confidence level of 95% while the coverage percentage for NON-Clustering is just 52.05%. Besides the coverage comparison, we also compare the relative difference between the mean sequences generated by HMM-Bin and the ground-true values. If we take 4% relative difference as a threshold, Class 1 and Class 2 have over 70% SLAs within this threshold while there is no SLA from the all-in-one Class within this threshold. The majority ($\geq$ 80%) of Class 1 and Class2 SLAs have relative differences within 6% while that of all-in-one Class have relative differences within 10%. These results show the weakness of a plain HMM directly applied *WITHOUT* the clustering step.

The comparison with CC model also has been done to several SLAs and details can be found in [2]. In the case of long-term projection, HMM-Bin projections are smoother than CC results which change more rapidly. HMM-Bin only produces 10% to 13% relative differences while CC produces over 40% relative difference

[2] After clustering, there are 6 classes of SLAs but only 2 of them have enough training data.

compared with the mean sequences. The reason why CC changes rapidly is because it is so hard to make assumptions about the future population changes especially for the migration part, for which it is almost impossible to predict accurately.

4 Conclusion

In this paper, we have proposed a HMM-based hierarchical framework for long-term population projection. We have evaluated it on various Canberra suburbs and compared with the traditional CC Model and a plain HMM. The HMM-Bin framework could generate more accurate population projections comparing some ground-true data from ABS. Moreover, the HMM-Bin framework has low dependency of data availability. This is quite useful for those *small areas* lacking of high quality of census data like suburbs in Australia. Furthermore, HMM-Bin framework produces flexible outputs in the form of intervals (population probability distribution in general), instead of specific point values generated by other models like the traditional CC model. Last but not least, because HMM-Bin framework has less requirements of data, it is more suitable than CC model for long-term projections for small areas like suburbs. Actually, it is quite hard to make assumptions especially for migration for CC model and these assumptions normally have low accuracy. Future effort may be put in enabling HMM-Bin to handle disaggregate population components.

Acknowledgment

We are grateful to Mr. Phil Jorritsma from ACT-PLA [3] for help in domain knowledge. We also appreciate Australian Bureau of Statistics for free data and financial support from ACT-PLA and NICTA, which is funded by the Australian Government's Backing Australia's Ability initiative, in part through the Australian Research Council.

References

1. ACT Parliamentry Counsel. Land (Planning and Environment) Act (2007)
2. Jiang, B.: Better Long-term Population Projection. Master's thesis, the Australian National University (June 2007), Available at http://feng.marco.jiang.googlepages.com
3. Jin, H.-D., Leung, K.-S., Wong, M.-L., Xu, Z.-B.: Scalable model-based cluster analysis using clustering features. Pattern Recognition 38(5), 637–649 (2005)
4. MacQueen, J.B.: Some Methods for classification and Analysis of Multivariate Observations. In: Proceedings of 5-th Berkeley Symposium on Mathematical Statistics and Probability, pp. 281–297 (1767)
5. Rabiner, L.R.: A tutorial on Hidden Markov Models and selected applications in speech recognition. Proceedings of the IEEE 77(2), 257–286 (1989)
6. Smith, S.K., Tayman, J., Swanson, D.A.: State and Local Population Projections: Methodology and Analysis. Kluwer Academic Plenum Publishers, Boston, MA (2001)

[3] ACT Planning and Land Authority.

Backbone of the p-Median Problem

He Jiang, XianChao Zhang, and MingChu Li

School of Software, Dalian University of Technology, 116621 Dalia, P.R. China
{jianghe, xczhang, mingchul}@dlut.edu.cn

Abstract. PMP is a well-known NP-hard problem with extensively wide applications in location science and clustering. In this paper, we presented computational complexity results about the backbone, the shared common parts of all the optimal solutions to the PMP. We showed that it is intractable to approximate the backbone of the PMP with any performance guarantee under the assumption that $P \neq NP$.

Keywords: p-median, computational complexity, backbone.

1 Introduction

Given a set of users and potential facilities, the p-median problem (PMP) is to identify the locations of a pre-determined number of facilities, in such a way as to minimize the total distance that user must traverse to reach its nearest facility. The PMP is a well-known NP-hard problem (Kariv and Hakimi, 1979), with wide applications in location science and clustering. Since it is intractable for NP-hard problems to obtain optimal solutions, many heuristics have been developed for solving large PMP instances. A recent survey on the state of the art in this area can be found in Mladenović et al. (2007). As an important tool to design heuristics for NP-hard problems, the backbone (i.e., the shared common parts of all optimal solutions for a NP-hard problem instance) has received widespread attention because no optimal solution can be computed unless the backbone is obtained. However, backbone analysis is still at the beginning state with few theoretical analysis results. As to our knowledge, the only analytical result is given by Kilby, Slaney, and Walsh (2005) that it is NP-hard to obtain the backbone of the TSP. Its basic idea is to map any instance of the TSP to an instance whose solution is unique. Therefore, finding the backbone of the modified instance is equivalent to finding an optimal solution of the original instance.

This paper investigates the computational complexity of the backbone of the PMP as follows. Firstly, we proved that there is no polynomial time algorithm to obtain the full backbone of the PMP under the assumption that $P \neq NP$. The same principle by Kilby, Slaney, and Walsh (2005) was used in the proof. We gave the definition of the compact PMP instance and showed that its backbone is the same to the backbone of the related PMP instance. By constructing the biased compact PMP instance with unique optimal solution, we proved it is intractable to approximate the backbone assuming that $P \neq NP$. Secondly, we indicated that it is intractable to obtain a fixed fraction of the backbone under the assumption that $P \neq NP$ as well.

2 Preliminaries

In this section, we shall give some related definitions and related properties.

Definition 1. Given a set $F=\{1,2,...,m\}$ of m potential facilities, and a set $U=\{1,2,...,n\}$ of n users, a matrix $D=(d_{ij})_{n\times m}$ where d_{ij} represents the distance the user u_i away from the facility f_j for all $i\in U$ and $j\in F$, a predefined $p<m$, the PMP instance is denoted as $PMP(F,U,D,p)$. A *feasible solution* to $PMP(F,U,D,p)$ is a subset $J\subset F$, $|J|=p$, whose cost function is defined as $C_J(F,U,D,p)=\sum_{i\in U}\min_{j\in J} d_{ij}$. In the PMP, the goal is to find a feasible solution J^* that minimizes the cost, i.e., $C_{J^*}(F,U,D,p)=\min(\{C_J(F,U,D,p)\mid J\in\Pi\})$, where Π is the set of all the feasible solutions.

Definition 2. Given a PMP instance $PMP(F,U,D,p)$, let $\Pi^*=\{J_1^*,J_2^*,...,J_q^*\}$ be the set of all optimal solutions to $PMP(F,U,D,p)$, where $q=|\Pi^*|$ represents the number of optimal solutions. *Backbone* of the PMP instance $PMP(F,U,D,p)$ is defined as $bone(F,U,D,p)=J_1^*\cap J_2^*\cap\cdots\cap J_q^*$.

The backbone $bone(F,U,D,p)$ is essential for heuristic algorithms design, since a heuristic cannot obtain an optimal solution to a PMP instance unless it obtains the full backbone $bone(F,U,D,p)$. On the other hand, if the backbone $bone(F,U,D,p)$ is obtained, the search space could then be effectively reduced by assigning users in the backbone to some locked facilities.

In Definition 3, Definition 4, and Definition 5, some new conceptions are introduced for later proof of the two main theorems in this paper.

Definition 3. Given a PMP instance $PMP(F,U,D,p)$, a *compact solution* is a subset $J\subset F$ ($|J|\le p$) such that $\forall j\in J,\sum_{i\in U}\min_{k\in J-\{j\}} d_{ik}>\sum_{i\in U}\min_{k\in J} d_{ik}$. Given a PMP instance $PMP(F,U,D,p)$, the *compact PMP instance* aims to find a compact solution $\tilde{J}^*$ that minimizes the cost, i.e., $C_{\tilde{J}^*}(F,U,D,p)=\min(\{C_J(F,U,D,p)\mid J\in\tilde{\Pi}\})$, where $\tilde{\Pi}$ is the set of all the compact solutions. In this paper, the compact PMP instance of $PMP(F,U,D,p)$ is denoted as $cPMP(F,U,D,p)$.

Property 1. Given a compact PMP instance $cPMP(F,U,D,p)$, the cost of optimal solutions to $cPMP(F,U,D,p)$ equals to that of optimal solutions to $PMP(F,U,D,p)$.

Definition 4. Given a compact PMP instance $cPMP(F,U,D,p)$, let $\tilde{\Pi}^*=\{\tilde{J}_1^*,\tilde{J}_2^*,...,\tilde{J}_r^*\}$ be the set of all the optimal solutions to $cPMP(F,U,D,p)$, where $r=|\tilde{\Pi}^*|$ represents the number of optimal solutions. *Backbone* of the compact PMP instance $cPMP(F,U,D,p)$ is defined as $cbone(F,U,D,p)=\tilde{J}_1^*\cap\tilde{J}_2^*\cap\cdots\cap\tilde{J}_r^*$.

Property 2. Given a PMP instance $PMP(F,U,D,p)$ and its compact PMP instance $cPMP(F,U,D,p)$, $cbone(F,U,D,p) = bone(F,U,D,p)$.

Definition 5. Given a compact PMP instance $cPMP(F,U,D,p)$, the *biased matrix* of D is defined as $\hat{D} = (\hat{d}_{ij})_{n\times m}$, where $\hat{d}_{ij} = d_{ij} + 1/2^{(i+j)mn+|i-j|}$ for all $i \in U$ and $j \in F$. In this paper, the *biased compact PMP instance* of $cPMP(F,U,D,p)$ is defined as $cPMP(F,U,\hat{D},p)$.

3 Computational Complexity of Backbone

Without loss of generality, we assume that the entries d_{ij} in the matrices D are nonnegative integers, i.e., $d_{ij} \in Z^+ \cup \{0\}$ for all $i \in U$ and $j \in F$. In Lemma 1 and Lemma 2, we shall study the relationship between the solution of a compact PMP instance and that of the biased compact PMP instance.

Lemma 1. Given the matrix $D = (d_{ij})_{n\times m}$ ($d_{ij} \in Z^+ \cup \{0\}$), for any two distinct solutions J_1, J_2 to the compact PMP instance $cPMP(F,U,D,p)$, if $C_{J_1}(F,U,D,p) < C_{J_2}(F,U,D,p)$ then $C_{J_1}(F,U,\hat{D},p)$ $< C_{J_2}(F,U,\hat{D},p)$ for the biased compact PMP instance $cPMP(F,U,\hat{D},p)$.

Proof. By the assumption of Lemma 1, we have $C_{J_2}(F,U,D,p) - C_{J_1}(F,U,D,p) \in Z^+$. By definition of the biased compact PMP instance, for any feasible solution J, we have

$$C_J(F,U,\hat{D},p) = \textstyle\sum_{i\in U} \min_{j\in J} \hat{d}_{ij} = \sum_{i\in U} \min_{j\in J} (d_{ij} + 1/2^{(i+j)mn+|i-j|}).$$

Since $d_{ij} \in Z^+ \cup \{0\}$ and $0 < 1/2^{(i+j)mn+|i-j|} \le 1/2^{2mn}$ for all $i \in U$ and $j \in J$, we have $\min_{j\in J} d_{ij} < \min_{j\in J} (d_{ij} + 1/2^{(i+j)mn+|i-j|}) \le \min_{j\in J}(d_{ij}) + 1/2^{2mn}$ for $i \in U$. Thus, $C_J(F,U,D,p) < C_J(F,U,\hat{D},p) \le C_J(F,U,D,p) + \sum_{i\in U} 1/2^{2mn}$.

If $C_{J_1}(F,U,D,p) < C_{J_2}(F,U,D,p)$ for any two distinct solutions J_1, J_2, we have $C_{J_2}(F,U,\hat{D},p)$ $- C_{J_1}(F,U,\hat{D},p) \ge C_{J_2}(F,U,D,p) - C_{J_1}(F,U,D,p) - \sum_{i\in U} 1/2^{2mn}$.

Since $0 < \sum_{i\in U} 1/2^{2mn} = n/2^{2mn} < 1$, $C_{J_2}(F,U,D,p) - C_{J_1}(F,U,D,p) \ge 1$, we have $C_{J_2}(F,U,\hat{D},p) -$ $C_{J_1}(F,U,\hat{D},p) > 0$. Thus, this lemma is proved. □

Lemma 2. Given the matrix $D = (d_{ij})_{n\times m}$ ($d_{ij} \in Z^+ \cup \{0\}$), if $C_{J_1}(F,U,D,p) = C_{J_2}(F,U,D,p)$ for any two distinct solutions J_1, J_2 to the compact PMP instance $cPMP(F,U,D,p)$, then $C_{J_1}(F,U,\hat{D},p)$ $\ne C_{J_2}(F,U,\hat{D},p)$ for the biased compact PMP instance $cPMP(F,U,\hat{D},p)$.

Proof. For any feasible solution J, we have $0 < 1/2^{(i+j)mn+|i-j|} \le 1/2^{2mn}$ for all $i \in U$ and $j \in J$. Let $K(J,i) = \{k \mid d_{ik} = \min_{j\in J}(d_{ij})\}$ for all $i \in U$, we have $\min_{j\in J}(d_{ij} + 1/2^{(i+j)nm+|i-j|}) = \min_{j\in J}(d_{ij}) + \min_{k\in K(J,i)} 1/2^{(i+k)nm+|i-k|}$.

Therefore, if $C_{J_1}(F,U,D,p) = C_{J_2}(F,U,D,p)$ for any two distinct solutions J_1, J_2 to the compact PMP instance $cPMP(F,U,D,p)$, then we have

$$C_{J_2}(F,U,\hat{D},p) - C_{J_1}(F,U,\hat{D},p)$$
$$= \sum_{i\in U}(\min_{k\in K(J_2,i)} 1/2^{(i+k)mn+|i-k|} - \min_{k\in K(J_1,i)} 1/2^{(i+k)mn+|i-k|}) .$$

When viewed as binary encoded strings, the integer part of $C_{J_2}(F,U,\hat{D},p)$ equals to that of $C_{J_1}(F,U,\hat{D},p)$. Thus, we only need to consider the fractional part of these binary strings.

Since $J_1 \neq J_2$, thus we have $J_1 \cup J_2 \setminus J_1 \cap J_2 \neq \varnothing$. Therefore, there must be $j^* \in (J_1 \cup J_2 \setminus J_1 \cap J_2)$. In the following proof, we consider the case that $j^* \in J_1$ and $j^* \notin J_2$. For the case that $j^* \in J_2$ and $j^* \notin J_1$, it can be proven in a similar way.

According to Definition 1, there must exist $i^* \in U$ such that $d_{i^* j^*} < d_{i^* j}$ for all $j \in J_1 \setminus \{j^*\}$. Therefore, we have

$$C_{J_2}(F,U,\hat{D},p) - C_{J_1}(F,U,\hat{D},p) = \sum_{i\in U-\{i^*\}}(\min_{k\in K(J_2,i)} 1/2^{(i+k)mn+|i-k|}$$
$$- \min_{k\in K(J_1,i)} 1/2^{(i+k)mn+|i-k|}) + \min_{k\in K(J_2,i^*)} 1/2^{(i^*+k)mn+|i^*-k|} - 1/2^{(i^*+j^*)mn+|i^*-j^*|} ,$$

i.e., the $(i^*+j^*)mn+|i^*-j^*|$ *th* bit of fractional part of $C_{J_1}(F,U,\hat{D},p)$ will be 1, however the $(i^*+j^*)mn+|i^*-j^*|$ *th* bit of fractional part of $C_{J_2}(F,U,\hat{D},p)$ will be 0. Hence, we have $C_{J_1}(F,U,\hat{D},p) \neq C_{J_2}(F,U,\hat{D},p)$.

Thus, this lemma is proved. □

By Lemma 1 and Lemma 2, the following theorem can be easily proved.

Theorem 1. Given the matrix $D = (d_{ij})_{n\times m}$ ($d_{ij} \in Z^+ \cup \{0\}$), there is one and exactly one optimal solution to the biased compact PMP instance $cPMP(F,U,\hat{D},p)$.

Corollary 1. Given the matrix $D = (d_{ij})_{n\times m}$ ($d_{ij} \in Z^+ \cup \{0\}$), the optimal solution to the biased compact PMP instance $cPMP(F,U,\hat{D},p)$ is also optimal to the compact PMP instance $cPMP(F,U,D,p)$.

Theorem 2. It is NP-hard to obtain the backbone of the compact PMP.

Proof. Otherwise, there must exist an algorithm denoted by Λ which is able to obtain the backbone of the compact PMP in polynomial time.

Given any arbitrary matrix $D = (d_{ij})_{n\times m}$ ($d_{ij} \in Z^+ \cup \{0\}$), the compact PMP instance $cPMP(F,U,D,p)$ could be transformed to its biased compact PMP instance $cPMP(F,U,\hat{D},p)$ by an algorithm denoted by Γ in $O(nm)$ time.

Since the $cPMP(F,U,\hat{D},p)$ is also a compact PMP instance, its backbone can be obtained by Λ in polynomial time (denoted by $O(\bullet)$). Also, by Theorem 1, the backbone is not only the unique optimal solution to $cPMP(F,U,\hat{D},p)$ itself, but also optimal to $cPMP(F,U,D,p)$.

Hence, any compact PMP instance can be exactly solved in $O(nm)+O(\bullet)$ time by Γ and Λ. It implies that any PMP instance can be exactly solved in polynomial time according to the proof of Property 2. Obviously, such conclusion contradicts with the result that the PMP is NP-hard by Kariv and Hakimi (1979). Thus, this theorem is proved. □

Corollary 2. It is NP-hard to obtain the backbone of the PMP.

Theorem 3. It is NP-hard to obtain a fixed fraction of the backbone of the compact PMP.

Proof. Otherwise, given any compact PMP instance $cPMP(F,U,D,p)$ ($d_{ij} \in Z^{+} \cup \{0\}$), we can construct a biased compact PMP instance $cPMP(F,U,\hat{D},p)$ according to Definition 5.

If there exists a polynomial time algorithm Λ to obtain a fixed fraction of the backbone, we can at least obtain a facility j^{*} in $cbone(F,U,\hat{D},p)$. Thus, we can construct a new biased compact PMP $cPMP(F\setminus\{j^{*}\},U,\hat{D}',p-1)$, where $\hat{d}'_{ij}=\hat{d}_{ij}$ for all $\hat{d}_{ij}\le\hat{d}_{ij^{*}}, j\in F\setminus\{j^{*}\}, i\in U$ and $\hat{d}'_{ij}=\hat{d}_{ij^{*}}$ for all $\hat{d}_{ij}>\hat{d}_{ij^{*}}$, $j\in F\setminus\{j^{*}\}$, $i\in U$. By repeat calls to Λ, an optimal solution to the original instance can be found in polynomial time, a contradiction. □

Corollary 3. There exists no polynomial time algorithm to obtain a fixed fraction of the backbone of the PMP unless $P=NP$.

4 Conclusions

As an essential tool for heuristics design of the PMP, analytical results on backbone were presented in this paper. We showed that it is intractable to approximate the backbone of the PMP with any performance guarantee under the assumption that $P\neq NP$.

Acknowledgments. This work was supported by the Natural Science Foundation of China (No.60673066, 60673046, 60503003), the Natural Science Foundation of LiaoNing Province (No.20051082), and the Gifted Young Foundation of DUT.

References

1. Kariv, O., Hakimi, S.L.: An algorithmic approach to network location problems, Part 2: The p-medians. SIAM Journal of Applied Mathematics 37, 539–560 (1979)

2. Kilby, P., Slaney, J., Walsh, T.: The backbone of the traveling salesperson. In: IJCAI 2005. Proceedings of the 19th International Joint Conference on Artificial Intelligence, Morgan Kaufmann Publishers, San Francisco (2005)
3. Mladenović, N., Brimberg, J., Hansen, P., Moreno-Pérez, J.: The p-median problem: a survey of metaheuristic approaches. European Journal of Operational Research 179(3), 927–939 (2007)

One Shot Associative Memory Method for Distorted Pattern Recognition

Asad I. Khan and Anang Hudaya Muhamad Amin

Clayton School of IT, Monash University, Clayton 3168 VIC, Australia
{Asad.Khan, Anang.Hudaya}@infotech.monash.edu.au

Abstract. In this paper, we present a novel associative memory approach for pattern recognition termed as Distributed Hierarchical Graph Neuron (DHGN). DHGN is a scalable, distributed, and one-shot learning pattern recognition algorithm which uses graph representations for pattern matching without increasing the computation complexity of the algorithm. We have successfully tested this algorithm for character patterns with structural and random distortions. The pattern recognition process is completed in one-shot and within a fixed number of steps.

Keywords: One-Shot Learning, Associative Memory, Pattern Recognition.

1 Introduction

In any pattern recognition systems, image measurement, feature extraction, and classification are three sub-problems that need to be addressed [1]. In this paper we are addressing the classification problem with regards to the pattern recognition systems. We would also introduce a finely distributed one-shot learning associative memory (AM) approach for distorted optical character recognition. Sussner and Valle have also implemented one shot learning - morphological associative memories (MAM) - but as a tightly coupled algorithm [2]. Some of the related character pattern recognition undertakings can be found in [3-5]. In this paper we will describe the implementation of a pattern recognition system using our distributed associative memory approach termed as Distributed Hierarchical Graph Neuron (DHGN). The DHGN algorithm offers unsupervised learning capability and in-network processing. It is useful for recognition process involving large number of patterns, with fast store/recall time.

2 Distributed Associative Memories for Pattern Recognition

DHGN is a novel one-shot learning AM algorithm which is based on the graph theory for pattern analysis and classification. It is an extension of the Graph Neuron (GN) algorithm [6, 7]. GN is an associative memory algorithm which is highly scalable and is able to match multiple streams of input with historical data within the network in real-time [8]. The pattern is stored as a graph-like structure where each atomic component of the structure has its own *(value, position)* representation as shown in Fig. 1.

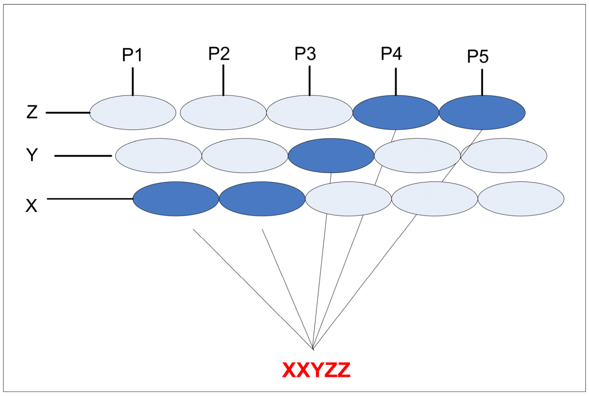


Fig. 1. Data representation within the GN nodes for an input pattern XXYZZ

The number of nodes required, X_{GN} , within the network can be calculated using the following equation:

$$X_{GN} = np \ . \tag{1}$$

Where n represents the possible values in a given pattern and p represents the pattern size. The pattern recognition process using GN algorithm occurs in three phases [8]: Input, Synchronisation, and Bias Array Update phases.

The scalability of the GN algorithm has been checked by testing it with up to 16,384 nodes. The test has shown that the computational complexity does not increase significantly with the increase in the size of the network [9]. The pattern crosstalk can occur within the simple GN implementation [8]. We propose an algorithm known as an extended GN approach such that it provides a bird's eye view of the overall pattern, as well as the capability to distribute the pattern analysis over multiple processors. This approach we term as Distributed Hierarchical Graph Neuron (DHGN). Figure 2 shows the overall architecture of the DHGN algorithm.

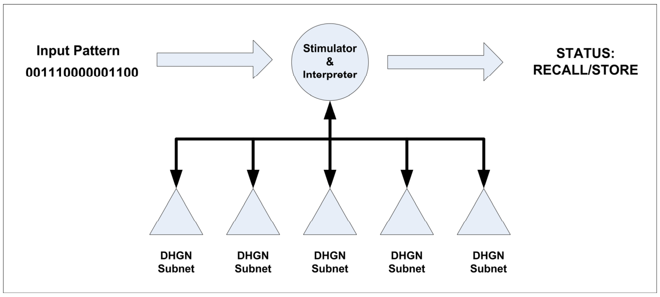


Fig. 2. DHGN architecture comprising the Stimulator & Interpreter (SI) and the GN arrays [8]

Input to the DHGN arrays is coordinated by the Stimulator and Interpreter (SI module). The SI sends out all the patterns to the DHGN network and consequently receives the result from each of the top layer nodes of DHGN sub-networks.

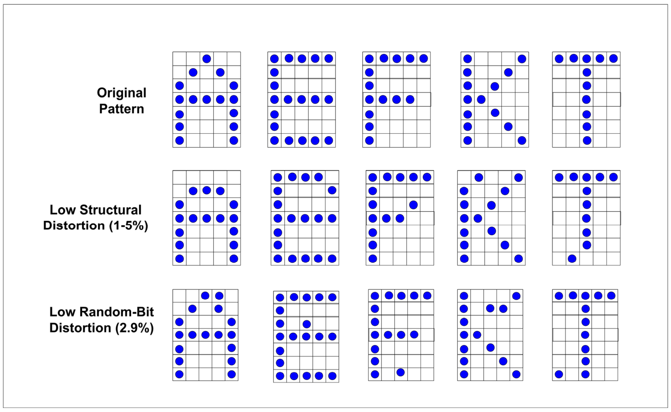
2.1 DHGN Sub-networks

DHGN algorithm provides the capability to decompose the pattern into sub-patterns and is able to conduct recognition processes at sub-pattern levels. The DHGN sub-networks, which we term as the compositions later on, independently analyse the pattern segments. Equation (2) shows the number of nodes X_{DHGN} requires, given the number of compositions m, with pattern size b, and possible values for the pattern a_i:

$$X_{DHGN} = \sum_{i=1}^{m} a_i \left(\frac{b_i + 1}{2} \right) . \qquad (2)$$

3 DHGN for Character Pattern Classification

Two different types of distortion have been taken into consideration. These comprise structural and random-bit distortions. We have developed a tool to implement the processes involved in pattern recognition using DHGN algorithm. The simulation tool has been developed to recognize patterns of 35-bit length. The character patterns used for this simulation are of 7 x 5 bit representations as shown in Figure 3.



Original Pattern
Low Structural Distortion (1-5%)
Low Random-Bit Distortion (2.9%)

Fig. 3. Character patterns with structural and random distortions

In our tests, the distortion levels for character patterns have been set to low, medium, and high.

3.1 Results and Discussion

In order to determine the recall rates of distorted character patterns, DHGN uses bias entry indices as indicators whether the pattern or sub-pattern is a recalled pattern or a new pattern. Equation (3) shows the DHGN recall rate P_{RECALL} calculation. GN_i^{RECALL} represents the GN nodes that produce the recalled index entry, and GN_i^{TOTAL} represents the total number of GN nodes in a given sub-network.

$$P_{RECALL} = \sum_{i=1}^{m} \left(\frac{GN_i^{RECALL}}{GN_i^{TOTAL}} \right) . \tag{3}$$

Figure 4 show the results of the recognition process.

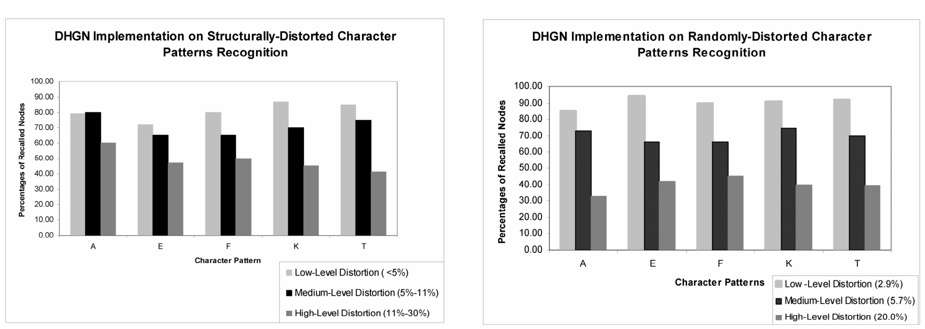


Fig. 4. DHGN pattern recognition with structural and random distorted character patterns

It may be seen from Figures 4 that an increase in the level of distortion reduces the ability to recognize the patterns correctly. It may also be noted that pattern recognition with random-bit distortions gives higher recognition rates as compared to structural distortions. This is owing to the fact that with structural distortions the shape of the character is often affected. Whilst with random-bit distortions, the structure of the characters often stays preserved.

We partially tested our hypothesis by applying DHGN to a higher bit representation. The character pattern 'A' was enhanced from a 7 x 5 bit representation to a 14 x 10 bit representation. The converted pattern was then tested for random-bit distortion. The results are shown in Table 1.

Table 1. Percentage of recalled nodes for two different pattern representations of character A with 2.9% random-bit distortion

Pattern	Recall Rate
7 x 5 – Bits Representation	79.0% (2.9% Distortion)
14 x 10 Bits Representation	85.0% (2.9% Distortion)

4 Conclusions

We have presented a one-shot learning associative memory method for distorted character pattern recognition. The effectiveness of the approach has been tested with relatively small-scale patterns at this stage. The tests show that our method can handle both structural and random pattern distortions. Our tests however show that DHGN perform better with random distortions. The lower accuracy in handling structural distortion is attributed to our implementation of DHGN, where two-dimensional patterns are processed as one-dimensional bit strings. Doing so prevents exploiting the structural relationships which exist within similar patterns. Our method clearly demonstrates that distorted pattern recognition can be implemented as a one-shot learning algorithm. The GN has been shown to be highly scalable with respect to the number of nodes in the network. We believe that structural distortions can be more accurately analysed by implementing two-dimensional DHGN compositions.

Acknowledgments. This work was supported through funding provided by Commonwealth Government's Backing Australia's Ability Strategic Information Initiative, Alphawest, and Sun Microsystems.

References

1. Tou, J.T., Gonzalez, R.C.: Pattern Recognition Principles. In: Kalaba, R. (ed.) Applied Mathematics and Computation, Addison-Wesley Publishing Co., Reading, Massachussets (1974)
2. Sussner, P., Valle, M.E.: Gray-Scale Morphological Associative Memories. IEEE Transactions on Neural Networks 17, 559–570 (2006)
3. Leimer, J.J.: Design Factors in the Developmentof an Optical Character Recognition Machine. IEEE Transactions on Information Theory, 167–171 (1962)
4. Casasent, D.: Coherent Optical Pattern Recognition. In: Proceedings of the IEEE (1979)
5. Smagt, P.V.D.: A comparative study of neural network algorithms applied to optical character recognition. In: International conference on Industrial and engineering applications of artificial intelligence and expert systems, ACM Press, Charleston, South Carolina, United States (1990)
6. Khan, A.I.: A Peer-to-Peer Associative Memory Network for Intelligent Information Systems. In: The Proceedings of The Thirteenth Australasian Conference on Information Systems, Melbourne, Australia (2002)
7. Khan, A.I., Mihailescu, P.: Parallel Pattern Recognition Computations within a Wireless Sensor Network. In: ICPR 2004. Proceedings of the 17th International Conference on Pattern Recognition, IEEE Computer Society, Cambridge, United Kingdom (2004)
8. Khan, A.I., Amin, A.H.M.: An On-line Scheme for Threat Detection Within Mobile Ad Hoc Networks. In: Yang, L.T., et al. (eds.) Mobile Intelligence: Mobile Computing and Computational Intelligence, John Wiley & Sons, England (in print, 2007)
9. Baqer, M., Khan, A.I., Baig, Z.A.: Implementing a graph neuron array for pattern recognition within unstructured wireless sensor networks. In: Proceedings of EUC Workshops (2005)

Efficiently Finding Negative Association Rules Without Support Threshold

Yun Sing Koh and Russel Pears

School of Computing and Mathematical Sciences,
Auckland University of Technology, New Zealand
{ykoh, rpears}@aut.ac.nz

Abstract. Typically association rule mining only considers positive frequent itemsets in rule generation, where rules involving only the presence of items are generated. In this paper we consider the complementary problem of negative association rule mining, which generates rules describing the absence of itemsets from transactions. We describe a new approach called MINR (Mining Interesting Negative Rules) to efficiently find all interesting negative association rules. Here we only consider the presence or absence of itemsets that are strongly associated. Our approach does not require a user defined support threshold, and is based on pruning items that occur together by coincidence. For every individual itemset we calculate two custom thresholds based on their support: the positive and negative chance thresholds. We compared our implementation against Pearson ϕ correlation.

1 Introduction

Association rule analysis is the task of discovering association rules that occur frequently in given transaction data set [1]. Initially this was proposed for market basket analysis. Given a set of transactions D, association rule mining finds the complete set of association rules whose *support* is greater than a user-defined minimum support threshold (*minsup*) and *confidence* greater than a user-defined minimum confidence threshold (*minconf*). The following is a formal statement of association rule mining for transaction databases. Let $I = \{i_1, i_2, \ldots, i_m\}$ be the universe of items. A set $X \subseteq I$ of items is called an itemset. A transaction $t = (tid, X)$ is a tuple where tid is a unique transaction ID and X is an itemset. A transaction database D is a set of transactions. The *count* of an itemset X in D, denoted by count(X), is the number of transactions in D containing X. The *support* of an itemset X in D, denoted by supp(X), is the proportion of transactions in D that contain X. The rule $X \rightarrow Y$ holds in the transaction set D with *confidence* c where $c = $ conf$(X \rightarrow Y)$ and conf$(X \rightarrow Y) = $ supp$(XY)/$supp(X).

The dominant theme in traditional association mining is the discovery of positive association rules in frequently occurring itemsets. However in this paper we consider the complementary problem called negative association rule mining. A negative association rule describes a relationship whereby the occurrence of some itemset implies the absence of some other itemset, i.e., what items that are not purchased together in a market basket scenario. An example of such a rule is 80% of customers who buy "butter" do not buy "margarine". A negative association rule is an implication of the form

$X \rightarrow \neg Y$, $\neg X \rightarrow Y$, or $\neg X \rightarrow \neg Y$, where $X \subseteq I$ and $Y \subseteq I$. $\neg X$ and $\neg Y$ are negations, and denote the absence of itemsets. A strong negative association between itemsets whose distribution fulfils the minimum confidence requirements implies a negative association rule between the two itemsets. Finding negative association rules is not a straight forward process. In a transaction database there may be tens of thousands of items. It is possible that many of the combinations of items may not appear in any of the transactions. Finding all combinations that lead to rules will be prohibitively expensive. If the absence of certain items is taken into consideration it will produce millions of rules, many of which are uninteresting. Section 2 presents related work. In Section 3 we introduce our approach to negative association rule mining. Experimental results are described in Section 4. We summarise our research in Section 5.

2 Related Work

Negative association was first noted by Brin et al. [2]. Since then there have been several attempts to solve the problem of negative association rule mining. Savasere et al. [3] proposed a method to mine strong negative association rules. They discovered positive association rules and combined them with domain knowledge to constrain the search space to mine interesting negative association rules. The main drawback of this approach is that it requires a predefined taxonomy. Consequently, their approach is hard to generalise as it is domain dependent. Yuan et al. [4] proposed a method of mining negative association rules based on locality of similarity. Items that belong to the same locality of similarity tend to participate in similar association rules. However this approach has the same drawback to the approach proposed by Savasere et al. [3], in that it requires a predefined taxonomy. Wu et al. [5] proposed a new method of mining for both positive and negative association rules. They defined a heuristic interest function. Itemsets with an interest value greater than or equal to a user defined threshold are considered potentially interesting. While this method yields certain types of negative association rules, the main drawback was that it is incapable of finding rules that are interesting but do not meet the support threshold. Antonie and Zaïane [6] introduced an automatic progressive threshold process. They used the Pearson's ϕ correlation coefficient to measure the strength of an association. If there are no strongly correlated rules found in any round, the threshold is lowered progressively until some rules are found with moderate correlation. However this method still requires the user to define a minimum support threshold.

3 Mining Interesting Negative Rules (MINR)

The basic philosophy behind our proposed method is to use Fisher's exact test in identifying itemsets that occur together with a statistically significant probability. Any itemset which has a support greater than the generated positive chance threshold is considered to be a candidate for positive rule generation, whereas any itemset which has a support less than the generated negative chance threshold is considered to be a candidate for negative rule generation. The calculations of these thresholds were adapted from the minabssup method introduced by Koh et al. [7]. We examine the probability of seeing two itemsets occurring together. For N transactions in which the antecedent A occurs in a transactions and consequent B occurs in b transactions, we can calculate the probability that A and B will occur together exactly c times by chance. The probability is calculated as Pcc in (1) which is derived from Fisher's exact test. The probability that A and B will occur together exactly c times is:

$$\text{Pcc}\left(c\middle|N,a,b\right) = \frac{\binom{a}{c}\binom{N-a}{b-c}}{\binom{N}{b}} \tag{1}$$

The chance threshold is calculated independently for each candidate itemset. Given Pcc in (1), we want the least number of collisions above which Pcc is larger than some large p-value (usually 0.9999) for a positive association and small p-value (usually 0.0001) for a negative association. This is:

$$\text{chance}\,(N,a,b,p) = \min\left\{ s \middle| \sum_{i=0}^{i=s} \text{Pcc}\left(i\middle|N,a,b\right) \geq p \right\} \tag{2}$$

We use a chance threshold s, that allows us to identify itemsets that do not appear together by chance. For instance, Pcc($c = 100\big|N = 10000, |A| = 1000, |B| = 1000) = 0.044$. If the probability Pcc(count(AB) $= s\big|N, |A|, |B|$)≤ 0.0001 and $\sum_{i=0}^{i=s}$ Pcc($s\big|N, |A|, |B|) \leq 0.0001$, and say that if we see count(AB)$\geq s$ then AB is appearing less than chance would allow. Hence, the chance threshold is set to s. In this particular example, s is 68. Itemsets that appear together with a frequency lower than the chance threshold are considered to be candidates for generating negative association rules.

```
**MINR Algorithm**
Input: Transaction database $D$, universe of items $I$
Output: Negative itemsets
$Idx \leftarrow$ invert$(D, I)$
$k \leftarrow 1$
$L_k \leftarrow \{\{i\}|i \in$ dom $Idx$, support$(\{i\}, Idx) > 0\}$
$N_k \leftarrow \emptyset$
while ($L_k \neq \emptyset$ and $N_k \neq \emptyset$)
    $k \leftarrow k + 1$
    $C_k \leftarrow \{x \cup y | x, y \in L_{k-1}, |x \cap y| = k - 2\}$
    $L_k \leftarrow \{c | c \in C_k$, count$(c) >$ chance$(N, a, b, 0.9999)$
    $N_k \leftarrow \{n | n \in C_k \backslash L_k$, count$(n) <$ chance$(N, a, b, 0.0001)\}$
end while
return $\bigcup_{t=2}^{k-1} N_t$
```

Here we will take a look at how the chance threshold is used in MINR. On the first pass through the database, an inverted index is built using the unique items as keys and the transaction IDs as data. Initially, the set of frequent itemsets is empty. After generating all frequent 1-itemsets, we iterate to find all k-itemsets, with $k > 1$. Only candidate itemsets that are above the positive chance threshold for that itemset will be extended. Whereas candidate itemsets that are below the negative chance threshold for that itemset are considered during negative rule generation. The use of the chance threshold allows us to construct nodes through an objective measure of support whose value is tailored to each individual itemset.

The rule generation process mirrors that of traditional association rule generation except that rules are divided into three types. Type 1 represents rules which are of the form of $X \rightarrow \neg Y$, whereas Type 2 represents rules of the form $\neg X \rightarrow Y$. Type 3 represents rules which are of the form $\neg X \rightarrow \neg Y$. Type 1 and 2 rules are generated directly from the negative itemsets. If a pair of itemsets (X, Y) satisfies the positive chance and minimum confidence thresholds a frequent rule is generated. Otherwise, if an itemset fulfils the positive chance threshold, we generate the Type 3 rule, $\neg X \rightarrow \neg Y$, if its confidence is above the minimum confidence. The chance threshold is more strict on allowing Type 3 rules to be generated, which is required as there may be too many rules generated, many of which are not of any interest.

4 Results and Discussion

In this section, we compare the performances of the MINR algorithm based on pruning with the chance threshold against a rule generator based on Pearson's ϕ correlation [8]. Seven different datasets from the UCI Machine Learning Repository [9] where chosen. Table 1 displays the results of running MINR and Pearson's ϕ correlation. We only report on the negative association rules generated by both methods.

Table 1. Results of negative rule mining

Dataset	Chance Threshold (minconf = 0.90)						ϕ Correlation (minconf = 0.90)											
							Correlation Threshold = ±0.3						Correlation Threshold = ±0.5					
	Rules			Passes	Avg Neg Sets	Time (s)	Rules			Passes	Avg Neg Sets	Time (s)	Rules			Passes	Avg Neg Sets	Time (s)
	Type 1	Type 2	Type 3				Type 1	Type 2	Type 3				Type 1	Type 2	Type 3			
TeachingEval.	0	1	4	3	1	1	2	1	54	3	1	1	0	1	15	3	1	1
Bridges	33	0	10	6	3	8	51	0	229	5	1	8	11	0	113	5	1	6
Zoo	110	69	11	8	10	3	160	84	220	5	10	3	63	50	120	5	10	3
Hepatitis	23	13	7	5	4	23	35	14	710	6	1	23	4	4	241	6	1	20
Flag	90	34	85	6	13	62	81	32	663	6	1	60	11	7	169	6	1	52
Soybean-Large	545	73	198	16	22	141	282	51	746	7	6	6	69	22	530	7	6	4
Mushroom	2556	132	597	11	130	185	210	53	353	6	8	28	64	26	135	6	8	22

In terms of generating positive rules we used Cohen's guideline [8] to decide whether a pair of itemsets (X,Y), are strongly or moderately correlated with each other. With respect to negative association rules we used the inverse ranges. From the results in Table 1 it is evident that pruning with chance threshold on average results in less rules than Pearson's correlation at the 0.3 threshold. When we increased the Pearson correlation threshold to 0.5 there were fewer Type 1 and 2 rules found by Pearson's than chance threshold. Overall we notice that the chance threshold generated less Type 3 rules than Pearson's correlation. This is due to chance threshold being more strict.

One problem that Pearson's ϕ correlation faces is that the greater the difference between the support of the two itemsets analysed, the smaller the maximum possible correlation value becomes. We note that the significance of ϕ correlation is the same as that of chi-square. ϕ correlation tends to understate asymmetric relationships and is very sensitive to shifts in marginal distributions. To analyse the quality of the rules found we looked at the lift values generated by the negative itemsets. In these cases we considered the negative itemsets in the rules.We found that many of the rules found by ϕ correlation that were not found by chance threshold had a lift value only slightly above 1.0. For example, using Pearson with a correlation threshold of ±0.5, we found

8 out of 15 rules generated had a lift value less than 1.04. The use of Pearson's ϕ correlation involves the setting of an arbitrary threshold. Setting this particular threshold is a problem, if we set the threshold too low, it no longer filters uninteresting rules but if we set it too high then we may end up pruning some interesting rules. The dilemma faced by miner is to avoid overpruning while ensuring that interesting rules are generated. Resolving this dilemma requires the miner to experiment with different values of the Pearson's ϕ coefficient and to perform a comparative analysis of the results to identify an optimum value for the correlation threshold which is inherently time consuming. The latter value is also likely to be domain dependent, thus further compounding the difficulty of mining.

5 Conclusion

Negative association rules are harder to find as extra measures are required to ensure that only meaningful rules are generated. In this paper we introduce the MINR algorithm that generates negative association rules without the need to set a user defined threshold. We compared the results produced by our implementation against an implementation based on Pearson's ϕ correlation. On average, our method produces a more concise set of rules than Pearson's ϕ correlation. We noted that ϕ correlation has problems dealing with datasets where there is a wide variation in support amongst the itemsets, and the MINR algorithm does not suffer from this.

References

1. Agrawal, R., Imielinski, T., Swami, A.: Mining association rules between sets of items in large databases. In: Buneman, P., Jajodia, S. (eds.) Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data, pp. 207–216 (1993)
2. Brin, S., Motwani, R., Silverstein, C.: Beyond market baskets: Generalizing association rules to correlations. In: Peckham, J. (ed.) SIGMOD 1997. Proceedings ACM SIGMOD International Conference on Management of Data, May 13-15, ACM Press, Tucson, Arizona, USA (1997)
3. Savasere, A., Omiecinski, E., Navathe, S.B.: Mining for strong negative associations in a large database of customer transactions. In: ICDE, pp. 494–502 (1998)
4. Yuan, X., Buckles, B.P., Yuan, Z., Zhang, J.: Mining negative association rules. In: ISCC 2002. Proceedings of the Seventh International Symposium on Computers and Communications (ISCC 2002), p. 623. IEEE Computer Society, Washington, DC, USA (2002)
5. Wu, X., Zhang, C., Zhang, S.: Efficient mining of both positive and negative association rules. ACM Trans. Inf. Syst. 22, 381–405 (2004)
6. Antonie, M.L., Zaïane, O.R.: Mining positive and negative association rules: an approach for confined rules. In: Boulicaut, J.-F., Esposito, F., Giannotti, F., Pedreschi, D. (eds.) PKDD 2004. LNCS (LNAI), vol. 3202, pp. 27–38. Springer, Heidelberg (2004)
7. Koh, Y.S., Rountree, N., O'Keefe, R.: Finding non-coincidental sporadic rules using apriori-inverse. International Journal of Data Warehousing and Mining 2, 38–54 (2006)
8. Cohen, J.: Statistical power analysis for the behavioral sciences, 2nd edn. Lawrence Erlbaum Associates, Hillsdale, NJ (1988)
9. Newman, D., Hettich, S., Blake, C., Merz, C.: UCI repository of machine learning databases (1998), http://www.ics.uci.edu/~mlearn/MLRepository.html

A Data Mining Algorithm for Designing the Conventional Cellular Manufacturing Systems

ChenGuang Liu

Mathematical Institute, Tohoku University, Sendai 980-8578, Japan
liu@mail.tains.tohoku.ac.jp

Abstract. Cellular manufacturing is one of the most powerful management innovations in actualizing mass customization. In this paper, we develop a data mining algorithm for designing the conventional cellular manufacturing systems.

Keywords: Data Mining, Association Rule Mining, Cellular Manufacturing Systems.

1 Introduction

In today's environment, neither mass production nor job-shop production can successfully react to the customers in producing the right product at right price and at right time. How does transform the present manufacturing paradigm to mass customization is a challenge for every company. Converting the exiting flow lines or job shops to manufacturing cells has been seen as one of the most successful ways to cope with such a challenge.

Generally speaking, there are two ways in designing the cellular manufacturing systems [3]. The first way is to group the job shops into manufacturing cells based on group technology (GT). This conversion way have been extensively investigated in the literature over the last three decades. The designed manufacturing systems with respect to this way are called conventional cellular manufacturing systems in the literature. The second way is to dismantle flow lines into manufacturing cells according to a combination of Toyota's production theory and Sony's single-worker theory. Such a conversion way has been actively introduced by Japanese manufacturing industries since the 1990s, however, the substantial attention in the academic literature happened in recent years. The manufacturing systems that are converted by following such a way are known as Japanese cell production systems. In the present work, we are interested in the designing of the conventional cellular manufacturing systems.

Data mining has enjoyed popularity in recent years, with advances in both academia and industry. Indeed, most scientific disciplines are becoming data-intensive and turning to data mining as a tool. Association rule mining is one of the major forms of data mining and is perhaps the most common form of knowledge discovery in unsupervised learning systems. It was originally proposed by Agrawal, Imielinski, and Swami [1], and its basic idea is to discover important and interesting associations among the data items. Uncertain association rule mining, an extension of the original association rule mining, was developed by Shyu *et al.* [4]. The significant feature of this method is that it can capture the uncertain item relationships in the datasets.

The application of data mining to cellular manufacturing is still under utilized and infrequently used. To the best of our knowledge, the first academic literature [2] on this area appeared in 2003. In that work, Chen developed a cell formation approach based on association rule mining (Apriori algorithm). However, such a method ignored many real-life production factors, accordingly it can not be used to deal with cell formation problems that hold uncertain properties. In this paper, we develop a new cell formation algorithm based on *uncertain association rule mining* . The proposed algorithm takes the following key real-life production factors into consideration: operation sequence, production volume, batch size, alternative process routings, cell size, the number of cells, and path coefficient of material flows.

2 Notations

Given an operation sequence of part p in process routing r, we can represent such operation sequence by an ordered dataset $OD^{(p,r)}$. For example, we use a $OD^{(p,r)} = (m_1, m_5, m_3, m_4, m_1, m_5)$ for the operation sequence of part p in process routing r: $m_1 \rightarrow m_5 \rightarrow m_3 \rightarrow m_4 \rightarrow m_1 \rightarrow m_5$.

By the Γ and L, we denote the Γ-th process and the length of the operation sequence, respectively. By $block^l_\Gamma$, we denote the subsidiary operation sequence from Γ-th process with length l. We define the ordered closure $OC^{(p,r)}$ for $OD^{(p,r)}$ as $OC^{(p,r)} = \{block^l_\Gamma | 1 \leq \Gamma \leq L, 1 \leq l \leq L - \Gamma + 1\}$.

For a given operation sequence, by m_{ai}, we denote the machine that acts the ai-th process. By $OCN^{(p,r)}_{(m_{a1}, m_{a2}, \ldots, m_{ah})}$, we denote the number of ordered dataset $(m_{a1}, m_{a2}, \ldots, m_{ah})$ in $OC^{(p,r)}$. In order to incorporate the real-life production factor of unusual path coefficient of material flows between two machines, we define a modified value OCZ from OCN as follows:

• if the number of elements of ordered dataset is 2, the OCZ for (m_{a1}, m_{a2}) is given by $OCZ^{(p,r)}_{(m_{a1}, m_{a2})} = OCN^{(p,r)}_{(m_{a1}, m_{a2})} \times \alpha_{(m_{a1}, m_{a2})}$, where $\alpha_{(m_{a1}, m_{a2})}$ is the path coefficient of material flow $m_{a1} \rightarrow m_{a2}$;

• otherwise, the OCZ is given as $OCZ^{(p,r)} = OCN^{(p,r)}$.

Let $\{m_{a1}, m_{a2}, \ldots, m_{ah}\}$ be an itemset with h items. By $OCN^{(p,r)}_{\{m_{a1}, m_{a2}, \ldots, m_{ah}\}}$ we denote the number of order datasets in $OC^{(p,r)}$ that include all items of $\{m_{a1}, m_{a2}, \ldots, m_{ah}\}$ and only include the item of $\{m_{a1}, m_{a2}, \ldots, m_{ah}\}$. Corresponding to the modification of $OCN^{(p,r)}_{(m_{a1}, m_{a2})}$ to $OCZ^{(p,r)}_{(m_{a1}, m_{a2})}$ for ordered dataset (m_{a1}, m_{a2}), we can easily modify $OCN^{(p,r)}_{\{m_{a1}, \ldots, m_{ah}\}}$ to $OCZ^{(p,r)}_{\{m_{a1}, \ldots, m_{ah}\}}$ for itemset $\{m_{a1}, \ldots, m_{ah}\}$.

Now, we give some other notations that will be used in the next sections:

M the number of machines to be grouped into cells
m_i the i-th machine ($i = 1, \ldots, M$)

P	the number of part types
p	the p-th part($p = 1, ..., P$)
C	the number of cells to be formed
c	the c-th cell($c = 1, ..., C$)
R_p	the number of alternative process routings of part p
r	the r-th process routing ($r = 1, ..., R_p$)
Q_p	the production volume for part p
q_p	the batch size for part p
S	the cell size
$u_{k,h}$	the h-th itemset of with k elements
u_k^l	the $l-th$ selected itemset with k elements
$sup_{u_k^l}$	the support measure value of itemset u_k^l
λ	unit cost of intracell transportation;
β	unit cost of intercell transportation;

$$\zeta_{ij}^c = \begin{cases} 1 \text{ if both machine } m_i \text{ and } m_j \text{ are assigned to the same cell } c \\ 0 \text{ otherwise} \end{cases}$$

$$\xi_{ij}^c = \begin{cases} 1 \text{ if either machine } m_i \text{ or machine } m_j \text{ is assigned to the cell } c \\ 0 \text{ otherwise} \end{cases}$$

3 Algorithm Approach

Let a process routing of a part be a record identifier(transaction), and a machine be an item. An uncertain association rule mining algorithm for cell formation has the following inputs and outputs.

Inputs: Transaction dataset $\mathcal{D}$, and inter-relationship dataset $\mathcal{R}$. Transaction dataset $\mathcal{D}$ incorporates production volume, batch size. Inter-relationship dataset $\mathcal{R}$ incorporates alternative process routing, path coefficient of material flows, flow times, and operation sequences of each part.

Outputs: Machine cells, part families, and the chosen process routings.

The stages of the proposed algorithm are as follows:

Stage 1: Construct datasets $\mathcal{D}$ and $\mathcal{R}$ incorporating those production factors considered in this study. The transaction dataset $\mathcal{D}$ can be constructed as Table 1:

Table 1. Transaction Dataset $\mathcal{D}$

TID	Part	Routing	m_1	m_2	m_3	...	m_M
1	1	1	$d_1^{1,1}$	$d_2^{1,1}$	$d_3^{1,1}$	$d_{...}^{1,1}$	$d_M^{1,1}$
2	1	2	$d_1^{1,2}$	$d_2^{1,2}$	$d_3^{1,2}$	$d_{...}^{1,2}$	$d_M^{1,2}$
3	2	1	$d_1^{2,1}$	$d_2^{2,1}$	$d_3^{2,1}$	$d_{...}^{2,1}$	$d_M^{2,1}$
4	3	1	$d_1^{3,1}$	$d_2^{3,1}$	$d_3^{3,1}$	$d_{...}^{3,1}$	$d_M^{3,1}$
...	...	...					
t	p	r	$d_1^{(p,r)}$	$d_2^{(p,r)}$	$d_3^{(p,r)}$	$d_{...}^{(p,r)}$	$d_M^{(p,r)}$
...	...	...					

In Table 1, $d_i^{(p,r)}$ is the transportation times passing machine m_i for part p in process routing r. If part p is operated on m_i in process routing r, let $d_i^{(p,r)} = \lceil \frac{Q_P}{q_p} \rceil$. Otherwise, we set $d_i^{(p,r)}$ with 0.

Since the upper bound of the cell size S is predefined, the all itemsets with elements more than S can be omitted from the standard form. Hence, the inter-relationship dataset $\mathcal{R}$ in our algorithm can be constructed as Table 2:

Table 2. The modified inter-relationship Dataset $\mathcal{R}$

Itemsets	r_1	r_2	... r_R
$\{m_1\}$	$R^1_{\{m_1\}}$	$R^2_{\{m_1\}}$	... $R^R_{\{m_1\}}$
$\{m_1, m_2\}$	$R^1_{\{m_1,m_2\}}$	$R^2_{\{m_1,m_2\}}$	... $R^R_{\{m_1,m_2\}}$
$\cdots$			
ω	R^1_ω	R^2_ω	... R^R_ω
$\cdots$			
$\{m_{n-k+1}, ..., m_n\}$	$R^1_{\{m_{n-k+1},...,m_n\}}$	$R^2_{\{m_{n-k+1},...,m_n\}}$	... $R^R_{\{m_{n-k+1},...,m_n\}}$

In the table, R^r_ω is a weight of OCZ for ω over that for all itemsets with respect to process routing r, that is $R^r_\omega = \sum\limits_{p=1}^{P} OCZ^{(p,r)}_\omega \Bigg/ \sum\limits_{p=1}^{P} \sum\limits_{k=2}^{S} \sum\limits_{h} OCZ^{(p,r)}_{u_{k,h}}$.

Stage 2: Generate the *bpa* value. Maintaining datasets $\mathcal{D}$ and $\mathcal{R}$, generate the basic probability assignment *bpa* by $bpa^{(p,r)}_\omega = R^r_\omega \times \min\{d_i^{(p,r)} | m_i \in \omega\}$. In such a *bpa*, both the item multiplicity and item inter-relationships are embedded.

Stage 3: Apply the uncertain association rule mining approach to obtain the association rules among machines. Based on the *bpa* values, we apply the uncertain association rule mining approach proposed in [4], to obtain association rules among machines. The standard approach of uncertain association rule mining includes the following five steps:

step 1. Normalize the basic probability assignment *bpa*-values by *the normalized* $bpa^{(p,r)}_\omega = bpa^{(p,r)}_\omega \sum_k / \sum_h bpa^{(p,r)}_{u_{k,h}}$.

step 2. Compute the uncertain interval $[Bel^{(p,r)}_\omega, Pl^{(p,r)}_\omega]$.

step 3. Compute the Shannon-like measure $SL^{(p,r)}_\omega$.

step 4. Measure the support value sup_ω.

step 5. Normalize the support value sup_ω using *the normalized* $sup_\omega = \frac{sup_\omega}{SUP}$, where $SUP = \max_\omega\{sup_\omega\}$.

Stage 4: Select the itemsets based on association rules among machines. Based on the association rules among machines derived in stage 2, we select the itemsets that satisfy the following $SOLVE$ model:

step 1. Initial, set $k = S$.

step 2. Select the itemsets in maximizing $\sum_l sup_{u^l_k}$ subject to $\bigcap_l u^l_k = \phi$ and $(\bigcup_l u^l_k) \bigcap (\bigcup_{d=k+1}^{S} (\bigcup_l u^l_d)) = \phi$.

step 3. Check the value of $|\bigcup_{d=S}^{k}(\bigcup_l u_d^l)|$. If $|\bigcup_{d=S}^{k}(\bigcup_l u_d^l)| < M$, set $k = k-1$, go step 2.If $|\bigcup_{d=S}^{k}(\bigcup_l u_d^l)| = M$, return the all selected itemsets, end.

In this model, the objective function serves to select the itemsets with k elements such that sum of support values of selected itemsets are maximized. The constraints ensure that each machine is assigned to only one cell.

Stage 5: Configure the cell formation according to the selected itemsets. In this stage, there are three works to configure the cell formation with respect to each selected plans obtained in Stage 4.

1). Group the items into machine groups. This work is comprised from the following processes:

step 1. Construct a itemset table ordered by the number of items.

step 2. Denote the itemset that has the minimum number of items by S_{min}, and the itemset that has the maximum number of items by S_{max}.

case I. If the sum is larger than the cell size, then form a machine group by collecting the all items of S_{max};

case II. If the sum is less than the cell size, add the items S_{min} into S_{max}, then delete the original S_{min} from the ordered table. After that, check the number of itemsets in the revised table.

subcase II-1. If the number of itemsets in the revised table is larger than 1, select a itemset that has the minimum number of items in the revised table and denote it by S_{min}, then go to step 3;

subcase II-2. If the number of itemsets in the revised table is equal to 1, then form a machine group by collecting the all items of S_{max}, end.

2). Allocate the parts into part families: Based on the formed machine groups, dispatch parts to machine cells by the dispatching rule: assign each part to the machine group that processes the first operation of the part.

3). Chose the aim process routing: Once the machine groups and part families are determined, for any part p, the material flow cost in process routing r can be computed by $COST^{(p,r)} = \left[\sum_{c=1}^{C}\sum_{i=1}^{M}\sum_{j=1}^{M}\zeta_{ij}^{c}\lambda + \sum_{c=1}^{C}\sum_{i=1}^{M}\sum_{j=1}^{M}\xi_{ij}^{c}\beta\right] \times OCZ_{(i,j)}^{(p,r)} d_i^{(p,r)}$, where $OCZ_{(i,j)}^{(p,r)}$ is a short representation for $OCZ_{(m_i,m_j)}^{(p,r)}$ defined in Section 2. For any part p, we define a special process routing τ^p as $\tau^p = \min_r\{COST^{(p,r)}\}$. For each part p, we chose τ^p as the aim process routing.

In the case where there is a unique selected plan in Stage 4, the final result directly follows these three works. In the case where there are more than one selected plans in Stage 4, we compare the results of all selected plans with respect to the over-all costs of material flows, then select one plan that has the minimum over-all costs of material flows as the finial result.

References

1. Agrawal, R., Imielinski, T., Swami, A.: Mining Association Rules between Sets of Items in Massive Aatabase. In: Proceedings of the ACM/SIGMOD International Conference on Management of Data, Washington, D.C. (May 26-28, 1993)
2. Chen, M.: Configuration of Cellular Manufacturing Systems Using Association Rule Induction. International Journal of Production Research 41(2), 381–395 (2003)
3. Liu, C., Yin, Y.: Analysis and Modeling of Japanese Cell Production Systems and Conventional Cellular Manufacturing Systems. In: Proceedings of Special Symposium of Operations Research, Ito Hotel Juraku, Ito, Japan (August 29-31, 2007)
4. Shyu, M., Haruechaiyasak, C., Chen, S., Premaratne, K.: Mining Association Rules with Uncertain Item Relationships. Computers and Industrial Engineering 34(1), 3–20 (1998)

A Compromised Large-Scale Neighborhood Search Heuristic for Cargo Loading Planning

Yanzhi Li[1], Yi Tao[1,2], and Fan Wang[2]

[1] Department of Management Sciences, City University of Hong Kong, Hong Kong
yanzhili@cityu.edu.hk, yitao2@cityu.edu.hk
[2] Department of Management Sciences, Sun Yat-Sen University, P.R. China
fanwang@gmail.com

Abstract. In this work, we propose a compromised large-scale neighborhood, which is embedded in simulated annealing to solve a cargo loading planning problem arising in logistics industry. It is "compromised" because it makes a tradeoff between the extensive backward checking work incurred in traditional subset-disjoint restriction and the possible infeasibility resulting from the relaxing the restriction. Extensive experiments have shown the competitive advantages of the heuristic approach. The proposed neighborhood search method is generally applicable.

Keywords: search, heuristic, cargo loading.

1 The Cargo Loading Planning Problem

Our study is motivated by the air cargo business of one of the largest forwarders in the world. Because the company does not own its own flights, it has to rent containers/capacity from airlines to transport its cargos. To guarantee service quality in the increasingly competitive business, every morning the company reserves capacity for today's demand based on forecasting. When the cargos are ready for transport at the end of the day, they make a loading plan and transport the cargos as much as possible at the minimum cost.

To be specific, the company has received N cargos to be forwarded during the day, each with a weight w_i, $i = 1,..,N$. Container are classified into K types. Type k is described with the tuple $\{c_k, U_k, c_k^E, U_k^E, v_k\}$, specifying unit pivot cost, pivot capacity, unit extra cost (equal to unit special cost minus unit pivot cost), extra capacity (equal to maximum capacity minus pivot capacity), and the number of reserved such containers correspondingly. In other words, the cost of a container is piecewise linear function of its weight, starting with a slope c_k and then after the turning point U_k, with a slope $(c_k + c_k^E)$. J containers have been reserved in the morning, where $J = \sum_{k=1}^{K} v_k$. Note that the containers may belong to different airlines. The objective is to find the loading plan with the minimum cost.

The problem can be proved strongly NP-hard by showing the well-known 3-partition problem is its special case. The strongly NP-hardness implies that it is difficult to find an efficient optimal algorithm. Because the cargo loading planning decisions need to be made in a short time, we have to resort to heuristic approaches.

2 Compromised Large-Scale Neighborhood Search

Our algorithm starts with an initial solution. An improvement graph is constructed based on the solution. A neighborhood search procedure is then applied to find a possible improvement, according to which, the improvement graph is updated. The procedure is iterated until a stopping criterion is satisfied.

2.1 Generating Initial Solutions

Initial solutions may have a significant impact on the performance of improvement heuristics. Hence several methods have been tested and we found that the following one shows stably better performance. First, all the cargos are sorted in decreasing order of weight. We then adopt a two-stage first-fit policy. In the first stage, the cargos are taken one by one from the list and put into the first container whose pivot capacity will not be exceeded after the new cargo is loaded. If there are still cargos not loaded after all the pivot capacity has been used up, in the second stage, we take the remaining cargos one by one from the list and put them into the first container whose maximum capacity will not be exceeded after the new cargo is loaded.

2.2 Compromised Large-Scale Neighborhood Search

The neighborhood $N(S)$ of given solution S is defined as the set of all feasible solutions S' which are reachable from S by a single neighborhood move. We first construct an improvement graph, which facilitates an efficient search procedure for a possible improvement.

Constructing Improvement Graph. Given a solution S, the *improvement graph* is a directed graph $G(S)=(V,E)$. The vertex set V contains N nodes, each representing a cargo, and J pseudo nodes, each for a container. Let $\phi(i)$ denote the container of cargo i and π_j denote the type of container j, $j = 1, .., J$. There is a directed arc (i_1, i_2) between node (i.e., cargo) i_1 and i_2 if only (1) $\phi(i_1) \neq \phi(i_2)$ and (2) the concurrent moves – cargo i_2 leaving $\phi(i_2)$ and i_1 entering $\phi(i_2)$ – does not violate the maximum capacity constraint of container $\phi(i_2)$, i.e., $W_{\phi(i_2)} + w_{i_1} - w_{i_2} \leq U_{\pi_{i_2}} + U^E_{\pi_{i_2}}$, where $W_{\phi(i_2)}$ denotes the original weight in container π_{i_2}. An arc involving one (at most one) pseudo node means either no cargo leaving the container (if i_1 is the pseudo node) or no cargo entering the container (if i_2 is the pseudo node).

Let piecewise linear function $f_k(W_j)$ denote the cost of a container j of type k, where W_j is the total weight of cargos in the container. An arc length is also defined for each arc (i_1, i_2), which expresses the cost change of container $\phi(i_2)$ if the concurrent moves are realized:

$$A_{i_1 i_2} = f_{\pi_{\phi(i_2)}}(W_{\phi(i_2)} + w_{i_1} - w_{i_2}) - f_{\pi_{\phi(i_2)}}(W_{\phi(i_2)})$$

Note that the length $A_{i_1 i_2}$ reflects the cost change for the concurrent moves – cargo i_2 leaving $\phi(i_2)$ and i_1 entering $\phi(i_2)$; we need to consider whether container $\phi(i_2)$ exceeds its pivot capacity before and after the moves.

Neighborhood Structure. A large scale neighborhood involves exchange of multiple cargos. It can be a cyclic or path exchange. A cyclic exchange involving cargos $i_1, i_2, .., i_n$ conducts the following exchange: $\phi(i_1) := \phi(i_2),\ \phi(i_2) := \phi(i_3), .., \ \phi_(i_n) := \phi_(i_1)$(note here is the original container of i_1, before exchange), i.e., reassign a cargo to the container of its next neighborhood in a cyclic manner. A path exchange involving cargos $i_1, i_2, .., i_n$ conducts the similar changes as the cyclic exchange except that the cargo i_n does not go to $\phi(i_1)$; instead, it goes to a container j, where $j \neq \phi(i_k), \forall\ k = 1, .., n$. A path exchange is necessary in addition to a cyclic one because it has the potential to reduce the number of used containers.

A cyclic (path) exchange is called *subset-disjoint* if $\phi(i_j) \neq \phi(i_k), \forall\ 1 \leq j < k \leq n$. All the existing literature focuses on subset-disjoint exchanges. It has the advantage that there is a one-to-one correspondence between subset-disjoint exchanges and subset-disjoint cycles in the improvement graph. Hence a subject-disjoint cycle on the improvement graph with negative cost yields an improved solution. The disadvantage is that lots of effort has to be spent on extensive backward checking to enforce the subset-disjoint constraint.

A natural question arises: What if we allow $subset - nondisjoint$ exchanges? This brings some troubles. First, a subset-nondisjoint cycle in the improvement graph may correspond to an infeasible exchange because the capacity constraints are no longer be guaranteed. Second, even if the capacity constraints are satisfied, the cost change of a container involving multiple cargos may not be exact due to the piecewise linear cost function and hence the resulting negative cycle may not correspond to an improving exchange. On the other hand, allowing subset-nondisjoint exchanges frees us from tedious backward checking work for the subset-disjoint constraints. On the whole, if the bad moves due to the relaxation is controllable, it may be worthy to make such a compromise. We call neighborhood allowing subset-nondisjoint exchanges *compromised large-scale neighborhood*(CLSN). Next, we show an efficient searching procedure in CLSNs.

Searching for Improvement. We adopt a heuristic procedure to search for improvement. A node i_s is selected as source node. We look for a cycle with negative cost in the improvement graph, starting from s.

The label-correcting method is used to realize this. Each node is assigned a label. At the beginning, node s is assigned l_s=0 and other nodes are assigned a label equal to ∞. We then use node s to update the label of its adjacent nodes and continue to use updated nodes to update its adjacent nodes, etc. A detailed description of label-correcting procedure can be found in [1]. We stop the procedure once we find node s is relabeled as negative.

Note that we do not exclude subset-nondisjoint exchanges and thus the search procedure may result in many *invalid* exchanges, i.e., exchanges corresponding to a negative cycle in the improvement graph while turning out to be infeasible or non-improving. To reduce such redundancies and improve the efficiency of the label-correcting procedure, we utilize the following observation.

Observation 1: If $i_0, i_1, i_2, .., i_n, i_0$ is a cycle with negative cost, then there exists a node i_r in the cycle so that each partial path $i_r, i_{r+1}, .., i_{(r+k) \texttt{mod}\ n}$, where $k = 1, .., n$, is a negative cost path.

This observation is easy to prove and hence omitted here; a similar proof can be found in [2]. With the observation, our searching procedure can be improved significantly. It indicates that we should start with an arc with negative cost, probably the more negative, the better. During the label-correcting procedure, we only maintain nodes with negative labels and use them to update their adjacent nodes. Those nodes with positive labels are discarded because (1) if they do not belong to some negative cycle, by doing this, we avoid some unnecessary work and (2) if they do belong to some negative cycle, according to Observation 1, the negative cycle can still be detected by starting from another proper node. Computational experiments show that such a revision to the label-correcting procedure is very successful. We can detect an improvement very quickly and the total algorithm time is shortened while the performance gets better.

2.3 CLSN Search Embedded in Simulated Annealing

In the existing literature about large-scale neighborhood search, each iteration is an improvement. However, we are not guaranteed to have obtained an improving exchange even if we stop at a negative cycle because we allow subset-nondisjoint cycles. Hence we need a careful treatment for non-improving exchanges. We have decided to embed CLSN in a simulated annealing framework. Such an integration has a positive side effect – our algorithm has a chance of escaping from the trap of local optimum. The algorithm is summarized in Algorithm 1.

3 Computational Results

The algorithm was tested with a data set randomly generated according to realistic container specifications. The result is very encouraging. (1) **Effect of Compromise** We find that subset-nondisjoint exchanges account for a high percentage after we relax the subset-disjoint restriction. However, 65.93% of the detected subset-nondisjoint cycles are actually improving exchanges. It implies that the impact of the relaxation is positive rather than negative. The compromise is quite worthy. (2) **Algorithm performance** We compared our method to the best available results obtained from an integer programming model, which was solved by ILOG CPLEX 9.0. We find that our approach has obtained very closed-to-optimal(within 1%) performance in a very short time for cases of

Algorithm 1. CLSN Search Embedded Simulated Annealing Algorithm

```
Read input: $w_i, \{c_k, U_k, c_k^E, U_k^E, v_k\}$, temperature updating coefficient $\rho$
$Temperature \leftarrow T_{max}$; $NodesCounter$=0
Generate initial solution $S$ and construct improvement graph
WHILE $true$ DO
{
  $p$=uniform[0,1];
  IF $p > 0.5$
     search for cyclic improvement, randomly select a source node $s$;
  ELSE
     search for path improvement, randomly select a pseudo source node $s$;
  {
   Take the most negative arc emanating from $s$, say, $(s,t)$;
   Set the label of node $t$ as $A_{st}$; enqueue node $t$ and apply the label-correcting
   procedure(only maintaining nodes with negative labels in the queue). At the end
   of the label-correcting procedure,
   IF we get a feasible negative cycle
       IF the new solution $S'$ satisfies $e^{(g(S)-g(S'))/T}$ >uniform[0,1], where function
             $g(S)$ gives the cost of solution $S$
         the exchange is accepted; update the improvement graph;
         T:=T*$\rho$
       ELSE
         continue the label-correcting procedure
   ELSE
       IF we get an infeasible negative cycle
          continue the label-correcting procedure
       ELSE (the queue maintaining nodes with negative labels is empty)
          $NodesCounter := NodesCounter + 1$. If $NodesCounter > N/2$, END;
          Select a new source node $s$ from the untested.
  }
}
END
```

various scales. Since integer programming model can not solve the problem efficiently, in fact some of our results were compared to the best available lower bound. (3) **Comparison with VLSN**. Our approach is better than VLSN with GRASP for 19 out of 20 test cases. Although we have tried our best to improve performance of VLSN, this is still not an absolutely objective comparison, since performance of VLSN relies largely on the initial solutions, which depends further on people-related GRASP procedure. It, however, does show the general applicability of our algorithm framework, less relying on people.

Acknowledgement

The research of the first author was substantially supported by a grant from City University of Hong Kong (Project No. 7200068). The third author would like to thank the support in part by the Guangdong Natural Science Foundation, China, under Grant 06023185.

References

[1] Ahuja, R.K., Magnanti, T.L., Orlin, J.B.: Network Flows: Theory, Algorithms, and Applications. Prentice-Hall, Englewood Cliffs, N.J. (1993)
[2] Lin, S., Kernighan, B.W.: An effective heuristic algorithm for the traveling-salesman problem. Operations Research 21 (1973)

Concurrent Temporal Planning Using Timed Petri Nets - Policy Evaluation

Melissa Liew and Langford B. White

National ICT Australia
School of Electrical and Electronic Engineering
The University of Adelaide
Australia
{mliew,lwhite}@eleceng.adelaide.edu.au

Abstract. This paper is the first in a series which address the development of a temporal planner using timed Petri net (TPN) models of concurrent temporal plans. Unlike previous work which uses Petri net unfoldings to develop partially ordered plans (in the untimed case), the approach presented here is purely algebraic, and is based on a (min,max,+) discrete event dynamical system description of a TPN. This paper focuses primarily on *policy evaluation*, with subsequent work detailing how to use this approach for efficient optimal temporal planning.

1 Introduction

Planning has been defined as "the reasoning part of acting" [1] and thus automated planning is clearly a core part of both artificial intelligence and machine learning. Indeed, the Dynamic Planning, Optimisation and Learning Project (DPOLP) in the Statistical Machine Learning programme of National ICT Australia (NICTA - see nicta.com.au), has brought together international experts in AI planning, machine learning and optimisation in order to form a dedicated team examining the interactions of these key disciplines as applied to automated planning. In 2003, author White, as part of DPOLP, was the first to recognise that specific methods of reachability analysis for Petri Nets (PNs) called unfoldings, could be applied to solve concurrent (also known as non-linear) planning problems. [1] Subsequent work utilising PN unfoldings in planning problems has been published in [3] and [4]. These works have been only applied to untimed problems, although in [5], there is consideration of temporal planning via unfoldings. The key conclusion from this research is that the PN unfolding approach can yield planners which are at worst competitive with the state-of-the-art (eg temporal logic or SAT based approaches), and in many cases, superior. In particular, in problems which possess a significant level of concurrency of actions, the PN based approach can be considerably better.

The purpose of this paper is to report on the application of PNs to planning problems where there are temporal constraints introduced due to durative actions (ie actions that

[1] We subsequently found a similar suggestion in the Ph. D. thesis of K. L. McMillan [2], however no further development in this area was subsequently provided by McMillan, whose focus was on verification of asynchronous machines.

take time to complete after being initiated). Clearly, such planning problems are, in general, more realistic than untimed problems, and are widely found in many important areas of economic activity. One particular problem domain that we are interested in is planning and scheduling for operational level activities in the Australian Defence Organisation. Indeed, DPOLP has been successful in delivering a number of innovative solutions to this domain by virtue of the involvement of the DSTO as a key member of DPOLP. This domain is characterised, in part, by a wide range of temporal properties and constraints (see eg [6]). Coloured Petri Net (CPN) solutions were successfully applied to this problem, however, we believe that the level of abstraction inherent in CPNs as applied in this problem, obscured many of the computational advantages which might arise from the more efficient exploitation of concurrency offered by the lower level *timed* PNs utilised in our work.

The key difference and indeed innovation of this paper is that our approach to the synthesis and optimisation of concurrent temporal plans as modelled by timed Petri Nets (TPNs) is algebraic in nature. The approach reported in [5], generalises the untimed case by particular choice of cost function, however it is still based on the use of symbolic unfoldings. We shall argue that such an approach can be inefficient from both a computational and memory utilisation viewpoint, although it does offer some inherent advantages which our approach does not. Although we have developed a prototype concurrent temporal planner (TPNPlan), space limitations have led us to focus on one key aspect of the planning approach, namely policy evaluation. The main difference in our approach as compared to the standard policy iteration approach in (Semi-) Markov Decision Processes (SMDPs) in that we never create a Markov process by enumeration of a state space. Our approach works in the so-called "plan space" or the space of propositions and durative actions. This avoids the combinatorial explosion inherent in MDP models due to the artificial interleaving of concurrent actions [7]. Our policy evaluation approach is based on an innovative method of solution to the equations which define the dynamic evolution of the planning problem under a specified policy. More specifically, these equations can be described by a (min,max,+) hybrid dynamical system.

In this paper, we are only able to briefly describe our approach. We firstly mention the connection between temporal planning and timed Petri nets (TPNs), and state the dynamic equations which define the TPN behaviour. We then briefly indicate our approach to the solution of the dynamic equations and then how a plan is generated. Some generalisations and future work are then briefly indicated.

2 Temporal Plans and Timed Petri Nets

A temporal plan consists of a set of actions, each having a strictly positive duration, and a set of precondition and postconditions. Preconditions and postconditions are each sets of logical literals,that is the values taken by a set of propositions. There is an initial state, which is a set of literals which are initially true, and a goal state. A feasible plan is a partially ordered set of actions which take the state of the system from the initial state to the goal state. An action is enabled whenever all its preconditions are true. After the action has completed, it sets all its postconditions to be true. The plan terminates when all the goal literals are true. Often we are interested in finding an optimal plan, namely a partially ordered set of actions that results in the goal being achieved in the minimum time over all feasible plans.

We model temporal plans using timed Petri nets (TPNs). Space limitations preclude a detailed discussion of TPNs here, but the reader is referred to [8] for details. Specifically, we represent propositions by places and actions by transitions, and use a conversion programme called *petrify* to guarantee a TPN which is a consistent representation of the planning problem (see [5]). The solution approach of [5] is based on PN unfoldings, whereas our approach is purely algebraic in nature. We form vectors $y[k]$ and $w[k]$ for $k \geq 1$, which represent, respectively, the sequence of times of completion of transition firings, and the times of arrival of tokens in those places having more than one upstream transition (conflict). It is shown in [8] that these sequences obey a set of (min,max,+) dynamical equations:

$$y[k] = A \otimes y[k] \oplus B \otimes y[k-1] \oplus C \otimes w[k] \oplus D \otimes w[k-1] \tag{1}$$

$$w_i[k] = \bigwedge_{j_1+j_2+\cdots+j_{\gamma(i)}=k} \bigoplus_{n=1}^{\gamma(i)} y_{\beta(i,n)}[j_n]\ , \tag{2}$$

for $k \geq 2$, where $\otimes$ and $\oplus$ are respectively, the (max,+) matrix multiplication and addition operation, and $\bigwedge$ denotes minimisation. Places with conflict are labelled by $\alpha(.)$ and the $\gamma(i) > 1$ parent transitions of place $p_{\alpha(i)}$ are labelled by $\beta(i,.)$. The equation for the initial firing times ($k = 1$) is

$$y[1] = A \otimes y[1] \oplus C \otimes w[1] \tag{3}$$

$$w_i[1] = \bigwedge_{n=1}^{\gamma(i)} y_{\beta(i,n)}[1]\ . \tag{4}$$

The matrices in (1) can be explicitly (and uniquely, up to relabelling of transitions and places) constructed from the TPN. The solution of the general equations (1) and (2) is complicated by the presence of non-causality. Causality in the (max,+) part can be resolved by use of the *-operator [8]. However resolution of acausality involving (initially unmarked) places with conflict needs to be resolved in an iterative manner until all such places become causal. This occurs when every upstream transition has completed at least one firing. Space limitations preclude a more expansive description of this process. We would however remark that the complete (causal) dynamic evolution of the system is achievable once all such conflicts are resolved. Complete details will appear in a longer version of this paper [9].

One novel aspect of our solution approach is to dynamically reduce the size of the state space (by which we mean the number of active transitions) as events evolve. We thus remove all inactive transitions from the state vector $y[k]$. We partition the remaining transitions into the subset of those initially enabled, and the rest. Those transitions which are initially enabled always lead to evolution equations which are causal. The remaining transitions may be associated with places with conflict, and in such a case, will lead to acausality which needs to be resolved. We have a greedy algorithm for conflict resolution which is guaranteed to lead to a correct solution in a finite number of steps. At each step, we also remove transitions which we infer will never fire subsequently.

This is done with the aid of the *-operator referred to earlier. We also remove all places with conflict that receive no further tokens and the associated downstream transition. This process is iterated at each step until no further inactive transitions can be removed. Although equation (2) appears to require a memory of length k, we can keep track of the most recent arrival times of tokens in each place with conflict, and the epoch at which they arrived. This information is sufficient to forward-solve the equations.

We also point out that most TPNs resulting from choosing a policy for a planning problem lead to sparse matrices in the system equations, and thus considerable computational savings. We also note that many computational platforms support IEEE floating point standard arithmetic implemented in the hardware, so computations involving infinities can be performed most efficiently if these attributes can be exploited.

3 Generating a Plan

The conversion process in *petrify* generally results in a significantly larger number of transitions in the TPN than actions in the original plan in order to ensure logical consistency. In general, this expansion is exponential. However, the reachability analysis then remains polynomial in the number of transitions. The TPNs produced by *petrify* have a unique transition which fires if and only if the goal is reached. For planning, we are only interested in the first firing time of this transition, so we need only compute $y[1]$ to determine the plan makespan under each policy. Those policies which do not result in the goal being reached will also be identified. The optimal policy can then be selected. Indeed, our planner actually overlays the results of all policies on a common reachability graph enabling the selection of non-stationary policies which may yield shorter plans. We provide details in [9] which also includes comparisons with some other planning approaches on some benchmark problems.

The method described here in its generality also gives a completed reachability analysis of a general TPN.

4 Conclusion

In this paper, we have described the application of timed Petri nets to automated planning problems. The problems addressed are concurrent, deterministic propositional plans with durative actions. We derive one-safe, choice-free, timed Petri net models for each policy associated with the planning problem and use an algebraic solution method for the associated (min,max,+) dynamical system equations. We have argued that this approach may be beneficial, based on the success of Petri net unfolding based planners. This paper has mainly focused on the policy evaluation step in determining an optimal temporal plan. In future work, we will describe in detail, and benchmark, the planning tool "TPNPlan" which uses the approach described in this paper as its fundamental basis.

References

1. Ghallab, M., Nau, D., Traverso, P.: Automated Planning: Theory and Practice. Morgan Kaufmann, San Francisco (2004)
2. McMillan, K.L.: Symbolic Model Checking. PhD thesis, School of Computer Science, Carnegie Mellon University (1993)

3. Hickmott, S.L.: Optimal deterministic and probabilistic concurrent planning using petri net unfoldings. In: Proc. Int. Conference on Automated Planning and Scheduling - Doctoral Consortium (2006)
4. Hickmott, S.L., Rintanen, J., Thiébaux, S., White, L.B.: Planning using petri net unfoldings. In: Proc. Int. Joint Conference on Artificial Intelligence, pp. 1904–1911 (2007)
5. Hickmott, S.L.: Directed Unfolding - Reachability Analysis of Concurrent Systems and Applications in Automated Planning. PhD thesis, School of Electrical and Electronic Engineering, The University of Adelaide (2007)
6. Janczura, C., Gallasch, G., Billington, J., Zhang, L., Kristensen, L.M.: A coloured petri net based tool for course of action analysis and development. In: Workshop on Formal Methods applied to Defence Systems, Conferences in Research and Practice of Information Technology, vol. 12 (2002)
7. McMillan, K.L.: A technique of state space search based on unfolding. Formal Methods in System Design 6 (1995)
8. Baccelli, F., Cohen, G., Olsder, G.J., Quadrat, J.P.: Synchronization and Linearity - An Algebra for Discrete Event Systems. Wiley, Chichester (1992)
9. Liew, M., White, L.B.: Temporal planning using timed petri nets (in preparation, 2007)

Validation of a Reinforcement Learning Policy for Dosage Optimization of Erythropoietin[⋆]

José D. Martín-Guerrero[1], Emilio Soria-Olivas[1], Marcelino Martínez-Sober[1], Mónica Climente-Martí[2], Teresa De Diego-Santos[2], and N. Víctor Jiménez-Torres[2,3]

[1] Department of Electronic Engineering, University of Valencia, Spain
[2] Pharmacy Unit, University Hospital Dr. Peset, Valencia, Spain
[3] Department of Pharmacy and Pharmaceutical Technology, University of Valencia, Spain

Abstract. This paper deals with the validation of a Reinforcement Learning (RL) policy for dosage optimization of Erythropoietin (EPO). This policy was obtained using data from patients in a haemodialysis program during the year 2005. The goal of this policy was to maintain patients' Haemoglobin (Hb) level between 11.5 g/dl and 12.5 g/dl. An individual management was needed, as each patient usually presents a different response to the treatment. RL provides an attractive and satisfactory solution, showing that a policy based on RL would be much more successful in achieving the goal of maintaining patients within the desired target of Hb than the policy followed by the hospital so far. In this work, this policy is validated using a cohort of patients treated during 2006. Results show the robustness of the policy that is also successful with this new data set.

1 Introduction

Anemia is a nearly universal sequel in an End-Stage Renal Disease (ESRD) patient. Until the advent of Erythropoietin (EPO), ESRD patients faced severe anemia and cardiovascular complications, or the requirement for multiple transfusions [1]. It is crucial to make a good use of this drug by means of personalized dosages and number of administrations. This will guarantee an adequate pharmacotherapy as well as a reasonable cost of the treatment.

Several attempts at the automation of the EPO delivery have already been reported based on parametric identification in a Bayesian framework [2] and on a fuzzy rule-based control strategy [3]; nevertheless, none of these methodologies have achieved acceptance in a clinical environment. Only very recently, a few efforts have been undertaken to utilize Artificial Neural Network (ANN) models in individualized EPO dosing [4,5,6]. These works are based on approximators or predictors that can obtain the Hb concentration or the optimal EPO dosage for the next control in order to attain a certain Hb concentration. This is an interesting approach but it shows two major flaws. First, it does not take into account the dynamic relationships among Hb, Erythropoietic Stimulating Factors (ESFs) and the remainder of variables which define the state of the

[⋆] This work has been partially supported by the research projects ARVIV/2007/094 and TIN2007-61006.

patient. Second, this kind of predictors optimize the output variable a pre-defined number of steps ahead, but it is extremely difficult to obtain long-term predictors because the models are restrictive in terms of both the number of considered delays and steps ahead.

What makes Reinforcement Learning (RL) a suitable technique in this problem is its way of processing. This is because RL finds a suitable policy, i.e., given a patient in a certain state, RL provides the sequence of actions that will lead the patient to the best possible state. The goodness of the state is appropriately defined by means of the rewards assigned to the different possible values of Hb levels.

RL is a methodology based on the theory of Markov Decision Processes (MDP). Given certain states that define the input spaces, actions that can be taken, and reinforcements/rewards associated to the different possible states, RL algorithms provide an optimal policy, i.e., that sequence of actions that maximizes the long-term reward. There are two basic RL approaches: on-policy and off-policy [7].

An on-policy RL method modifies the starting policy towards the optimal one. In our particular application, patients would be probed by possibly non-optimal policies during an episodic learning process. As a result, on-policy episodic RL tools can discover a useful dosing policy, as a product of a learning process, which may be however unacceptably long and dangerous in real-time pharmacotherapy.

In an off-policy approach, the optimal policy can be computed while the agent is interacting with the environment by means of another arbitrary policy. In this work, we use the most widely known off-policy RL method (Q-learning) [8]. Our goal is to stabilize the Hb level within the target range of $11.5 - 12.5$ g/dl. The Q-learning mechanism avoids probing the system by suboptimal dosing policies during long training episodes [8] for evaluation of the state/action pairs. The proposed learning system determines the optimal drug dose using reinforcements, which are produced immediately after state transitions occurring within the patient dynamics during treatment. To the authors' knowledge, there is only one attempt of applying RL to EPO dosage individualization [9]. This is an interesting approach in which patients are simulated and an optimal policy is then obtained. In a previous work, we used real patients (data from the year 2005) to obtain a policy for EPO dosage individualization in a real-life application [10]. In the present work, we validate the obtained policy with patients from the same hospital, but using data from the year 2006 to show the robustness of the policy to provide good results not only for the data used to obtain the policy.

2 Reinforcement Learning

RL algorithms are based on the interaction between an agent and its environment, as shown in Fig. 1. The agent is the learner which interacts with the environment, making decisions according to observations made from it. The environment is formed by the conditions that are external to the agent [7]. The task of the learning agent is to optimize a certain objective function; this optimization is carried out using only information from the state of the environment, i.e. without any external teacher. RL algorithms are basically defined by the following terms:

- **Set of states (S):** Set formed by all the possible states.
- **State of the environment (s_t):** Available information to define the environment at time t.
- **Set of Actions (A):** Set formed by all the possible actions that can be made.
- **Action (a_t):** Action taken by the agent at time t.
- **Policy ($\pi(s,a)$):** Probability distribution over the actions in the state s.
- **Immediate reward (r_{t+1}):** Value returned by the environment to the agent depending on the action at time $t+1$.
- **Long-term reward (R_t):** Sum of all the immediate rewards throughout a complete decision process. It is the objective function that the agent is interested in maximizing by taking the right actions [7]. Its mathematical expression is:

$$R_t = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \tag{1}$$

  where parameter $\gamma \in [0,1]$ is called the *discount-rate*. This factor determines the present value of future rewards. Small values of γ indicate that only next rewards are taken into account. However, as γ approaches 1, further future rewards become more and more relevant.

The goal is to maximize R_t executing a policy that selects the optimal action in each state. Therefore, an estimation of the expected R_t as the result of an action a_t from a state s_t is required. This estimation is usually called *action-value function*:

$$Q^{\pi}(s,a) = E_{\pi}[R_t | s_t = s, a_t = a] \tag{2}$$

where $Q^{\pi}(s,a)$ is the expected R_t starting at state s and taking action a, following policy $\pi(s,a)$. Therefore, the optimal policy is given by the following expression [7]:

$$\pi^*(s,a) = \arg\max_{a \in A} Q^{\pi}(s,a) \tag{3}$$

RL algorithms are devoted to the computation of the action-value function $Q^{\pi}(s,a)$ for a given arbitrary policy in order to obtain the optimal policy using (3). Numerous

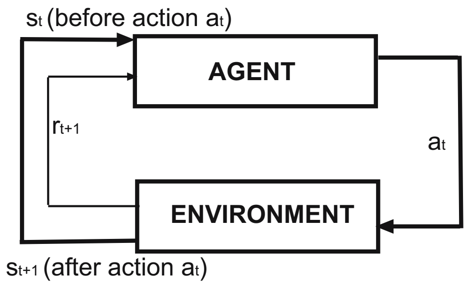


Fig. 1. Characterization of the Reinforcement Learning model. a_t is the action taken by the agent at time t, s_t is the state of the environment at time t, and r_{t+1} denotes the reward at time $t+1$.

methods for carrying out this task are proposed in the bibliography, but we focus on Temporal Difference methods (TD), since techniques of this kind do not require a model of the environment [7].

As it was discussed in Section 1, the use of off-line methods in a real medical application is more convenient than on-line methods, since the latter would imply the experimentation with potentially non-optimal policies on real patients. Therefore, an off-line algorithm (Q-learning) was used, whose update is described as follows:

$$Q_{t+1}(s_t, a_t) \leftarrow Q_t(s_t, a_t) + \alpha \left[r_{t+1} + \gamma \max_{a \in A}(Q_t(s_{t+1}, a)) - Q_t(s_t, a_t) \right] \quad (4)$$

where α is the step-size or learning rate of the update. Q_t stands for the action-value function for a particular state before being visited at time t. Q_{t+1} is the updated value of that state once it has been visited.

Methods that explicitly save the different values obtained from the interaction are called tabular methods because $Q(s, a)$ becomes a table with the state and the action as entries. As the number of states and/or actions increases, the Q-table becomes too big (combinatorial explosion) [7]. In addition, generalization capabilities are required to handle new data. This problem is usually solved by using a function approximator to predict $Q(s, a)$ [11].

In order to carry out the training, R_t, can be computed by different methods. Although the use of Monte-Carlo (MC) methods is theoretically recommended for convergence reasons, those methods which do bootstrap (Dynamic Programming and TD) usually perform better than MC methods in practice [7]. An additional advantage of those methods which do bootstrap is that it is not necessary to compute the return at the end of the episode. Therefore, the TD-style return was used as desired target for the Q-function approximation:

$$target = r_{t+1} + \gamma Q_t(s_{t+1}, a_{t+1}) \quad (5)$$

In this work, we used an Artificial Neural Network (ANN), such as the Multilayer Perceptron (MLP), as function approximator [12]. The inputs to the MLP are given by the variables that define the state and also by the taken action. The desired output of the MLP is given by (5).

3 Problem Modeling

3.1 Description of the Problem

Physicians tend to use protocols for drug administration that are developed from the so-called population models. However, this approach is not suitable for the problem tackled in this work, and it is preferable to focus on an individualized treatment instead. This is due to two reasons; first, the response to the treatment with EPO is highly dependent on the patient; and second, it may be very dangerous to apply a trial-and-error procedure in real patients.

Using data from 2005, we obtained a policy that turned out to be considerably better than that used in the hospital in terms of the percentage of patients that were within the targeted range of Hb, and also in the economic savings involved. The best results were obtained using the Q-learning algorithm for already experienced situations and the function approximator for those cases that had not yet been previously experienced. In particular, being N_m the number of monitorings and I the best value of the Q, i.e., the value associated to that action which guarantees the maximum value of the Q-function (greedy approach), the hospital policy evaluated by the SARSA algorithm showed a value of $\sum_{i=1}^{N_m} I_h(i) = 2359.7$, whereas the RL policy had a value of $\sum_{i=1}^{N_m} I_q(i) = 2922.0$, which involved an increase of 19% in the value of $\sum_{i=1}^{N_m} I(i)$, which was the used value to evaluate the goodness of the policy according to the rewards shown in (6):

$$\text{reward} = \left\{ \begin{array}{ll} 10, & if\ Hb \in [11.5 - 12.5] g/dl \\ \\ 0, & \begin{array}{l} if\ Hb \in [10.0 - 11.5) g/dl \\ or\ Hb \in (12.5 - 13.0) g/dl \end{array} \\ \\ -5, & \begin{array}{l} if\ Hb \in [9.0 - 10.0) g/dl \\ or\ Hb \in [13.0 - 14.0) g/dl \end{array} \\ \\ -10, & otherwise \end{array} \right\} \tag{6}$$

Our proposed policy guaranteed that Hb levels of patients were closer to the desired range of Hb than using the hospital protocol. Moreover, the proposed policy might involve a considerable economic saving for the hospital. The percentage of patients undergoing chronic renal failure represents between 0.1% and 0.2% of the population. The investment in ESFs of the Pharmacy Unit (PU) of the hospital analyzed in this study is around €30,000 per year. This PU has estimated than an optimal policy would save between €100 and €200 per patient and year, which would involve a rough saving of €1,000,000 for all Valencian Region[1].

3.2 Data Collection

In this work, we focus on validating the previously obtained policy with a new cohort of patients, who were treated during the year 2006. Patients were monitored monthly. Seventy-seven patients were analyzed during 2006. It represents a robust test for the policy obtained during 2005, since it was obtained using a cohort of 64 patients. Several factors for each patient were collected in their follow-up: plasmatic concentration of Hb (g/dl), Haematocrit concentration (%), ferritin (mg/l), administration of Intra-Venous Iron (IV Fe) (mg/month), number of administrations of IV Fe, weekly dosage of EPO beta (International Units, IU), weekly dosage of darbepoetin alpha (μg), and number of administrations of these ESFs. Since Hb and Haematocrit were linearly correlated, only Hb was considered in ulterior analyses. Moreover, dosage of IV Fe and number of administrations of IV Fe also contained the same information since all the administrations consisted of 100 mg.

[1] Region of Valencia is a Spanish region with approximately 5 millions of inhabitants and an autonomous government, which manages its own Health Care System.

3.3 Experimental Setup

In order to accommodate our data to the RL algorithms, we considered each monthly test as an episode. Seven variables were used to define the state space: Hb level, ferritin, dosage of IV Fe, dosage of EPO beta, dosage of Darbepoietin alpha and number of administration of these ESFs.

Percentile analysis as well as expert advice were used to define the most adequate thresholds for the ranges used to discretize the variables. In particular, the variable Hb was divided into seven ranges[2], ferritin into five ranges, IV Fe dose into four ranges, EPO beta into five ranges, number of administrations of EPO beta into five ranges, and finally, both Darbepoietin alpha and number of administrations of Darbepoietin alpha into four ranges.

Nine actions were used: increase/decrease EPO beta dosage; increase/decrease Darbepoietin alpha dosage; increase/decrease number of administrations of EPO beta; increase/decrease number of administrations of EPO Darbepoietin alpha; maintain dosage and number of administrations with no changes. Rewards depended on the Hb level, and were defined according to experts' advice, as shown in Eq. (6).

3.4 Validation Results

The application of the previously obtained policy to the cohort of patients treated during 2006, showed an improvement in the value of $\sum_{i=1}^{N_m} I(i)$ equal to 15%. It involved an increase of 22% in the number of patients that were within the desired range of Hb at the end of year 2006. It guarantees a better Quality-of-Life, and it also involves considerable economic savings for the HCS, as it was discussed in Section 3.1. Therefore, the achieved results indicate that the RL policy is robust and generalizable to patients different from those used to obtain the policy.

4 Conclusions

We have presented a work focused on the application of RL to Pharmacy, which is a field of knowledge very different to the typical ones that are related to RL, such as Robotics. The proposed approach is completely general and can be applied to any problem of dosage optimization, given certain states and actions that can be made.

The obtained results are really encouraging, since they show that the policy obtained using data from year 2005, is also suitable when it is applied to a different set of patients, what is definitively the final goal of the proposed methodology.

Our ongoing research is related to the study of new methods to define the state and action spaces efficiently and automatically. Future work will be related to the improvement of the policy using feedback from the new data.

It would also be very interesting to analyze the monthly evolution of the ratio of patients that are within the desired range of Hb; this would help answer questions like: How long does it take for a certain patient to reach this desired target? Are they stable?,

[2] These seven ranges were the same as those used to select the different values of rewards. These values were selected by the nephrologists due to their medical relevance.

i.e., once they are within the desired target, what is the probability to be out of the range again in the future? Is the evolution slow and progressive?

References

1. Gaweda, A.E., Jacobs, A.A., Brier, M.E., Zurada, J.M.: Pharmacodynamic population analysis in chronic renal failure using artificial neural networks: a comparative study. Neural networks 16, 279–292 (1992)
2. Bellazzi, R.: Drug delivery optimization through bayesian networks: an application to erythropoietin therapy in uremic anemia. Computers and Biomedical Research 26, 274–293 (1992)
3. Bellazzi, R., Siviero, C., Bellazzi, R.: Mathematical modeling of erythropoietin therapy in uremic anemia. Does it improve cost-effectiveness? Haematologica 79, 154–164 (1994)
4. Jacobs, A.A., Lada, P., Zurada, J.M., Brier, M.E., Aronoff, G.: Predictors of hematocrit in hemodialysis patients as determined by artificial neural networks. Journal of American Nephrology 12, 387A (2001)
5. Martín, J., Soria, E., Camps, G., Serrano, A., Pérez, J., Jiménez, N.: Use of neural networks for dosage individualisation of erythropoietin in patients with secondary anemia to chronic renal failure. Computers in Biology and Medicine 33(4), 361–373 (2003)
6. Martín, J., Camps, G., Soria, E., Serrano, A., Pérez, J., Jiménez, N.: Dosage individualization of erythropoietin using a profile-dependent support vector regression. IEEE Transactions on Biomedical Engineering 50(10), 1136–1142 (2003)
7. Sutton, R., Barto, A.: Reinforcement Learning: An Introducion. MIT Press, Cambridge, MA, USA (1998)
8. Watkins, C., Dayan, P.: Q-learning. Machine Learning 8, 841–845 (2003)
9. Gaweda, A.E., Muezzinoglu, M.K., Aronoff, G.R., Jacobs, A.A., Zurada, J.M., Brier, M.: Individualization of pharmacological anemia management using reinforcement learning. Neural Networks 18, 826–834 (2005)
10. Martín, J., Soria, E., Chorro, V., Climente, M., Jiménez, N.: Reinforcement learning for anemia management in hemodialysis patients treated with erythropoietic stimulating factors. In: European Conference on Artificial Intelligence 2006, Proceedings of the Workshop Planning, Learning and Monitoring with uncertainty and dynamic worlds, Riva del Garda, Italy, pp. 19–24 (2006)
11. Bertsekas, D., Tsitsiklis, J.: Neuro-Dynamic Programming. Athenas Scientific, Beltmont, USA (1996)
12. Haykin, S.: Neural Networks: A Comprehensive Foundation, 2nd edn. Prentice-Hall, Upper Saddle River, NJ, USA (1999)

Pixel-Based Colour Image Segmentation Using Support Vector Machine for Automatic Pipe Inspection

John Mashford, Paul Davis, and Mike Rahilly

Commonwealth Scientific and Industrial Research Organisation
PO Box 56, Highett, Vic. 3190
Australia
{John.Mashford, Paul.Davis, Mike.Rahilly}@csiro.au

Abstract. This paper presents a new approach to image segmentation of colour images for automatic pipe inspection. Pixel-based segmentation of colour images is carried out by a support vector machine (SVM) labelling pixels on the basis of local features. Segmentation can be effected by this pixel labelling together with connected component labelling. The method has been tested using RGB, HSB, Gabor, local window and HS feature sets and is seen to work best with the HSB feature set.

1 Introduction

In this paper we propose an approach to image segmentation for automatic pipe inspection based on colour images.

The PIRAT system [1] was made up of an inspection device providing range images (in which the pixel values are distances) together with an interpretation system using neural networks and other AI techniques [2]. Here we consider inspection systems based on colour images.

We assume that the inspection system has the ability to combine together all of the obtained images to obtain a single unwrapped pipe image such as that of which part is shown in Figure 1 (for example the Panoramo system of the Ibak company has this capability).

2 The Segmentation Method

There are a number of approaches to colour image segmentation including histogram thresholding, feature based clustering, region-based approaches, edge detection approaches, fuzzy approaches and neural network approaches. The segmentation method used in the PIRAT system was successful. Therefore it seems reasonable to try and apply it to the case of colour images.

This method used in PIRAT is a supervised method which means that the segmentation system is trained by using training examples which are labelled with their correct classifications. The method of segmentation involved feature extraction followed by classification of the feature vector [3]. In the PIRAT project differential

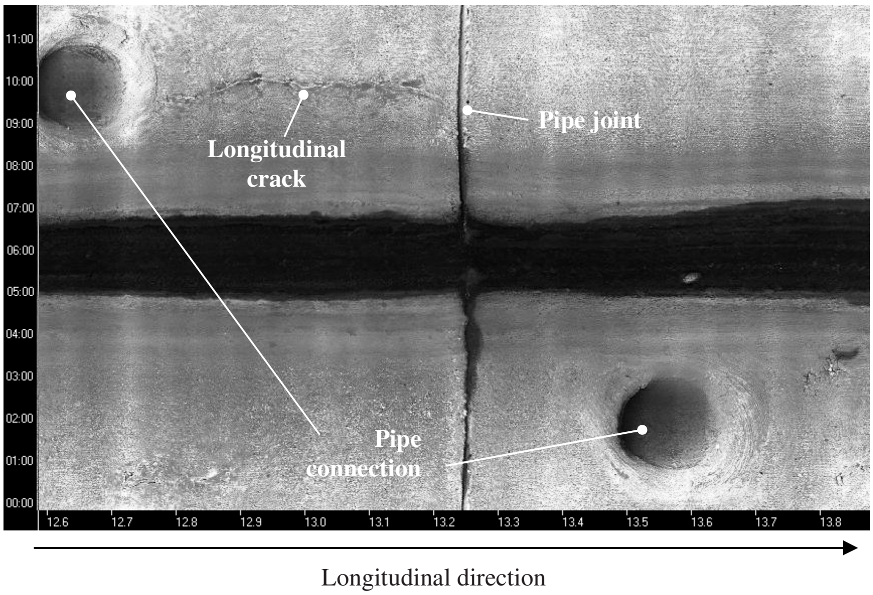


Fig. 1. A section of an unwrapped pipe image

geometric features were used and the classifier was a nearest neighbour classifier. Differential geometric features are mainly only applicable to range images. Therefore other features will have to be used.

A number of authors have used a feature vector consisting simply of the RGB (red, green and blue) values of a pixel as inputs to a classifier [4].

We have also used as a feature set the H, S and B components in the HSB (hue, saturation and brightness) colour space [5]. In order to obtain a segmentation system which is independent of lighting conditions we have also tried a feature set consisting of the H and S components of the HSB colour space.

In addition, a slightly more sophisticated feature set has been implemented in which the max, min and average are computed for RGB values in a window (of size say 7x7) about each pixel together with the max, min and average of an intensity feature I given by

$$I = \sqrt{(R^2 + G^2 + B^2)} \,. \tag{1}$$

Also we have used a more elaborate feature set constructed from Gabor texture filter banks [6]. Gabor filters are useful for analysing the texture properties of images. An even-symmetric Gabor filter is obtained by convolving a unit impulse response function of the form

$$h(u,v) = h(u_0,\theta)(u,v) = h(u_0,\theta,u,v) = \exp\{-(1/2)[x^2/\sigma_x^{\ 2} + y^2/\sigma_y^{\ 2}]\}\cos(2\pi u_0 x) \,, \tag{2}$$

where

$$x = u\cos(\theta) - v\sin(\theta) \,,$$

$$y = u\sin(\theta) + v\cos(\theta) \,,$$

with an input image. The Gabor filter $\Im(u_0,\theta)$ corresponding to spatial frequency u_0 and orientation θ is given by

$$\Im(u_0,\theta)\omega = h(u_0,\theta)*\omega\,, \tag{3}$$

for $\omega : \mathbf{Z}^2 \to [0,\infty)$,
where

$$(h(u_0,\theta)*\omega)(i,j) = \sum_{k=-\infty}^{\infty}\sum_{l=-\infty}^{\infty}\omega(i-k, j-l)h(u_0,\theta,k,l)\ . \tag{4}$$

Since the mask $h(u_0,\theta)$ vanishes exponentially with increasing distance from the origin it can be well approximated by a finite mask. The filter can be considered to act on finite images by null extending finite images to infinite images.

A bank of Gabor filters is obtained by taking a set of values for the radial frequency u_0 and the orientation θ. We have used a set of radial frequencies of the form $\{\sqrt{2}\ 2^{-2-i} : i = 0, 1,. \ldots, n\_u_0-1\}$ where $n_u_0 \in \{1, 2, \ldots\}$and a set of orientations of the form $\{0, \pi/4, \pi/2, 3\pi/4\}$ [6]. The above radial frequencies are an octave apart.

For the classifier we used an SVM. SVMs have certain properties that make them superior to neural networks such as requiring smaller training sets and having better generalisation ability. We used the LIBSVM system of the National University of Taiwan with radial basis function kernel.

For generation of a training set for the SVM a set of feature vectors for individual pixels together with their classifications must be extracted from the training data images. The feature vectors can be extracted automatically but the classification labels must be determined manually (if they could be determined automatically then we would have already solved the problem of segmentation).

A simple approach to generating training sets for the pixel classifier is to select rectangular regions of constant classification using standard tools. There will be a problem with this method if regions of constant classification do not contain rectangular subsets of a reasonable size.

This approach was used for the experiments of the present paper. A number of rectangular regions were selected and extracted from the unwrapped pipe image of a concrete sewer pipe (part of which is shown in Figure 1). The regions were extracted from regions of corrosion and regions of good pipe. Two files called good_1.bmp and good_2.bmp containing regions of good pipe were used for testing the classifier while the remaining corrosion and good pipe files were used for both training and testing.

The bitmap files were processed by a program to produce text files containing the RGB values. The RGB values were then processed to produce feature vectors, either RGB, HSB, Gabor, HS or local window features.

Training sets and testing sets were then randomly selected from the feature vector files. Then the data was rescaled and the SVM training was carried out. The results of testing the pixel classifier on the testing sets that had been set aside during training are shown in Table 1. The table shows that all feature sets except the HS feature set result in good testing results on the data that had been set aside and on good_1.bmp and good_2.bmp. Training and testing times were of the order of a few seconds.

In order to carry out wider testing, the SVM pixel classifier algorithms were applied to the image of Figure 1. The result for the HSB case is shown in Figure 2.

The image labelling results for the RGB and Gabor feature set cases were very similar to but not quite as good as the HSB case. The result in the case of the HS feature set was a very poor segmentation.

The segmentation in the Gabor case is slightly worse than the segmentation in the RGB case. This is because the pipe joint (appearing as the thin vertical stripe in the centre of the image) breaks up to a certain extent with the result that after connected component labelling it would form more than one defect or feature region.

The single pixel defects and other very small defects which are scattered through the RGB, Gabor and HSB pixel labelling images, especially in the lower left side of the images, are perhaps not a problem because such defects can be filtered out at a higher level by the pipe inspection system during image classification and analysis.

The local window based pixel classifier image while not containing very small defects contains regions which are square artefacts of the window method. Also the pipe joint does not extend across the full width if the pipe.

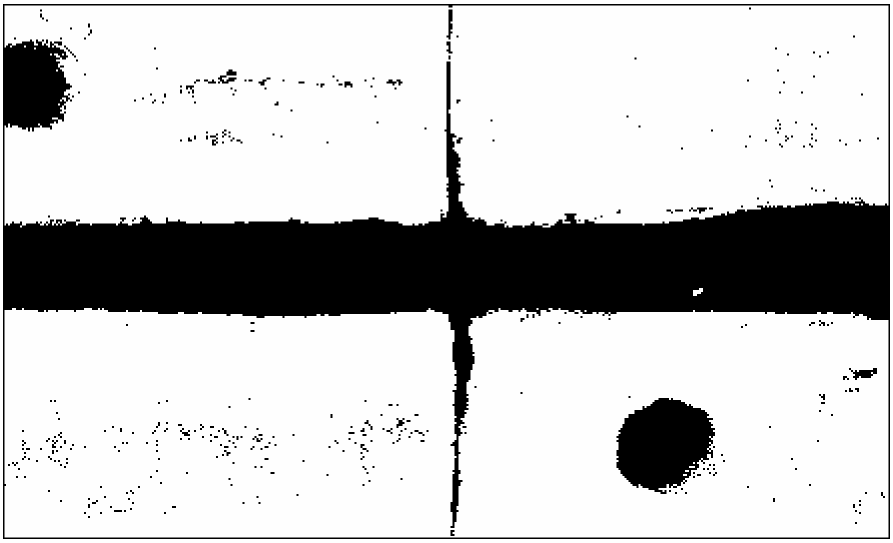


Fig. 2. HSB-SVM Pixel labelling

Table 1. Classifier test results

Feature Set	No. of training cases	No of Testing cases	% accuracy of testing	% accuracy on good_1.bmp	% accuracy on good_2.bmp
RGB	3000	17867	96.88	99.15	99.93
HSB	1000	19867	97.30	99.39	99.98
Gabor	5000	15867	97.11	99.25	99.92
Window	3999	10100	99.83	99.98	100.00
HS	1000	19867	67.81	64.11	92.61

The best result for segmenting the image of Figure 1 is obtained by using the HSB feature set. Both the pipe joint and the crack on the top left hand side of the image are comparatively strongly demarcated. Similar results have been obtained for other test images but tests have shown that the system may need to be retrained for different pipe types.

3 Conclusions

A new approach to pixel based segmentation of colour images using SVM for automatic pipe inspection has been presented. The approach has been tested using RGB, Gabor, local window, HSB and HS feature sets. It has been found that the method works best in the case of the HSB feature set. One advantage of using colour images for segmentation rather than range images as in the PIRAT system is that regions of discolouration indicating the development of corrosion or other defects can be detected while still not having any effect on the pipe geometry.

Acknowledgement

The authors thank Scott Gould for help with this work. Also the authors would like to thank Tristan Day for supplying the colour images of sewers.

References

1. Kirkham, R., Kearney, P.D., Rogers, K.J., Mashford, J.: PIRAT - A system for quantitative sewer pipe assessment. The International Journal of Robotics Research 19(11), 1033–1053 (2000)
2. Mashford, J.S.: A neural network image classification system for automatic inspection. In: Proc. Of the 1995 IEEE International Conference on Neural Networks, Perth, Australia (1995)
3. Blanz, W.E., Gish, S.L.: A real-time image segmentation system using a connectionist classifier architecture. International Journal of Pattern Recognition and Artificial Intelligence 5(4), 603–617 (1991)
4. Gomez-Moreno, H., Gil-Jimenez, P., Lafuente-Arroyo, S., Vicen-Bueno, R., Sanchez-Montero, R.: Color image segmentation using the support vector machines. Recent Advances in Intelligent Systems and Signal Processing, 151–155 (2003)
5. Gonzalez, R., Woods, R.: Digital Image Processing, 2nd edn., p. 295. Prentice Hall Press, Englewood Cliffs (2002)
6. Randen, T., Husøy, J.H.: Filtering for texture classification: a comparative study. IEEE Trans. Pattern Anal. Machine Intell. 21(4), 291–310 (1999)

An Approach to Spanish Subjunctive Mood in Japanese to Spanish Machine Translation

Manuel Medina González and Hirosato Nomura

Kyushu Institute of Technology,
680-4 Kawazu, Iizuka, Fukuoka, 820-8502, Japan
manuel@dumbo.ai.kyutech.ac.jp, nomura@ai.kyutech.ac.jp
http://dumbo.ai.kyutech.ac.jp/nomura-ken

Abstract. This paper discusses a model to decide when the predicates of various types of sentences should use Spanish subjunctive mood in Japanese to Spanish machine translation. It consists of a hierarchy for the elements of the sentences and a series of rules applied to them at transfer phase. We compare our results against some commercial machine translation systems. Our experiments show that our model outputs more accurate results, a necessary step to get a fairly good and natural translation.

1 Introduction

In machine translation, analyzing and translating the mood is still a difficult task. In Japanese to Spanish machine translation, most of the existent systems seem to translate via English, an strategy that cannot necessarily generate a correct and high quality translation. In order to overcome such inconvenence, this paper presents an approach to the direct translation from Japanese to Spanish, focusing on Spanish *subjunctive mood*. This is one of the most difficult topics in this approach because Japanese does not subdivide it in detail. For further information about Spanish subjunctive mood, [4] and [5] may be a good source of information for English speakers, and [6] for Japanese speakers.

In this paper, we propose a model that analyzes a number of patterns and sentences in order to get the correct tense and mood for each predicate within a sentence. The model consists of a hierarchy for the elements of the sentences and rules applied at transfer phase. We test the accuracy of those rules in a prototype system and compare the results against others obtained using commercial software.

2 Model

2.1 Hierarchy

We propose the following hierarchy: *Paragraph*, *conjunction*, *sentence*, *embedded sentence* and *bunsetsu*. A paragraph is basically the input, where each element has a defined position, and the relationships between them have been already identified. We divide the sentences by applying a very simple and similar version of the algorithm presented in [8]. Thus, we get some *key bunsetsu* (a bunsetsu is the smallest meaningful sequence in Japanese) with a conjunctive element within. We consider these as *conjunctions*. Each conjunction will have at most 2 *sentences*: Left and right. If there is more than one conjunction, the right sentence of a conjunction becomes the left sentence of the next. This process repeats until we reach the last sentence. Here, if there are elements that modify elements in other sentences, we rewrite the sentences and update the relationships between them.

2.2 Applying the Model

We create a series of rules and apply them to the predicates of the sentences at transfer phase. The rules are applied in word, sentence and paragraph level in a top-down sequence, being word the highest level (one) and paragraph the lowest (three). A rule in a low level has more priority that one in an high level. However, some special expressions can give top-priority to a rule in an high level.

We always consider sentences in pairs. Left sentences are considered *subordinated sentences* and right sentences are considered *main sentences*.

The flow of the model is shown in Fig. 1. The phrase used is 山に行って弁当を食べた (I went to the mountain and ate a lunch box):

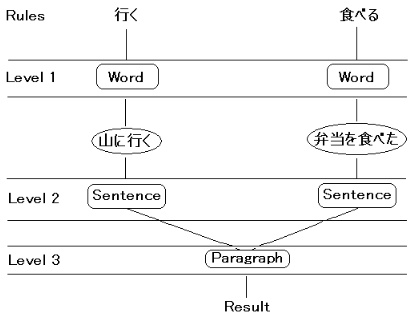


Fig. 1. After passing the 3 levels, we get a final result for each predicate

2.3 Rules

Examples for all the rules are presented in [2].

Verbs that express feelings. We create a list of verbs. If the predicate of the main sentence is contained in this list, then we change the mood of the verb of the subordinated sentence to subjunctive mood, providing that the subjects of the sentences are different. Also, the tenses of both verbs must agree: If the main verb is in present tense, the verb of the subordinated sentence must be in present tense, and if it is in past tense, then the subordinated sentence must be also in past tense.

Causative sentences. The use of subjunctive mood in causative sentences is optional. If we choose to use it, then first we conjugate the auxiliary verb. Next, we need to add the Spanish conjunction "que" before the conjugation in subjunctive mood of the main verb. Last, we set the tense of the main verb according to the tense of the auxiliary verb.

Noun clauses modified by embedded sentences. First, we search for volition in the predicate of the main sentence by finding the auxiliary verb "たい"(tai). Then we check the tenses of the verbs of both sentences and we decide the mood and the final tense of the translation according to the information presented in[2].

Impersonal expressions. This category refers to paragraphs whose subject is a sentence (considered as embedded) and its predicate is an expression like 必要だ (necessary), 難しい (difficult), 無駄だ (pointless), among others.

Subjunctive mood is used when there is a subject in the embedded sentence. If there is no subject, then indicative mood is used. Also, we use the same tense in all the predicates.

Expressing an opposite reality. Here, we will find paragraphs containing the conditionals "ば" (ba) and "たら" (tara), and, sometimes, some auxiliary particles at the end of the main sentence to emphasize the meaning. They generally indicate the speaker's feelings toward what he or she is saying. Those particles are: "に" (ni), "たろう" (darou), "のに" (noni) and "な" (na).

In order to apply the rules for this category, we have to consider that (1) "に" is only used after "だろう"; (2) "だろう" can be omitted if there's no particle after it; and (3) "だろう" is never used together with "のに".

As the conditional expressions "ば" and "たら" are used in this category, their grammatical differences become crucial to determine the tense and mood of the predicates of the paragraph. Based on the statements in [6] and in [9], we create a series of rules for each of them.

Special expressions. They are divided according to their behavior. As there are many expressions in this category, we only mention some of them here: "前" (mae), verbs in て (te)-form + "から", "限り", "次第", among others.

Special verbs. There are some verbs that do not behave like the others from the point of view of subjunctive mood, and as they can not be classified into some of the categories already explained, we handle each of them separately: ある, いる *(to be, to exist)*, 探す（捜す）*(to search for, to look for, to seek for)* and 知る *(to know).*

3 Evaluation

We implemented a prototype system that analyzes sentences and applies the explained rules to select a correct mood and tense for all the predicates within them. The system is based in the model presented in [3], and 239 full sentences to be evaluated were extracted from [9], [6], [7], [10], and [11].

To compare the results obtained by this model, we also tested the sentences in two machine translation systems: 同時通訳、日本語ースペイン語 (DojiTsuyaku, http://www.ifour.co.jp), and the SystranBox (http://www.systran.co.uk/). In the former, we could translate directly from Japanese to Spanish, but in the latter we had to translate first to English and, from there, to Spanish.

As the main purpose is to test the accuracy of our rules, we prefer to evaluate the whole input as one entity instead of the predicates within it. The evaluation includes 3 possible values: *Correct*, if the correct mood and tense were used in all the predicates within the sentence; *incorrect*, if at least one of the predicates is using the incorrect mood or tense; and *error*, a value only present in our system that refers to errors caused because the results of the morphological and syntactic analysis, delegated to external tools as mentioned in [3], were not correct. Also, the quality of the obtained translation is not relevant.

The results are shown in Table 1.

Table 1. Evaluation results

Evaluation	Our system	DojiTsuyaku	Systran
Correct	63.59%	51.89%	42.26%
Incorrect	30.12%	48.11%	57.74%
Error	6.29%	0%	0%

The results prove that our model is more accurate. However, the incorrect results reveal that some of the rules are not enough to analyze certain types of sentences in a category. For instance, the cases when "ば" or "たら" are not exactly used as conditionals, as in 風邪が吹けば桶屋が儲かる (After wind blows basinsmiths get money). As for the error category, the rate might be reduced only by editing the sentences in order to force a more correct analysis of the input. There are, however, cases where even editing is not enough. For instance, when the conjunction "のに" is considered as 2 different word. This leads to a wrong decision of the relationships between the elements, causing that rules are not triggered, or that unexpected errors occur in our system.

4 Discussion

It is evident that the created rules are still not enough to cover all the possible cases where subjunctive mood must be used. Moreover, sometimes it is very difficult to output an accurate translation because the lack of context knowledge. In many cases, the analysis and addition of elements that do not exist in the source language is the key point to get an accurate translation. For instance, in the sentence 彼女は頭が良いと彼らは思ってない (they don't think she's intelligent), if we do not have information regarding what the speaker knows about the subject of the subordinated sentence 彼女は頭がいい (she is intelligent), we can not do anything but output two results: One where 頭が良い (to be intelligent) is in indicative mood, and the other where it is in subjunctive mood. If we want to output the best result for a certain situation, we need to get information regarding the knowledge of the speaker, because selecting the mood in this case depends on it. A more detailed explanation about this type of sentences can be found in [3].

This type of analysis, the addition of rules and cases where subjunctive mood must be used and the implementation of those rules in a machine translation system are part of our future work.

5 Summary

In this paper, we proposed a model to approach to the correct use of Spanish subjunctive mood in Japanese to Spanish machine translation. It consists of a hierarchy for the elements of the sentences and a series of rules that are applied to certain types of Japanese sentences to identify, analyze and process patterns in order to decide whether subjunctive mood should be used. We have empirically demonstrated the effectiveness of our method by running experiments and comparing the results with two commercial machine translation software. Our future work shall focus on the analysis of the context of the sentences and the addition of cases where subjunctive mood must be used.

References

1. 野村浩郷:, 自然言語処理の基礎技術, 電子情報通学会編, 日本, (1993)
2. Manuel Medina González and Hirosato Nomura:, A Model to Decide the Correct Mood in Japanese to Spanish Machine Translation, Technical Report in Computer Science and Systems Engineering, Kyushu Institute of Technology, to appear
3. Manuel Medina González and Hirosato Nomura:, Spanish Case Information Analysis and its Application to Japanese to Spanish Machine Translation, Master Thesis, Kyushu Institute of Technology, Japan, (2006)
4. Aoife Ahern and Manuel Leonetti:, The spanish subjunctive: Procedural semantics and pragmatic inference, in "Rosina Marquez-Reiter and Mara Elena Placencia: Current Trends in the Pragmatics of Spanish", 35–57, John Benjamins Publishing Company, (2004)
5. Mark Jary:, Mood in relevance theory: a re-analysis focusing on the spanish subjunctive, UCL Working Papers in Linguistics, 14:157–187, (2002)
6. 小池和良:, スペイン語作文の方法 構文編, 第三書房, 東京, (2002)
7. 田近洵一:, くわしい国文法, 文英堂, 東京, (2002)
8. Sadao Kurohashi and Makoto Nagao:, Dynamic programming method for analyzing conjunctive structures in Japanese, Proceedings of the 14th conference on Computational linguistics Vol. 1 170–176, (1992)
9. グループ ジャマシイ:, 日本語文型辞典, くろしお出版, (2006)
10. 九州地区中学校教育研究協議階会編:, 中学生のための国文法　美しい日本語, 九州教科研究協議会, 平成 10 年
11. NTT Natural Language Research Group:, http://www.kecl.ntt.co.jp/mtg/resources/index.php

Reasoning About Hybrid Systems Based on a Nonstandard Model

Katsunori Nakamura[1,2] and Akira Fusaoka[1]

[1] Department of Computer Science, Ritsumeikan University
Nojihigashi, Kusatsu-city, SIGA, Japan 525-8577
[2] Heian Jogakuin (St.Agnes') University
Takatsuki-city, OSAKA, Japan 569-1092
fusaoka@cs.ritsumei.ac.jp

Abstract. In this paper, we propose to introduce a nonstandard analysis into a logical modeling of continuous dynamics and present a new framework called hyper-finite hybrid automaton (HHA). HHA is a nonstandard interpretation of hybrid automata in the domain of $^*\mathbb{R}$. We also enlarge the linear temporal logic LTL to *LTL to describe the system specification. By using this framework, we examine the validation of the system consistency of the hybrid system, especially the existence and reachability of Zeno point.

1 Introduction

In this paper, we propose to introduce a nonstandard analysis into a logical modeling of continuous dynamics and present a new framework called hyper-finite hybrid automaton (HHA). HHA is a nonstandard interpretation of hybrid automata in the domain of $^*\mathbb{R}$. We also enlarge the linear temporal logic LTL to *LTL to describe the system specification. By using this framework, we examine the validation of the system consistency of the hybrid system, especially the existence and reachability of Zeno point.

The advantages of using a nonstandard framework are:

(1) The continuous dynamics is reduced to the recurrence equation that represents the infinite iteration of infinitesimal discrete changes with infinitesimal duration. Therefore, we can handle the hybrid dynamics based only on discrete paradigm.
(2) The representation of dynamics based on nonstandard analysis is complete and the limit point of discrete change, like Zeno, can be handled.

2 Hyper-Finite Hybrid Automata

For the discrete modeling of the continuous dynamics, we introduce a new type of automata called hyper-finite hybrid automata on the basis of nonstandard analysis. The essential point of this automaton is the replacement of continuous dynamics (differential equation) by the infinite iteration of infinitesimal change (Hyper finite recurrence equation).

Definition 1. Hyper-real $^*\mathbb{R}$
$^*\mathbb{R}$ is an enlargement of $\mathbb{R}$. Namely, it is an interpretation of the axioms of $\mathbb{R}$ in the domain including infinitesimal and infinite numbers. We introduce two constant ε: infinitsimal and ω infinite number, where $\varepsilon \times \omega = 1$. We also introduced an equivalence relation $x \approx y$ that means $x - y$ is infinitesimal.

Definition 2. Hyper-finite hybrid automaton (HHA)
A hyper-finite hybrid automaton $\mathcal{A}$ on $^*\mathbb{R}$ is a 11-tuple
$\mathcal{A} = (Q, \Sigma, G, E, I, F, \Xi, \Xi_0, X, X_0, D)$ *where*

1. *Q is the finite set of states.*
2. *Σ is the finite set of* **actions** *each of which is label of each element of D.*
3. *G is the finite set of guard conditions.*
4. *E is the finite set of transition rules; $E \subseteq (Q \times \Sigma \times Q)$.*
5. *$I \in Q$ is the initial state. $X_0 \in X$ is the initial situation.*
6. *$F \subset Q$ is the set of the final states.*
7. *Ξ is the set of discrete variables. $\Xi_0 \in \Xi$ is the set of initial values.*
8. *$X \subseteq {}^*\mathbb{R}^n$ is the set of situations.*
9. *D is the set of dynamics for each action of Σ. The dynamics for each action is given in the form of $x' = f(x)$ where $f(x)$ is a mapping $f : {}^*\mathbb{R}^n \to {}^*\mathbb{R}^n$. Each dynamics is labeled by each corresponding action in Σ.*

In a standard hybrid automaton, continuous dynamics is represented by the differential equation at each state, and when some condition is satisfied (or violated), discrete state change will happen. In HHA, there is no continuous change in any state. Every change is discrete and occurs along with the state transition.

Definition 3 (Transition, Execution)

(1) **state transition.** *A word $\alpha = a_1 a_2 \cdots a_n \in \Sigma^\dagger$ has a transition $(q_0, q_1, \cdots q_n)$ of the HHA if and only if there exists $(q_i, a_i, q_{i+1}) \in E$.*
(2) **nonstandard execution.** *Let $\alpha = a_1 a_2, \cdots a_n$ be a sequence of actions of HHA $\mathcal{A}$ with the transition $q_0, q_1, \cdots q_n$. α is $*-$***executable** *if and only if there exists a sequence of situations $w_0, w_1, \cdots, w_n \subseteq X$ such that $w_0 = X_0$. And for all $i = 1, 2, \cdots, n$, if $(q_i, a_i, q_{i+1}) \in E$ and $p(x(w_i))$ holds where $p(x) \in G$ is the guard for the transition of a_i, then $x(w_{i+1}) \approx x(w_i)$.*
(3) **execution in the standard sense.** *Note that each situation w_i of α is always defined over $^*\mathbb{R}$ if it is $*-$executable. However, it is feasible only when all values in w_i have their shadows in the standard world.*

3 The Description of Hybrid System

3.1 The System Description in a Hyper-Finite Hybrid Automaton

Consider a coupling of two water tanks. Let x, y denote the level of water in Tank A and Tank B. We assume that the tap in the bottom of each tank discharges the water at a rate proportional to the level of each tank. Also the constant flow

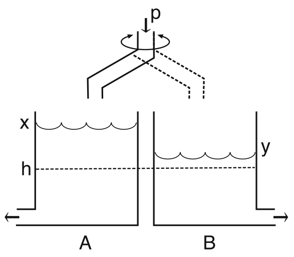


Fig. 1. Alternative water tank

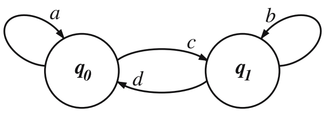


Fig. 2. HHA for the two tanks (before Zeno)

denoted by p of the water is poured exclusively to either Tank A (we call the state A, which is denoted by a proposition A) or Tank B (the state B represented by $\neg A$) at each time (Fig.1).

Therefore, the dynamics of the system is given by

$$A \supset \frac{dx}{dt} + kx = p \wedge \frac{dy}{dt} + ky = 0, \qquad \neg A \supset \frac{dx}{dt} + kx = 0 \wedge \frac{dy}{dt} + ky = p.$$

Assume that we want to make the both tanks to be the same level, namely $x \approx h \wedge y \approx h$. In order to achieve this goal, we select the simple control plan such that pouring into the tank A when $x < h$ and pouring into the tank B when $y < h$. Namely,
if $\ A \wedge x \geq h \wedge y < h$ **then** switch to $\neg A$,
if $\neg A \wedge y \geq h \wedge x < h$ **then** switch to A.
This control strategy is one of Zeno plans. Although it leads the system to the equi-level state starting from the initial state $x > h \wedge y > h$, it requires the infinite state transitions between A and $\neg A$ before reaching the goal. But Zeno plans are asymptotically feasible and sometimes useful in the actual situation.

The HHA for the two tanks is given in the following. The graphical representation is given in (Fig.2).

$\mathcal{A} = (Q, \varSigma, G, E, I, F, \varXi, \varXi_0, X, X_0, D)$ where $Q = \{q_0, q_1\}$, $\varSigma = \{a, b, c, d\}$,
$X = \{x, y, t\}, \quad E = \{(q_0, a, q_0), (q_1, b, q_1), (q_0, c, q_1), (q_1, d, q_0)\}$,
$G = \{a : (x \leq h \vee y > h), b : (x > h \vee y \leq h), c : (x > h \wedge y \leq h)$,
$d : (x \leq h \wedge y > h)\}$,
$D = \{a : (x' = (1 - k\varepsilon)x + p\varepsilon, y' = (1 - k\varepsilon)y, t' = t + \varepsilon, A' = A)$,
$b : (x' = (1 - k\varepsilon)x, y' = (1 - k\varepsilon)y + p\varepsilon, t' = t + \varepsilon, A' = A)$,
$c : (x' = x, y' = y, t' = t, A' = \neg A), d : (x' = x, y' = y, t' = t, A' = \neg A)\}$,
$\varXi = \{A\}$: A takes '$true$' or '$false$'. , $\quad \varXi_0 = \{A = true\}$,
$I = q_0, \quad X_0 = \{x = x_0, y = y_0, w = x_0 + y_0, t = 0\}$,

4 Verification of the Control Plan

4.1 Nonstandard Linear Temporal Logic (*LTL)

Linear temporal logic (LTL) is a modal logic for timed phenomena. Basically, LTL consists of some modal operators like bellow.

$\circ\varphi$: φ is true in the next state.
$\Box\varphi$: φ is always true.
$\Diamond\varphi$: φ will be true at some time in the future.
$\psi\mathcal{U}\varphi$: ψ is true until φ becomes true.

The *LTL is a nonstandard extension of LTL. Namely, *LTL has the same axioms as LTL but all formulas are interpreted on the time structure in $^*\mathbb{N}$ rather than $\mathbb{N}$. For example, it allows a formula $\circ^{\alpha}\varphi$ for any $\alpha \in {}^*\mathbb{N}$, which means φ holds after α(infinite) times of transitions. Also φ may be a predicate on $^*\mathbb{R}$.

4.2 Model Checking

Reachability

The proof of reachability is usually more complicated than that of the stability, because it requires the mathematical induction for the system variable in addition to the inherent induction axiom of temporal logic. We use the following induction schema.

$$(Q(1) \supset \Diamond\varphi) \land \forall n \left[Q(n+1) \supset (\Diamond Q(n) \lor \Diamond\varphi)\right] \supset \forall n \left[Q(n) \supset \Diamond\varphi\right]$$

Note that the mathematical induction holds in $^*\mathbb{N}$ due to the Transfer Principle.

Let us try to prove that if the system is in the state q_0 and $y > h$ then it will eventually reach to q_1.

We represent y by $y = h + k\varepsilon n$. This is possible if we select the appropriate infinite number for n. Let $Q(n) \equiv y < h + k\varepsilon n$, and $\varphi \equiv (y < h)$.

$Q(1) \supset \Diamond\varphi$ because if $y = h + k\varepsilon$ then $y' = (1 - k\varepsilon)y < (1 - k\varepsilon)(h + k\varepsilon) < h$. So that $Q(1) \supset \circ\varphi$. Namely, $Q(1) \supset \Diamond\varphi$. Assume that $Q(n+1)$, namely

$$y = h + k\varepsilon(n+1),$$
$$y' = (h + k\varepsilon(n+1))(1 - k\varepsilon) < h + k\varepsilon n - k\varepsilon(h-1) < h + k\varepsilon n.$$

So that $Q(n+1) \supset \Diamond Q(n)$. By the induction schema, we have $Q(n) \supset \Diamond\varphi$. Finally, we can conclude that $\forall n[(A \land y = h + k\varepsilon n) \supset \Diamond(y < h)]$.

Because the guard $(y < h)$ is eventually satisfied, the system will transfer to the state q_1 at some situation, say s_1. Let us assume s_1 is reached after r times iteration of the action α. Then the levels of both tanks at s_1 are given by

$$y(s_1) = y_0\,(1 - k\varepsilon)^r = h, \;\; x(s_1) = x_0\,(1 - k\varepsilon)^r + \frac{p}{k}\left\{1 - (1 - k\varepsilon)^r\right\}$$

By the calculation of $^*\mathbb{R}$, we have the corresponding standard values

$$y_t \approx y_0 e^{-kt} = h, \;\; x_t \approx x_0 e^{-kt} + \frac{p}{k}\left(1 - e^{-kt}\right) \text{ where } t = \varepsilon r.$$

Reachability for Zeno point

We prove that the Zeno plan leads to the goal such that $(x \approx h) \land (y \approx h)$. We use here the following Weierstrass' theorem rewritten as a formula of temporal logic instead of the mathematical induction (the theorem is true even in $^*\mathbb{R}$ due to the Transfer Principle).

$$\exists a \Box (z \geq a) \wedge \forall b \Box [z = b \supset$$
$$\exists c [b \geq c \wedge \Diamond (z = c) \wedge (z < b) \mathcal{U} (z = c) \supset \exists w \Diamond \Box (z \approx w)]]$$

From the proof of reachability above, the system at the state q_0 sometime transfers to q_1 if $y > h$. Similarly we we can prove that if the system is in the state q_1 and $x > h$ then it will eventually reach to q_0, and then transfers to q_1 and then q_0, so on.

Let's assume that s_{2n} is the situation of transition after n times visit of q_1 and q_0, and s_{2n+1} is the situation after one transition ($q_1 \rightarrow q_0$) from s_{2n}. The levels of water x, y at the end of q_0 or q_1 are described below.

$$x(s_{2n+1}) = \tfrac{p}{k} + \tfrac{h\left(h - \frac{p}{k}\right)}{y(s_{2n})}, \;\; y(s_{2n+1}) = h, \;\; x(s_{2n}) = h, \;\; y(s_{2n}) = \tfrac{p}{k} + \tfrac{h\left(h - \frac{p}{k}\right)}{x(s_{2n-1})}$$

We can prove the relations

$$h < x(s_{2n+1}) < x(s_{2n-1}), \;\; h < y(s_{2n}) < y(s_{2n-2})$$

from the mathematical induction.

Between s_{2n-1} and s_{2n}, $A = \mathit{false}$ holds (at q_1) so that x is decreasing. Therefore, $x < x(s_{2n-1})$. And between s_{2n} and s_{2n+1}, $A = \mathit{true}$ holds (at q_0) so that x is monotonically increasing. Therefore, $x < x(s_{2n+1})$. But due to $x(s_{2n+1}) < x(s_{2n-1})$, we have $x < x(s_{2n-1})$ between s_{2n-1} and s_{2n+1}. Therefore, we have

$$\Box [x = x(s_{2n-1}) \supset [x(s_{2n-1}) \geq x(s_{2n+1}) \wedge$$
$$\Diamond (x = x(s_{2n+1})) \wedge (x < x(s_{2n-1})) \mathcal{U} (x = x(s_{2n+1}))]] \; .$$

Also, x is always bounded, that is, $\Box (x \geq h)$.

From the Weierstrass' theorem, we have $\exists w \Diamond \Box (x \approx w)$. Namely, we have a standard $\hat{x}, \hat{y}, \hat{t}$ such that $x(s_{2n+1}) \approx x(s_{2n-1})$. From $x(s_{2n+1}) \approx x(s_{2n-1}) \approx \hat{x}$ and a recurrence equation, $x(s_{2n+1}) = \frac{p}{k} + \frac{h\left(h - \frac{p}{k}\right)}{\frac{p}{k} + \frac{h\left(h - \frac{p}{k}\right)}{x(s_{2n-1})}}$ we have $\hat{x} = h$. Similarly, we can get $\hat{y} = h$. On the other hand, we have:

$$(x(s_{n+1}) + y(s_{n+1})) = (1 - k\varepsilon)(x(s_n) + y(s_n)) + p\varepsilon$$

from the system description. By solving this recurrence equation, we have $\hat{t} = \frac{1}{k} log \left(\frac{2ck - p}{2hk - p} \right)$. $\hat{t}$ is finite so that we can conclude that the situation $\hat{s} = \left\langle h, h, \frac{1}{k} log \left(\frac{2ck - p}{2hk - p} \right) \right\rangle$ is Zeno point. Namely, $\hat{s}$ is reached during the finite time after repeating infinitely the value change of A.

References

1. Alur, A., Courcoubetis, C., Henzinger, T.A., Ho, P.: Hybrid Automata: An Algorithmic Approach to the Specification and Verification of Hybrid Systems. In: Grossman, R.L., Ravn, A.P., Rischel, H., Nerode, A. (eds.) Hybrid Systems. LNCS, vol. 736, pp. 209–229. Springer, Heidelberg (1993)

2. Ames, A., Abate, A., Sastry, S.: Sufficient Conditions for the Existence of Zeno Behavior. In: IEEE Conference on Decision and Control (2005)
3. Nakamura, K., Fusaoka, A.: On Transfinite Hybrid Automata. In: Morari, M., Thiele, L. (eds.) HSCC 2005. LNCS, vol. 3414, pp. 495–510. Springer, Heidelberg (2005)
4. Podelski, A., Wagner, S.: Model Checking of Hybrid Systems: From Reachability towards Stability. In: Hespanha, J.P., Tiwari, A. (eds.) HSCC 2006. LNCS, vol. 3927, pp. 507–521. Springer, Heidelberg (2006)
5. Robinson, A.: Nonstandard Analysis. North Holland, Amsterdam (1974)
6. Zhang, J., Johansson, K.H., Lygeros, J., Sastry, S.: Dynamical Systems Revisited: Hybrid System with Zeno Executions. In: Lynch, N.A., Krogh, B.H. (eds.) HSCC 2000. LNCS, vol. 1790, pp. 451–464. Springer, Heidelberg (2000)

Merging Algorithm to Reduce Dimensionality in Application to Web-Mining

Vladimir Nikulin[1] and Geoffrey J. McLachlan[2]

[1] Suncorp, Brisbane
{vladimir.nikulin@suncorp.com.au}
[2] Department of Mathematics, University of Queensland
{gjm@maths.uq.edu.au}

Abstract. Dimensional reduction may be effective in order to compress data without loss of essential information. It is proposed to reduce dimension (number of the used web-areas or *vroots*) as a result of the unsupervised learning process maximizing a specially defined average log-likelihood divergence. Two different web-areas will be merged in the case if these areas appear together frequently during the same sessions. Essentially, roles of the web-areas are not symmetrical in the merging process. The web-area or *cluster* with bigger weight will act as an attractor and will stimulate merging. In difference, the smaller cluster will try to keep independence. In both cases the powers of attraction or resistance will depend on the weights of the corresponding clusters. The above strategy will prevent creation of one super-big cluster, and will help to reduce the number of non-significant clusters. The proposed method is illustrated using two synthetic examples. The first example is based on an ideal *vlink* matrix, which characterizes weights of the *vroots* and relations between them. The *vlink* matrix for the second example is generated using a specially designed web-traffic simulator.

Keywords: distance-based clustering, data compression, log-likelihood, web-traffic data.

1 Introduction

A general problem faced in computer science is to reduce the dimensions of large datasets in order to make sense of the information contained in them.

Given a significantly high number of *vroots* and low average number of different pages visited during one separate session, we are interested to group pages into relatively homogeneous clusters in order to avoid sparse tables. For example, [1] considered grouping according to the logically sensible approach. Another approach may be based on statistical methods: for example, we can consider projection pursuit with such special techniques as principal component, discriminant, and factor analyses [2]. The corresponding methods optimize in some sense the linear transformation from the given to the known low-dimensional space. However, in practice, the dimension or number of clusters may not be known [3].

The proposed unsupervised clustering approach is based on the *vlink* matrix (1), and is presented in the following Sect. 2. Section 3 illustrates the main idea behind the proposed method using two synthetic examples. Importantly, further application of the same algorithm with the same settings against *msweb* dataset (*UCI KDD* archive at the University of California) produced the same graphical structure of the target function (see Fig. 2(d)).

As a next step after dimensional reduction we can consider the problem of predicting a user's behavior on a web-site, which has gained importance due to the rapid growth of the world-wide-web and the need to personalize and influence a user's browsing experience. Markov models and their variations have been found well suited for addressing this problem. In general, the input for these problems is the sequence of web-pages that were accessed by a user and the goal is to build Markov models that can be used to model and predict the web-page that the user will most likely access next. This study will help to explore and understand human behavior within an internet environment [4].

2 The Model

Suppose we have a dataset $\mathbf{X} := \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ of n records of web areas (classified into m different areas or *vroots*), which users visited during one session: $\mathbf{x}_j := \{x_{ij}, i = 1, \ldots, m\}$ where $x_{ij} = 1$ if j-user visited area i, alternatively, $x_{ij} = 0$.

Assuming that $\mathbf{x}_j$ is a vector-column, we form the *vlink* matrix

$$\mathcal{S} = \sum_{j=1}^{n} \mathbf{x}_j \cdot \mathbf{x}_j^T = \{s_{ik}, i, k = 1, \ldots, m\}, \tag{1}$$

where high values of $z_{ik} = s_{ik} \left(s_{ii} \cdot s_{kk}\right)^{-0.5}, i \neq k$, indicate higher similarity between areas i and k, the value of s_{ii} may be used as a measurement of the weight of the area i, and we employ the assumption

$$s_{ii} = \max_k s_{ik} \geq 1 \quad \forall i = 1, \ldots, m. \tag{2}$$

We can make a conclusion that the $i - vroot$ was always accompanied by the $j - vroot$ if $s_{ii} = s_{ij}$. Accordingly, we will call i and $j - vroots$ as equivalent if $s_{ii} = s_{ij} = s_{jj}$. Figure 1(a) illustrates an example of the *vlink* matrix, where the first three rows/columns represent equivalent *vroots*.

We form the matrix of probabilities $P = \{p_{ik}, i, k = 1, \ldots, m\}$, where

$$p_{ik} = \begin{cases} 0 \quad \text{if} \quad i = k \quad \text{or} \quad C_i = \sum\limits_{\substack{k=1 \\ k \neq i}}^{m} s_{ik} = 0; \\ \qquad\qquad\qquad\qquad \dfrac{s_{ik}}{C_i}, \quad \text{otherwise.} \end{cases}$$

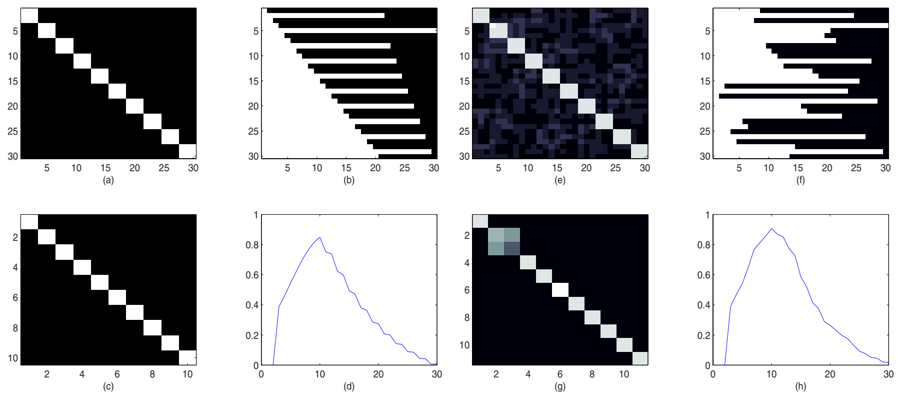


Fig. 1. *vlink* matrices (a) $\mathcal{S}_0$ and (e) $\mathcal{S}_\star$; (c) *vlink* matrix, which corresponds to the peak of d); (g) *vlink* matrix, which corresponds to the 19th step -one step before the peak of (h) (note that dark area of the image (g) is not uniform as it may be understood); (d, h): behavior of (5) as a function of the number of clusters, the following parameters were used $m_0 = 30, \gamma = 0.2, \tau = 3, \alpha = 0.005, \beta = 0.00001, \varphi = 0.0001$. In (b) and (f) vertical axis represents *vroot*, horizontal axis represents step of the merging process: light colour was changed to dark colour when the corresponding *vroot* was absorbed by another cluster.

Remark 1. The probabilistic component p_{ik} indicates similarity between rows (or corresponding *vroots*) i and k. As a result of the setting $p_{ii} = 0$, we exclude from the definition of the following below target function (5) weights of the clusters.

We are interested to maximize information (or minimize similarity) per unit cluster independently of the cluster's weights using an average symmetrical log-likelihood divergence (5). We will use the log-likelihood function in order to measure distance between i and k web-areas,

$$d_{ik} = \sum_{\substack{v=1 \\ v \neq i,k}}^{m} \xi_{ikv}, \tag{3}$$

where

$$\xi_{ikv} = \begin{cases} -p_{iv} \cdot \log p_{kv} - p_{kv} \cdot \log p_{iv} & \text{if} \quad p_{iv}, p_{kv} \geq \alpha; \\ \beta, \quad \text{otherwise}, \end{cases} \tag{4}$$

and where $\alpha > 0$ and $\beta \geq 0$ are regulation parameters. Accordingly, the average distance will be defined as

$$\mathcal{D}(\mathcal{S}, \alpha, \beta) = A(m) \sum_{i=1}^{m-1} \sum_{k=i+1}^{m} d_{ik}, \;\; A(m) = \frac{1}{m(m-1)\log(m)}, m \geq 3. \tag{5}$$

Figures 1(b), 1(f) and 2(b) illustrate merging process: the absorbed cluster changed colour from light to dark.

Algorithm 1. Merging process.

```
1: Initial setting: $k_i = i, i = 1, \dots, m$, where $m$ is a size of the squared matrix $\mathcal{S}$ defined in (1).
2: Find preferable pair for merging maximizing
       $$\max\{\frac{s^{\gamma}_{k_i k_i} \cdot (z_{k_i k_j} + \varphi)}{s^{\tau}_{k_j k_j}}, \frac{s^{\gamma}_{k_j k_j} \cdot (z_{k_i k_j} + \varphi)}{s^{\tau}_{k_i k_i}}\}, i, j = 1, \dots, m, i \neq j \quad (6)$$
   where $z_{k_i k_j} = \frac{s_{k_i k_j}}{\sqrt{s_{k_i k_i} s_{k_j k_j}}}$; $\tau, \gamma$ and $\varphi$ are positive regulation parameters.
3: Suppose that $s_{k_i k_i} \geq s_{k_j k_j}$. Then,
       $s_{k_i k_v} := s_{k_i k_v} + s_{k_j k_v}, v = 1, \dots, m; s_{k_v k_i} := s_{k_v k_i} + s_{k_v k_j}, v = 1, \dots, m, v \neq i;$
       $k_v = k_v + 1, v = j, \dots, m.$
   In the alternative case ($s_{k_j k_j} > s_{k_i k_i}$)
       $s_{k_j k_v} := s_{k_j k_v} + s_{k_i k_v}, v = 1, \dots, m; s_{k_v k_j} := s_{k_v k_j} + s_{k_v k_i}, v = 1, \dots, m, v \neq j;$
       $k_v = k_v + 1, v = i, \dots, m.$
4: $m := m - 1$, and go to the Step 2 if $m \geq 3$.
```

Definition 1. *We denote the size of the 1) initial vlink matrix $\mathcal{S}_0$ by m_0, 2) current vlink matrix $\mathcal{S}$ by $m(\mathcal{S})$ or simply m.*

Essentially, Algorithm 1 is based on the original indices k_i which may not be sequential as a result of the merging process (in contrast to the sequential secondary index $i = 1, \dots, m$).

Remark 2. The main target of the parameter φ is to link small and isolated web-areas to other web-areas.

3 Illustration of the Main Idea Using an Ideal Synthetic Example

In order to simplify notations and without loss of generality we assume that (1) clusters have equal size, and (2) all *vroots* within any particular cluster have sequential indices.

Definition 2. *Let us denote by $Q(v, k)$ the following 2D set of indices:*

$$\begin{cases} i = v \cdot h + u, \\ j = i - u + 1, \dots, i - u + v \end{cases} \quad (7)$$

where $u = 1, \dots, v$ and $h = 0, \dots, k - 1$.

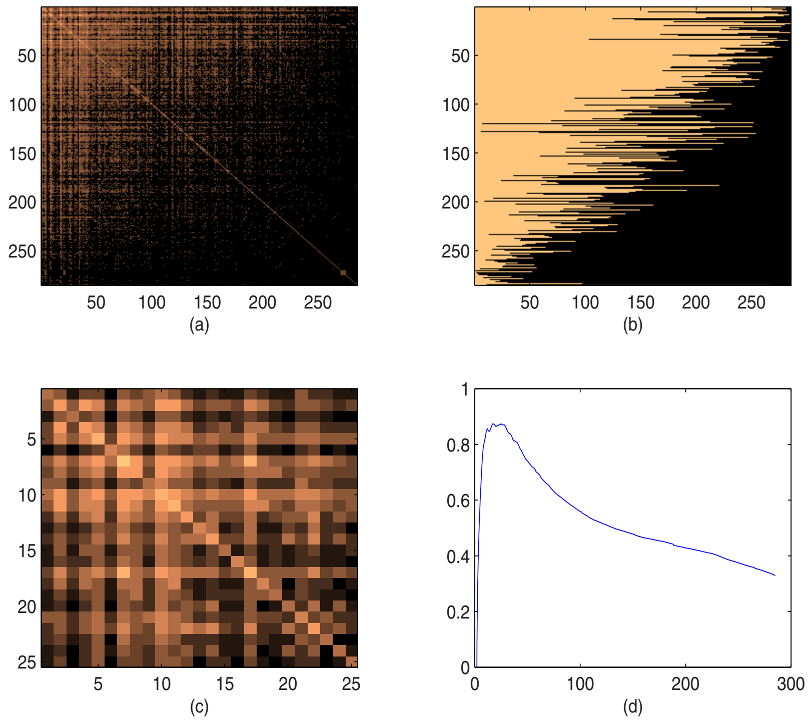


Fig. 2. (a) *vlink* matrix $\mathcal{S}_{mw}$ for *msweb* dataset; (b) merging process (see, also, Figure 1); (c) *vlink* matrix which corresponds to $m = 25$ - peak of the graph d); the following parameters were used in (d): $m_0 = 285, \gamma = 0.2, \tau = 3, \alpha = 0.005, \beta = 0.00001, \varphi = 0.0001$.

Definition 3. *We call the squared matrix G a (a, b)-diagonal if*

$$g_{ij} = \begin{cases} a & \text{if } \ i = j; \\ b, & \text{otherwise.} \end{cases}$$

We call the m-dimensional squared matrix G a $(v; a, b)$-diagonal if $m = v \cdot k$, where k is a natural number, and

$$g_{ij} = \begin{cases} a & \text{if } \ i \in Q(v, k); \\ b, & \text{otherwise.} \end{cases}$$

Proposition 1. *Suppose that $\mathcal{S}$ is a $(v; a, b)$-diagonal matrix, $v \geq 2$, and*

$$\frac{b}{(v-1)a + (m-v)b} < \alpha \leq \frac{a}{(v-1)a + (m-v)b}. \tag{8}$$

Then

$$\mathcal{D}(\mathcal{S}, \alpha, \beta) = -\frac{(v-1)(v-2)\left[\psi(Z_m) + 0.5\beta\right]}{(m-1)\log m} + \frac{(m-2)\beta}{2\log m}, \tag{9}$$

where $\psi(Z_m) = Z_m \log Z_m, \; Z_m = \frac{a}{(v-1)a+(m-v)b}$.

Proof. By definition $\mathcal{D}$ represents a sum with $0.5m(m-1)(m-2)$ terms. These terms may be split into 2 parts: (1) significant components (SC) with value $-2 \cdot \psi(Z_m)$ and (2) noise components (NC) with value β.

The size of the first group is $0.5m(v-1)(v-2)$. Similarly, the second group includes $0.5m((m-1)(m-2) - (v-1)(v-2))$ elements. □

Proposition 2. *Suppose that $\mathcal{S}$ is a (a, b)-diagonal matrix, $m \geq 2$, and*

$$\alpha \leq \frac{1}{m-1}. \tag{10}$$

Then

$$\mathcal{D}(\mathcal{S}, \alpha, \beta) = B_1(m) = \frac{(m-2)\log(m-1)}{(m-1)\log m}. \tag{11}$$

Proof. Similarly, as in the proof of the Proposition 1, $\mathcal{D}$ represents a sum with $0.5m(m-1)(m-2)$ uniform terms. The value of one particular term is $\frac{2\log(m-1)}{m-1}$. The required formula will be obtained as a product of the above two values multiplied by the norm coefficient A in (5). □

3.1 The Main Idea

Let us consider the simplified ideal case. Suppose that the *vlink* matrix may be effectively approximated by the $(v; a, b)$-diagonal matrix. Then, we can use formula (9) for divergence (5), which includes two terms (subject to the condition $v \geq 3$): 1) SC, which represents a decreasing function of m; 2) NC, which represents an increasing function of m. Assuming that the parameter β is small enough or equal to zero (means NC component is much smaller compared to the SC component) the divergence $\mathcal{D}$ will grow as a result of the sequence of merging operations. The growing process will continue until the corresponding *vlink* matrix $\mathcal{S}$ will be close to the (a, b)-diagonal shape (see Figure 1(c)). Figure 1(a) illustrates the initial *vlink* matrix with $(v; a, b)$-diagonal structure, and Figure 1(e) illustrates a matrix, which is close to the $(v; a, b)$-diagonal structure. Figure 1(g) represents a very important case just one step before the peak. After the peak, the target function $\mathcal{D}$ will decline according to Proposition 2.

3.2 Web-Traffic Simulator

Vlink matrix $\mathcal{S}_\star$ (see Figure 1(e)) for the second experiment was produced using Algorithm 2 with $T = \mathcal{S}_0$ and $E = 500$. Firstly, we simulated $n = 5000$ web-traffic records. Then, we computed $\mathcal{S}_\star$ according to (1).

Algorithm 2. Web-traffic simulator (repeats of *vroots* within any particular record are not allowed).

```
1: Order $m$ - number of web-areas; $T$ - vlink matrix (squared matrix with size $m$ and non-negative elements) and $E$ - exit weight.
2: Form vector of prior probabilities $q_i \propto T_{ii}, i = 1, \ldots, m$, and draw initial web-area $j_1$ according to $q_i$ using uniformly distributed random variable.
3: Draw second web-area $j_t, t = 2$, according to the probabilities proportional to the $j_1$ row of the matrix $T$ where $j_1$ vroot was excluded, and exit weight was added as a last element of the vector.
4: Stop the algorithm if $j_t = m - t + 2$ (exit index), alternatively, go to the next step.
5: $t := t + 1$; form vector of probabilities proportional to the minimal values of rows $j_k, k = 1, \ldots, t$, where columns $j_k, k = 1, \ldots, t$, are excluded (no repeats are allowed), and exit weight is added as a last element of the vector.
6: Draw web-area $j_t$ and go to the step 4.
```

Remark 3. The structure of the graph Figure 2(d) is remarkably similar compared with graphs Figure 1(d) and (h). However, image Figure 2(c) is much "smoother" compared to Figure 1(c).

4 Concluding Remarks

The proposed method was tested successfully against an ideal synthetic *vlink* matrix with known solution. As a next step, we considered a more complex and realistic case: we generated synthetic web-traffic data and computed the corresponding *vlink* matrix. Again, the automatic system produced the correct answer considering the inverse task.

Then, we applied the same system with identical regulation parameters to the real *msweb* dataset. As a result of the merging process the target function (5) grows initially to the point $m \approx 25$, then it declines to zero. This is in line with the main computations for the system produced transformation (merging) function. Using this function we can compress the original dataset with 285 *vroots* to a size of only 25.

References

[1] Giudici, P., Castelo, R.: Association Models for Web Mining. Data Mining and Knowledge Discovery 5, 183–196 (2001)
[2] Huber, P.: Projection Pursuit. The Annals of Statistics 13, 435–475 (1985)
[3] Nasraoui, O., Cardona, C., Rojas, C.: Using Retrieval Measures to Assess Similarity in Mining Dynamic Web Clickstreams. In: KDD 2005, Chicago, Illinois, USA (August 21-24, 2005)
[4] Cadez, I., Heckerman, D., Meek, C., Smyth, P., White, S.: Model-Based Clustering and Visualization of Navigation Patterns on a Web Site. Data Mining and Knowledge Discovery 7, 399–424 (2003)

Human Behavior Analysis for Human-Robot Interaction in Indoor Environments

Jung-Eun Park and Kyung-Whan Oh

Department of Computer Science and Engineering, Sogang University,
Mapo-Gu, Seoul 121-742, South Korea
{jepark, kwoh}@sogang.ac.kr

Abstract. Intelligent service robots should understand the needs of human beings, recognize their environment, and perform reliable tasks. To enable robots to achieve these activities, we need to understand the behavior patterns of human beings first. In recent research, there were some attempts to recognize human intentions through multimodal information systems. In this paper, human behaviors in indoor daily life were analyzed to obtain a pattern. We also suggested the use of the semantic segmentation method for the prediction of succeeding actions and for a better understanding of the robots' intentions. Finally, the validity of the suggested method was evaluated through real behavior data.

Keywords: Human's Sequential Behaviors, Semantic Segmentation, Indoor Daily Life, Human-Robot Interaction(HRI).

1 Introduction

The Human-Robot Interaction(HRI) is now becoming a more important technology to help realize intelligent robots. This technology can design and actualize the interactive environment through a variety of communication channels, as well as evaluate the environment.

We need two technologies for an effective interaction between human beings and robots. First is the interaction method, which can support various communication channels, and the second is the multimodal integration technology, which can combine information coming through diverse interactive channels. Using the integrated multimodal information, the robot could judge the situation in which it is located, understand the human intention, and perform the specific service called for by the situation. However, even if it could recognize physical situations such as the location of human beings and matter, it would still be very difficult for the robot to understand human intentions and to perform the corresponding service through multimodal information. Even though there were attempts to recognize the human intention via the dialogue interaction, this method could not achieve this goal as well.

Intelligent service robots (e.g., home robots) should understand the needs of human beings and perform the sought-after tasks in their environment. Furthermore, the behavior patterns of human beings should first be understood in order to aid in designing robots that would be helpful to human life.

In this paper, the semantic segmentation method is suggested to recognize the human intention and to predict the next step by analyzing the human behavior patterns in an indoor environment. A weighing on the N-grams method is used for the predictive model. Finally, an intention-recognition method is suggested on the basis of the above considerations. Experiment data were collected from real life models of human beings in an indoor environment. We then checked the validity of the suggested method by analyzing the moving pattern. This paper is organized as follows. Section 2 deals with the background of study and related studies. Section 3 analyzes the daily indoor life activities of human beings and explains the domain of the experiment. Section 4 presents the suggested method and the experiment results. Finally, the conclusion and the prospect of this study are described in Section 5.

2 Related Works

The HRI is based on important technical fields such as natural language processing, intention recognition, context processing, gesture recognition, and intention recognition [1]. In particular, research about intention recognition started in the 1970s and has been constantly studied until now in the field of intelligent robots.

According to the dictionary, an intention is defined as an idea or plan of what you are going to do [2]. Moreover, many research works described the character of intention as something that "… is not observed directly but all actions and purposes are related to the intention" and as "the individual differences according to the human affects building up the intention" [3][4].

The process of generation of human behavior is shown in Figure 1 [5]. Meanwhile, a goal is the desired state of affairs of a human and is the result of a sequence of actions. An intention is an idea or a mental state of what a human is going to do. After the human makes an action plan, he can achieve the goal by doing actions sequentially. Intention recognition is a reverse process of behavior generation. At first, human actions are observed. Then, a goal can be recognized through the observed actions. Through the achieved goal, we can recognize human intention under a context. In other words, the research about human behavior must be achieved for intention recognition. The analysis of human behaviors is also one of the important and challenging research topics in HRI.

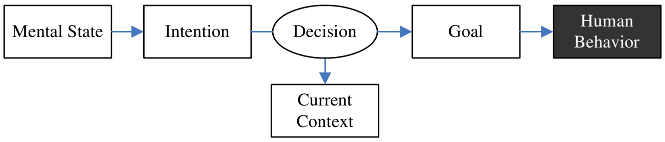


Fig. 1. The Generation of Human Behavior

Humans always behave towards accomplishing his/her own purpose or intention every day. Human behavior has been studied in many areas. It is especially based on Activities of Daily Living (ADL) and Instrumental Daily Living (IADL) tasks. The research about daily life has been studied in ergonomics and in medicine. However, these researches have been done within the viewpoint of home tasks or ADL, not for the developing manufacturers.

Meanwhile, a lot of research works are progressing steadily in the areas of computer science and engineering. Sun et al. segmented the human action sequences using the hierarchical reinforcement learning [6], and Sato et al. studied the recognition of daily human actions [7]. In addition, some studies are progressing in the areas of vision and pattern recognition. However, engineering research including the abovementioned studies have not yet analyzed the meaning of human behaviors. Most of the research focused only on segmentation based on a time-series, yet the meaning of actions is one of the most important factors in a human decision.

Therefore, in this paper, we will discuss the method which can provide a segmentation of a human's sequential behavior as their intention. The scope of behaviors will be limited indoors.

3 Human Behaviors in an Indoor Environment

Some researchers defined the daily life of human beings as follows:

"Daily life is a trifling but steady and natural story, where the pieces are connected within a schedule" [8],

"It is a most absurd, informal, and vague context which the people framed by worldly institutions try to manipulate with their own resources/wits" [9].

Human beings are living in similar but different ways with each other, especially indoors. Moreover, they move without pause to accomplish their purpose or intention in their daily lives. In this chapter, some presuppositions have been set up to match with the experimental environment by which we can analyze indoor human behaviors.

The basic presuppositions suggested for the research experiment are as follows:

- As the home service robot is not for all purposes but for a private person only, the object of this experiment is thus confined to one person.
- The object person does not leave the house (indoor environment).
- The behaviors of the object person are observed according to the sequential flow of time.
- The man-to-man communication within the experimental environment is ignored and only the sequential behavior of the object person is observed.

Table 1. Observation environment

Observation Target	Gender	:	Female
	Age	:	56
	Duration	:	April 2 ~ April 5, 2007
Observation Space	Apartment (main room, dressing room, kitchen, Living room, balcony)		

Table 1 provides a description of the observation environment. The object person was a 56-year old housewife. The experiment was carried out during the four weekdays when she hardly went out and had little communication with family members and friends.

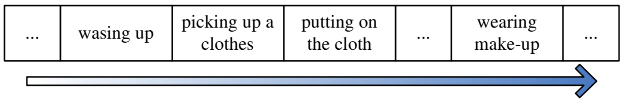


Fig. 2. Part of the sequential behaviors of the object person

Figure 2 shows the parts of the sequential behaviors of the object person.

All her behaviors, except the outgoing and man-to-man communication, were selected for analysis. In addition, except for the main room, dressing room, living room, kitchen, and balcony, the other spaces were excluded. From all the sequential behavior data, a total of 331 behaviors of the object person were filtered through the preprocessing. The items in Table 2 are parts of the data selected from daily life. They are presented on the basis of ADL and IADL indicators.

Table 2. Parts of the data selected from daily life

stretch out for the refrigerator	turn on the water	wash the food materials
cut the food materials	put out the pan/pot	put it on the gas stove
turn on the gas stove	put the food on the dish	carry the dish to the table
put the dish on the table	open the drawer	put out the clothes
close the drawer	pull up or down the clothes	zip up or button up
insert the plug of the vacuum cleaner into the outlet		etc.

4 Semantic Segmentation of Human Behaviors

4.1 Prediction Segmentation of Human Behaviors

The selected data were presented as follows. Each action of the object person was replaced by the corresponding number.

$$1 \rightarrow 2 \equiv \{1,2\} \tag{1}$$

It means that action 1 is followed by action 2. In the above case, one previous path was chosen as the criteria of prediction. This means that it is a one-hop request number and it is called the 1-gram method [10]. The behavior profile of the object person was composed of the patterns extracted from the analysis of one to six hops for the sequential behaviors of the object person that satisfy the above conditions. Table 3 is a part of the profile.

On the basis of the above experimental data, another observation of the object person's daily life was made on the 14th and 15th of May 2007. The result was compared with the previous one to check the predictive performance of the method. As an evaluation criterion, the well-known formula of James Pikow, *HitRatio=Pr(Hit)/Pr(Miss)*, was used. The above Figure 3 is the Hit Ratio graph for

Table 3. Part of the profile

generalized behavior patterns	predicted next requests
002 003	001
…	…
016 017 024	121
214 263 264	002
…	…

the change of N values. As a result of the experiment, the prediction performance of 3-grams was found to be the best.

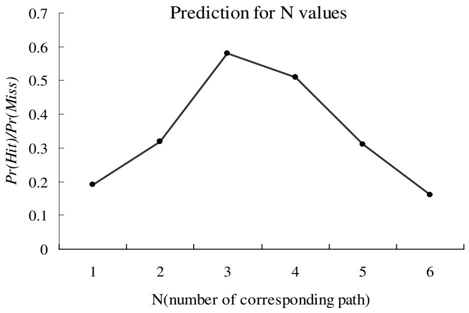


Fig. 3. Hit Ratio graph for the change of N values

4.2 Prediction of Human Behaviors by Semantic N-Grams Method

As mentioned above, the prediction performance of 3-grams was found to be the best. However, the extraction of data by this method required a lot of patterns. Therefore, the semantic N-grams method is suggested to reduce the number of moving patterns by giving weight to each pattern. The criteria for the semantic N-grams are as follows. According to these criteria, the behavior patterns of object person are then extracted.

- The sequential behavior in the experiment data is analyzed into one-hop actions. Divide the total request number with the number of moving patterns to find the average request number for one pattern. Only the moving patterns with an average or higher number are considered meaningful.
- Different sequential patterns can combine with each other. For example, action 001 and the action 002 are combined when their movement number is more than the 1.5 times weighed value of the previous pattern request number.
- The pattern to return to the previous action is excluded even though the request number satisfies the above condition.

As a result, the Hit Ratio increased to 0.75, up by 0.17, while the number of patterns was reduced by 36%.

4.3 Semantic Segmentation for Intention Recognition

Human beings move without pause to accomplish their purpose and intention in their daily lives. The object person in the above experiment also moved sequentially for her own purposes. On the basis of the experiment results, we try to determine the intention of the object person in her sequential behavior. For a pattern that was not predicted in 4.1 and 4.2, we placed a split bar between it and the previous pattern and then checked if they coincided with some of the sample data prepared in Table 4.

Table 4. Intention and sequential behaviors

Intention	Sequential Behaviors
phone calls	pick up the phone, push the phone buttons, talk, hang up the phone
take medicine	take out the water, pour the water in the cup, take out the pills
laundry	carry the clothes to the washing machine, put them into the washing machine, put the detergent into the machine, push the button
drying clothes	go to the washing machine, open the washing machine, hold up the washed clothes, hang out the wash
watching TV	pick up the remote controller, push the button on the remote, watch the TV
etc.	

a	Carry the clothes to washing machine.
b	Put them into the machine.
c	Put the detergent in it.
d	Push the button.
e	Go to the refrigerator.
f	Take out the water.
g	Pour the water in the cup.
h	Take out the pills.
i	Move to the living room.
j	Sit on the sofa.
...	etc.

Fig. 4. Sequential behavior scripts after placing a split bar

Figure 4 is the script that shows the real sequential behavior of the indoor object person. As shown by the figure, the e action pattern after the d action was not predicted. It means that the intention of the action was different. As compared with the intentions in Table 3, a~d actions were the laundry behavior, e~h were the taking medicine behavior. This kind of failure shows that the previous and subsequent actions have different intentions from each other.

5 Conclusion and Future Works

Many studies are currently being conducted to recognize human intentions for the development of intelligent service robots that can understand human needs and perform reliable tasks. To recognize human intentions, we need to understand the behavior patterns of human beings first.

In this paper, the semantic segmentation method was suggested in order to predict the next action and recognize the intention by recording and analyzing daily life behaviors in a real indoor environment. Through this, the service robots may be able to predict the next action of human beings and recognize their intentions so that they can provide more satisfactory services.

In this research, the daily life behaviors of only one person were considered. Hence, it will be significant for future studies to check the possibility of a service collision when there are two or more persons in the house. Moreover, if the user modeling method to assist in the above research is invented, it will be a good basis for the development of more advanced intelligent service robots in terms of their accuracy and performance.

Acknowledgments. This research paper was performed for the Intelligent Robotics Development Program, one of the 21st Century Frontier R&D Programs funded by the Ministry of Commerce, Industry and Energy of Korea.

References

1. Iba, S., Paredis, C.J.J., Khosla, P.K.: Interactive Multimodal Robot Programming. The International Journal of Robotics Research 24(1), 83–104 (2005)
2. Collins Cobuild English Dictionary for Advanced Learners (2001)
3. Geddes, N.D.: Understanding human operators' intentions in complex systems. Doctoral Thesis, Georgia Institute of Technology (1990)
4. Geddes, N.D.: The Use of Individual Differences in Inferring Human Operator Intentions. In: 2nd Aerospace Applications of Artificial Intelligence Annual Conference, pp. 30–41. AAAI Press, USA (1986)
5. Youn, S.J., Oh, K.W., des, N.D.: Intention Recognition using a Graph Representation. In: CISE, vol. 19, pp. 13–19 (2007)
6. Sun, R., Sessions, C.: Self-segmentation of sequences: automatic formation of hierarchies of sequential behaviors. IEEE Transactions on Systems, Man and Cybernetics, Part B 30(3), 403–418 (2000)
7. Mori, T., Shimosaka, M., Sato, T.: SVM-Based Human Action Recognition and Its Remarkable Motion Features Discovery Algorithm. In: Marcelo, H., Khatib Jr., O.A. (eds.) Experimental Robotics IX, pp. 15–25. Springer, Heidelberg (2006)
8. Lefevre, H.: The Grid: La Vie Quotidienne dans le Monde Moderne (Everyday Life in the Modern World). Gallimard (1968)
9. Bobone, L.: Theories of Everyday Life: A Search for Meaning or a Negation of Meaning? Current Sociology 37(1), 41–59 (1989)
10. Comlekoglu, F.M.: Optimizing a Text Retrieval System Utilizing Ngram Indexing. Ph.D. Thesis, The George Washington University (1990)

Fitness Functions in Genetic Programming for Classification with Unbalanced Data

Grant Patterson and Mengjie Zhang

School of Mathematics, Statistics and Computer Science
Victoria University of Wellington, P.O. Box 600, Wellington, New Zealand
{grant,mengjie}@mcs.vuw.ac.nz

Abstract. This paper describes a genetic programming (GP) approach to binary classification with class imbalance problems. This approach is examined on two benchmark and two synthetic data sets. The results show that when using the overall classification accuracy as the fitness function, the GP system is strongly biased toward the majority class. Two new fitness functions are developed to deal with the class imbalance problem. The experimental results show that both of them substantially improve the performance for the minority class, and the performance for the majority and minority classes is much more balanced.

1 Introduction

Recently, the class imbalance problem in classification has attracted continuous attention from the machine learning and pattern recognition communities [1,2,3,4]. In this problem, one class has only a very small number of examples (called the *minority class*), while the other class has a large number of cases (called *majority class*). When using the overall classification accuracy or error rate on the entire data set as the performance measure, the learning systems often achieved "good-looking results". However, when closely checking the performance on each class, the result of the minority is actually quite bad particularly for the relatively difficult problems. In many class imbalance problems, the minority class is the "main class" and correctly classifying the examples for this class is very important. Medical diagnosis tasks and object detection tasks are typical examples of this kind. Many efforts have been made toward solving this problem in the machine learning and pattern community [2,3,4].

This paper is focused on the fitness functions in GP for the class imbalance problem. Genetic programming (GP) [5,6] is a recent evolutionary learning technique for building reliable classification programs automatically from a set of example data. GP uses ideas analogous to biological evolution to search for good programs for a particular task. For classification problems, GP usually uses classification accuracy or error rate over the training set as the fitness function and has been applied to a range of classification tasks with some success [6,7,8,9].

This paper aims to investigate the effect of different fitness functions in GP for classification problems with unbalanced data classes. We first investigate whether

GP with the overall classification accuracy on the entire training set as the fitness function has similar or different effect to the classification performance on the minority class from other machine learning approaches reported in the literature. Based on the analysis of the results, two new fitness functions are designed and developed to investigate whether and how these new fitness functions improve the system performance. To achieve these goals, the GP approach with different fitness functions will be examined and compared on two standard data sets and two synthetic data sets.

The reset of the paper is organised as follows. Section 2 describes the four data sets used in the experiments. Section 3 describes the performance of the overall classification accuracy as the fitness function. Section 4 describes new fitness functions in GP for classification with unbalanced classes and reports the new results. Section 5 concludes the paper and gives future work directions.

2 Classification Data Sets

The experiments used two standard data sets and two synthetic data sets. The two standard data sets, *SPECT* and *PIMA*, were chosen from the *UCI Repository of machine learning databases* [10].

SPECT heart data. This dataset contains 267 data instances (patients) derived from cardiac Single Proton Emission Computed Tomography (SPECT) images. This is a binary classification task, where patients heart images are classified as *normal* or *abnormal*. The class distribution has 55 instances of the "abnormal" class (20.6%) and 212 instances of the "normal" class (79.4%), a class ratio of approximately 1:4. Each SPECT image was processed to extract 44 continuous features, which were further pre-processed to 22 binary features.

Pima Indian diabetes data. This data set comes from a study of diabetes among a Pima Indian population in Arizona, USA. The data was collected by the National Institute of Diabetes and Digestive and Kidney Diseases. All patients in the data set are females of at least 21 years of age and of Pima Indian heritage. Patterns consist of 8 numerical attributes, each obtained directly from the patient. Each pattern is classified as either *normal* or *abnormal*. The data set contains 768 instances, and has a class imbalance ratio of only approximately 1:2, which is more balanced than the SPECT data set. However, there are a large number of missing values in a number of attributes, where a zero is filled. The large number of instances with missing attributes makes this data set difficult to classify well.

The two synthetic data sets were generated using a computer program. For visualisation purpose, only two attributes were used in the two data sets, as shown in figure 1. The two class cases (○ for the minority class, and + for the majority class) were primarily put into different regions but some noise was added to provide classification problems of increasing difficulty: 5% and 20% of noise was added for the Synthetic1 and Synthetic2 data sets, respectively.

For all these data sets, one half was used as the training set for learning a program classifier, and the other half was used as the test set for measuring the

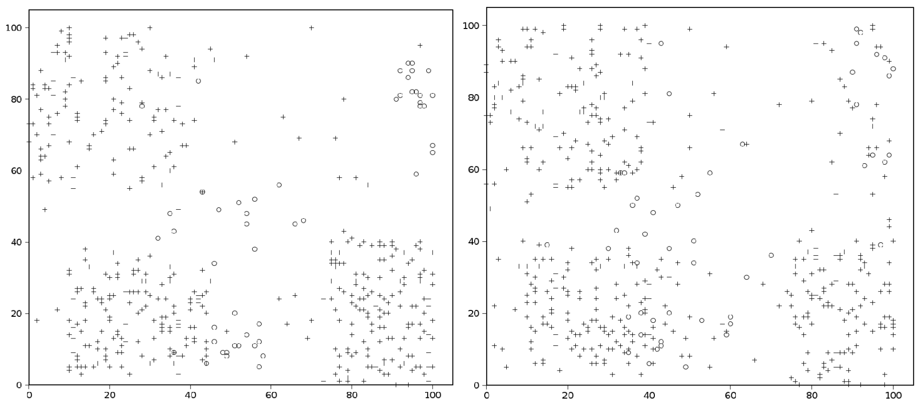


Fig. 1. Two synthetic data sets generated by a computer program. Left: Synthetic1; Right: Synthetic2.

classification performance of the evolved program classifiers. The training set and the test set were uniformly randomly chosen from these data sets so that they have a similar number of examples for each class.

3 Classification Accuracy as Fitness Function

This section examines whether GP using the classification accuracy of the entire training set as the fitness function can successfully classify the examples on both the majority class and minority class. We used the tree-based structure to represent genetic programs [5]. The ramped half-and-half method was used for generating programs in the initial population and for the mutation operator [6]. The proportional selection mechanism and the reproduction [7], crossover and mutation operators [11] were used in the evolutionary process.

3.1 Terminals and Functions

For the SPECT data set, the 22 attributes were used as terminals. For the PIMA data set, the eight attributes form the terminal set. For the two synthetic classification tasks, the feature terminal set only contains the two attributes (X and Y). In all the four data sets, we also used a constant terminal.

In the function set, the four standard arithmetic operators and a conditional operator were used: $\{+, -, \times, \div, if\}$. The $+$, $-$, and $\times$ operators have their usual meanings — addition, subtraction and multiplication, while $\div$ represents "protected" division which is the usual division operator except that a divide by zero gives a result of zero. Each of these functions takes two arguments. The *if* function takes three arguments. The first argument, which can be any expression, constitutes the condition. If the first argument is negative, the *if* function returns its second argument; otherwise, it returns its third argument.

3.2 Fitness Function — Classification Accuracy

The output of a genetic program in the GP system is a floating point number. For classification problems, this single output value needs to be translated into a set of class labels. As all the four classification tasks only have two classes, a majority class and a minority class, we used the division between negative and non-negative numbers of a genetic program output as the separation of the two classes in these tasks.

In this investigation, classification accuracy on the entire training set was used as the fitness function. Assuming that the numbers of the examples for the majority and the minority classes are N_{maj} and N_{min} and that the numbers of examples a genetic program p correctly classifies are N_{maj-p} and N_{min-p}, then the classification accuracy of the genetic program classifier will be

$$Accuracy_p = \frac{N_{maj-p} + N_{min-p}}{N_{maj} + N_{min}} \times 100\% \tag{1}$$

Clearly, this fitness function does not consider the fact that the number of examples for the minority classes is much smaller than that for the majority classes. We will examine the effect of this fitness function in the GP approach.

3.3 Parameter Values and Experiment Results

In the GP system, the population size is 500, the rates used for crossover, mutation and reproduction are 60%, 30% and 10%, respectively, and the maximum program depth is 6. The evolution will run 50 generations unless a "good enough" solution program is found, in which case the evolution is terminated early. All experiments were repeated for 50 runs with different random seeds and the means and standard deviations on the test sets are shown in table 1.

According to table 1, if we only look at the classification accuracy on all fitness cases (examples) in the test set, it seems that the GP system achieved very good results (more than 80%) except for the PIMA data set. This is particularly true for the two synthetic data sets, where the overall accuracies on the tests are about 90%. If we check the classification accuracy on the minority class, however, none of the results in any of the four data sets is good. On the second synthetic data sets in particular, the accuracy on the minority class is very close to zero. This is because almost all the examples were classified as the majority class. These results show that using overall classification accuracy as fitness function, the GP

Table 1. Results (mean ± standard deviation) of the GP approach using the standard fitness function on the four unbalanced data sets

Data set	overall (%)	majority class(%)	minority class (%)
SPECT	82.2 ± 6.7	91.3 ± 5.4	47.2 ± 25.9
PIMA	68.6 ± 5.5	85.5 ± 7.3	37.1 ± 13.3
Synthetic1	91.4 ± 2.6	99.0 ± 2.0	23.2 ± 23.2
Synthetic2	89.8 ± 0.8	99.7 ± 1.0	1.2 ± 4.7

system will be heavily biased toward the majority class. For the data sets with highly unbalance classes, such a GP system has the risk of evolving only a trivial program classifier which classifies all the examples of the entire data set as the majority class.

4 New Fitness Functions for Class Imbalance Problem

4.1 New Fitness Functions

To avoid the above problem, we developed two new fitness functions in the GP system to improve the system performance on the minority classes.

The first new fitness function aims to differentiate between the classification performance of both the majority and the minority classes. The idea is that poor performance in either class will lead to a poor overall fitness, with the goal of directing the evolutionary process towards a solution that performs well in classifying both classes. The fitness function is shown in equation 2, where p, N_{maj}, N_{min}, N_{maj-p}, and N_{min-p} are specified earlier.

$$NewFit1_p = \frac{\frac{N_{maj-p}}{N_{maj}} + \frac{N_{min-p}}{N_{min}}}{2} \times 100\% \tag{2}$$

In equation 2, the first component $\frac{N_{maj-p}}{N_{maj}}$ is the accuracy of program p on the majority class, and the second component $\frac{N_{min-p}}{N_{min}}$ is the accuracy of program p on the minority class, and the overall fitness measure of program p is a linear combination of them. The main consideration of division by 2 in the equation is to make the fitness measure "standardised", so that the range of the fitness measure is between [0, 1].

In the second new fitness function, while still considering to balance the performance of individual classes, we use the square of the accuracy of the individual classes to put more penalty on the poorer performance by a particular class than the first fitness function, as shown in equation 3.

$$NewFit2_p = \frac{(\frac{N_{maj-p}}{N_{maj}})^2 + (\frac{N_{min-p}}{N_{min}})^2}{2} \times 100\% \tag{3}$$

4.2 Results and Discussions

In this set of experiments, we used the same parameter values and termination criteria as the previous set on the standard accuracy as the fitness function. The average results with standard deviations over 50 independent runs are shown in table 2. For each data set, the results using the standard accuracy as fitness function are repeated for comparison purpose, followed by the results using the two new fitness functions.

According to table 2, the GP systems with both new fitness functions significantly improved the classification performances for the minority class (abnormal), and the performance on the two classes are generally much more balanced than that using the standard accuracy as fitness function. This suggests that,

Table 2. Results of the GP approach using the three new fitness functions on the four unbalanced data sets

Data set	Fitness Function	overall (%)	majority (%)	minority (%)
SPECT	Accuracy	82.2 ± 6.7	91.3 ± 5.4	47.2 ± 25.9
	New Fit 1	74.2 ± 9.1	75.3 ± 8.7	70.4 ± 21.3
	New Fit 2	72.8 ± 9.4	72.7 ± 9.7	73.3 ± 21.5
PIMA	Accuracy	68.6 ± 5.5	85.5 ± 7.3	37.1 ± 13.3
	New Fit 1	65.9 ± 6.7	64.3 ± 9.5	69.0 ± 10.6
	New Fit 2	46.1 ± 11.6	23.3 ± 26.9	88.8 ± 22.1
Synthetic1	Accuracy	91.4 ± 2.6	99.0 ± 2.0	23.2 ± 23.2
	New Fit 1	70.4 ± 12.4	68.7 ± 14.1	86.6 ± 16.6
	New Fit 2	63.2 ± 11.3	59.9 ± 13.1	92.7 ± 13.7
Synthetic2	Accuracy	89.8 ± 0.8	99.7 ± 1.0	1.2 ± 4.7
	New Fit 1	63.9 ± 10.4	62.5 ± 12.4	76.4 ± 24.4
	New Fit 2	54.7 ± 9.7	50.9 ± 11.0	88.5 ± 16.7

carefully designing and constructing fitness functions in GP can lead to more balanced results for all the classes and the classification performance on the minority class can be substantially improved.

In terms of the two new fitness functions, the degrees of improvements over the standard classification accuracy are different. However, these improvements on all the four data sets show a similar pattern, that is, on the minority classes (abnormal), the amount of the improvement achieved with the second new fitness function is larger than the first new fitness function. This is consistent with the nature of the new fitness functions: the square individual class accuracy has the larger penalty to very poor results than the simple linear weighted individual class accuracy. Depending on the goal of a particular classification task with the class imbalance problem, one can choose a fitness function appropriate for that task. For example, if finding all examples in the minority class is more important for a particular task, then the second new fitness function can be selected.

5 Conclusions

The goal of this paper was to investigate the problems of GP for classification tasks with unbalanced classes and how these problems can be solved in terms of the fitness function. The goal was successfully achieved by examining GP with the standard classification accuracy as the fitness function on four unbalanced data sets and by constructing two new fitness functions to improve the classification performance. The experimental results showed that when using the overall classification accuracy of the entire training set as the fitness function, the GP system could not correctly classify the fitness cases of the minority class and was strongly biased toward the majority class. Both two new fitness functions substantially improved the performance for the minority class, and the performance for the majority and minority classes was much more balanced. The fitness function based on the combination of the square of the individual class accuracy achieved good performance for the minority class.

This work only examined the binary classification tasks with class imbalance problems. We will investigate this approach on the multiclass classification problems in the future. We will also investigate methods such as the ROC curve measures for improving the performance of the minority class as well as the overall performance of the classification tasks with class imbalance problems.

Acknowledgement

We thank Dr Mark Johnston for his useful comments. This work was supported in part by the Marsden Fund council from the government funding (05-VUW-017), administrated by the Royal Society of New Zealand, and the University Research Fund (URF 7/39) at Victoria University of Wellington.

References

1. Japkowicz, N., Stephen, S.: The class imbalance problem: A systematic study. Intell. Data Anal. 6(5), 429–449 (2002)
2. Weiss, G.M., Provost, F.: The effect of class distribution on classifier learning: An empirical study (August 12, 2001)
3. Orriols, A., Bernadó-Mansilla, E.: The class imbalance problem in learning classifier systems:A preliminary study. In: Rothlauf, et al. (eds.) Genetic and Evolutionary Computation Conference (GECCO2005) workshop program, June 25-29, 2005, pp. 74–78. ACM Press, Washington, D.C., USA (2005)
4. Flach, P.A.: The geometry of ROC space: Understanding machine learning metrics through ROC isometrics. In: Fawcett, T., Mishra, N. (eds.) ICML, pp. 194–201. AAAI Press, USA (2003)
5. Koza, J.R.: Genetic programming: on the programming of computers by means of natural selection. MIT Press, Cambridge, Mass (1992)
6. 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 Publishers, San Francisco, Calif. (1998)
7. Zhang, M., Ciesielski, V., Andreae, P.: A domain independent window-approach to multiclass object detection using genetic programming. EURASIP Journal on Signal Processing, Special Issue on Genetic and Evolutionary Computation for Signal Processing and Image Analysis 2003(8), 841–859 (2003)
8. Muni, D.P., Pal, N.R., Das, J.: A novel approach to design classifier using genetic programming. IEEE Transactions on Evolutionary Computation 8(2), 183–196 (2004)
9. Krawiec, K., Bhanu, B.: Visual learning by coevolutionary feature synthesis. IEEE Transactions on System, Man, and Cybernetics – Part B 35(3), 409–425 (2005)
10. Newman, D., Hettich, S., Blake, C., Merz, C.: Uci repository of machine learning databases (1998)
11. Koza, J.R.: Genetic Programming II: Automatic Discovery of Reusable Programs. MIT Press, Cambridge, Mass (1994)
12. Bradley, A.P.: The use of the area under the ROC curve in the evaluation of machine learning algorithms. Pattern Recognition 30(7), 1145–1159 (1997)

Protein Structure Alignment Using Maximum Cliques and Local Search

Wayne Pullan

School of Information and Communication Technology,
Griffith University, Gold Coast, QLD, 4215, Australia
w.pullan@griffith.edu.au

Abstract. The protein structure alignment problem addresses the question of measuring the degree of similarity, in three–dimensional structure, of two proteins. The representation of each protein using a simple contact map allows the correspondence graph for the protein pair to be generated and the maximum clique within this graph provides a measure of the structural similarity between the two proteins. This study uses a recently developed maximum clique algorithm, Phased Local Search (PLS), to locate the maximum cliques within correspondence graphs.

1 Introduction

In bioinformatics, structural comparison of proteins is useful in several domains. For example, as protein function is intrinsically tied to a protein's structure [1], identifying structural similarity between a protein and other proteins whose function is known can allow the prediction of that protein's function. Over the last decade, a number of techniques for structurally comparing proteins have been developed however, none have proved adequate across a range of applications. A relatively new technique, Contact Map Overlap (CMO), first proposed in [2] (and subsequently shown to be NP-complete [3]), is to identify alignments between protein contact maps with the goal of maximising the number of consistent alignments. A protein consists of a chain of residues (amino acids). When a protein folds into its tertiary (lowest energy) structure, residues that are not directly adjacent in the chain may be physically close in space. The contact map for a protein is a simple representation of this three–dimensional structure and is the matrix of all pairwise distances between the components of the protein. For this study, the components of the protein are identified as the alpha carbon atoms (C_α) of each amino acid. The contact map can be further simplified into a 0–1 contact map by encoding each pairwise distance as one if the pairwise distance is less than some distance threshold (typically in the range 4 - 8 Å and 5.5 Å in this study) and zero otherwise. As shown in [4], the CMO problem can be directly translated to a maximum clique (MC) problem which calls for finding the maximum sized sub-graph of pairwise adjacent vertices in a given graph. Formally, the MC problem can be stated as: Given an undirected graph $G = (V, E)$, where V is the set of all vertices and E the set of all edges, find a maximum size clique in G, where a clique K in G is a subset of vertices, $K \subseteq V$, such that all pairs of vertices in K are connected by an edge, *i.e.*, for all $v, v' \in K$, $\{v, v'\} \in E$, and the size of the clique K is the cardinality $|K|$ of K. The maximum clique problem is to maximise $|K|$, the cardinality of K. MC is NP-hard and the associated decision problem is NP-complete [5]. Therefore, large and hard instances of MC are typically solved using heuristic approaches of which the most recent is Phased Local Search (PLS) [6], a reactive algorithm that interleaves sub-algorithms which alternate between sequences of iterative improvement and plateau search. The differences between these sub-algorithms are primarily in the vertex selection method and the perturbation mechanisms used to overcome search stagnation. Extensive computational experiments [6] have shown that PLS has equivalent, or improved, performance compared to other state-of-the-art MC search algorithms, on a range of widely studied benchmark instances.

2 The PLS Algorithm

PLS [6] is now described using the following additional notation:
$N(i) = \{j \in V : \{i, j\} \in E\}$ — the vertices adjacent to i; K — current clique of G; and, $C_p(K) = \{i \in V : |K \setminus N(i)| = p\}$, $p = 0, 1$ — the set of all vertices not adjacent to exactly p vertices in K.

Algorithm PLS (G, *tcs*, *max-selections*)

```
        Input:   graph $G$; integers tcs (target clique size), max-selections
        Output: $K$ of size tcs or 'failed'
 1.     $selections := 0, pu := 0, pd := 2$;
 2.     $sa := Random, iterations := 50$;
 3.     <Randomly select a vertex $v \in V$, $K := \{v\}$ >;
 4.     $\forall i \in V, p_i := 0$;
 5.     do
 6.         do
 7.             while $C_0(K) \setminus U \neq \emptyset$ do
 8.                 $v := Select(C_0(K), sa)$;
 9.                 $K := K \cup \{v\}$;
10.                 $selections := selections + 1$;
11.                 if $|K| = tcs$ then return $K$;
12.                 $U := \emptyset$;
13.             end while
14.             if $C_1(K) \setminus U \neq \emptyset$ then
15.                 $v := Select(C_1(K) \setminus U, sa)$;
16.                 $K := [K \cup \{v\}] \setminus \{i\}, U := U \cup \{i\}$, where $\{i\} = K \setminus N(v)$;
17.                 $selections := selections + 1$;
18.             end if;
19.         while $C_0(K) \neq \emptyset$ or $C_1(K) \setminus U \neq \emptyset$;
20.         $iterations := iterations - 1$;
21.         $UpdatePenalties(sa)$;
22.         $Perturb(sa)$;
23.     until $selections \geq max\text{-}selections$
24.     return 'failed';
```

Table 1. PLS performance results, averaged over 100 independent trials, for the benchmark instances from [4]. The maximum known clique size, for each instance, is shown in the 'Max. Clique' column; CPU(s) is the PLS run-time in CPU seconds, averaged over all successful trials, for each instance. 'SCPU(s)' is the CPU time reported in [4], scaled by 0.08 to allow some basis for comparison with the reference computer used in this study.

Problem Instance	G Vertices	G Edges	Success Rate	Max. Clique	PLS CPU(s)	SCPU(s)
1BPI-1KNT	2 279	385 009	100	31	0.0469	1.52
1BPI-2KNT	2 436	446 657	100	29	0.0574	14.56
1BPI-5PTI	3 016	698 195	100	42	0.0299	2.4
1KNT-1BPI	2 494	462 092	100	30	0.0959	8.8
1KNT-2KNT	1 806	240 521	100	39	0.0098	0
1KNT-5PTI	2 236	378 609	100	28	0.0353	3.68
1VII-1CPH	171	1 581	100	6	0.0001	0
2KNT-5PTI	2 184	364 315	100	28	0.0267	7.6
3EBX-1ERA	2 548	477 720	100	31	0.1257	18.88
3EBX-6EBX	1 768	225 761	100	28	0.0163	0.48
6EBX-1ERA	1 666	199 074	100	20	0.0169	8.08

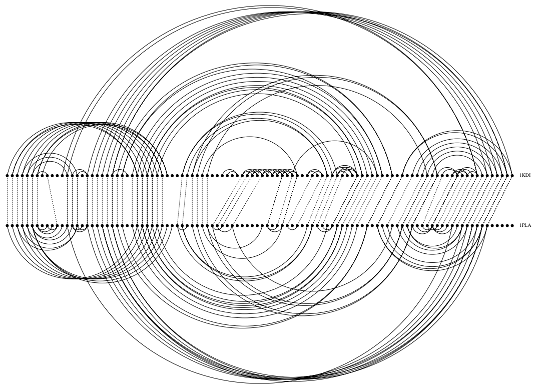


Fig. 1. Undirected graph representation of the 0–1 contact maps and putative maximal alignments for the 1KDI and 1PLA proteins. The vertices (dots) represent the residues of each protein, the solid edges (arcs) the contacts within each protein and the dashed edges show alignments identified by finding the maximum clique in the correspondence graph

PLS uses three sub-algorithms within the *Select* function which are effective for three different instance types. The first sub-algorithm, *Random*, effectively solves instances where the maximal clique consists of vertices with a wide range of vertex degrees. The second sub-algorithm, *Penalty*, uses vertex penalties to bias the search towards cliques containing lower degree vertices. The vertex penalties are increased when the vertex is in the current clique when a perturbation occurs and are subject to occasional decrease, which effectively allows the sub-algorithm to 'forget' vertex penalties over time. PLS adaptively modifies the frequency of penalty decreases to obtain near optimal performance. The third PLS sub-algorithm, *Degree*, uses vertex degrees to bias the search towards cliques containing higher degree vertices.

Table 2. PLS performance results, averaged over 100 independent trials, for the PDB protein pairs in Benchmark-2. 'G' is the correspondence graph for each protein pair, the sizes found for each maximum clique are shown as maximum, average and minimum found over the 100 trials while 'Sels.' is the average number of vertices that were added to the clique over the 100 trials.

Problem	G	G	Success		Clique				Sels. /
Instance	Vertices	Edges	Rate	Max.	Avg.	Min.	CPU(s)	Sels.	Sec.
1A8O–1F22	2 728	1 063 344	100	25	25	25	30.52	1 412 602	46 278
1AVY–1BCT	6 278	6 842 400	99	50	49.99	49	630.30	14 726 972	23 365
1B6W–1BW5	4 131	3 095 143	100	34	34	34	84.09	2 977 231	35 407
1BAW–2B3I	7 200	5 519 222	100	53	53	53	28.36	488 140	17 215
1BCT–1BW5	4 386	3 277 240	77	36	35.77	35	1 117.91	36 305 118	32 476
1BCT–1F22	3 784	2 101 393	100	25	25	25	20.38	695 072	34 111
1BCT–1ILP	4 988	3 866 071	100	30	30	30	0.36	10 465	28 941
1BPI–2KNT	1 848	362 922	100	32	32	32	0.28	18 291	64 976
1C7V–1C7W	2 401	886 821	100	34	34	34	17.70	971 361	54 871
1C9O–1KDF	2 805	729 697	100	21	21	21	0.45	18 950	41 775
1DF5–1F22	3 960	2 462 200	100	27	27	27	46.27	1 513 993	32 718
1KDI–1BAW	7 920	6 774 365	100	53	53	53	610.03	9 700 460	15 901
1KDI–1PLA	6 424	4 392 217	100	53	53	53	369.05	7 053 785	19 113
1KDI–2B3I	7 040	5 246 576	100	47	47	47	679.82	11 884 684	17 482
1KDI–2PCY	7 216	5 583 059	98	57	56.98	56	1543.28	26 656 690	17 272
1NMF–2NEW	2 728	800 896	100	21	21	21	55.00	2 272 339	41 313
1NMG–1WDC	4 698	2 754 536	100	17	17	17	0.98	26 320	26 928
1PFN–1SVF	5 992	5 197 600	100	30	30	30	83.02	1 922 495	23 156
1PLA–1BAW	6 570	4 512 505	100	55	55	55	368.79	6 860 641	18 603
1PLA–2B3I	5 840	3 510 020	100	47	47	47	80.16	1 646 885	20 544
1PLA–2PCY	5 986	3 725 919	100	57	57	57	218.30	4 381 399	20 070
1VII–1CPH	903	99 638	100	15	15	15	0.024	3 121	128 429
1VNB–1BHB	6 120	4 011 048	100	28	28	28	138.98	2 786 011	20 046
2KNT–1KNT	1 980	402 659	100	41	41	41	0.06	3 800	64 083
2NEW–3MEF	2 552	631 920	100	16	16	16	0.14	6 248	43 357
2PCY–1BAW	7 380	5 769 409	100	66	66	66	535.34	8 920 572	16 663
2PCY–2B3I	6 560	4 475 832	100	52	52	52	136.20	2 524 044	18 532
3EBX–1ERA	2 205	356 245	100	19	19	19	0.04	2 279	51 444
3EBX–6EBX	2 331	461 771	100	25	25	25	0.51	24 855	48 735
5PTI–1BPI	1 596	285 692	100	35	35	35	0.17	12 602	75 733
5PTI–1KNT	1 710	303 273	100	31	31	31	0.063	4 505	71 508
5PTI–2KNT	1 672	290 649	100	32	32	32	0.63	44 616	70 339
6EBX–1ERA	1 295	168 119	100	22	22	22	0.02	1 986	92 373

3 Empirical Performance Results

For this study, two protein structure alignment benchmarks were utilised to evaluate the performance of PLS on this type of problem. Benchmark–1 was used in [4] (the correspondence graphs for this benchmark were obtained directly from the authors of this paper) and consists of 10 different protein structure alignment problems. The proteins in this benchmark all contain approximately 50 residues and the correspondence graphs have up to 3 000 vertices and 700 000 edges.

Benchmark–2 was constructed using proteins from the Protein Data Bank [7]. The Universality Similarity Measure (USM) software[1] [8] (with a 5.5 Å threshold) was used to generate the contact maps for these proteins. From the contact maps for the proteins to be compared, the two-dimensional grid G was generated and the correspondence graph created by adding an edge when the two alignments represented by pairs of vertices are a feasible solution to the CMO problem. The proteins in this benchmark range in size from

[1] http://www.cs.nott.ac.uk/ nxk/USM/protocol.html

60 to 100 residues and the correspondence graphs have up to approximately 8 000 vertices and 9 000 000 edges. All experiments for this study were performed on a dedicated computer that, for the DIMACS Machine Benchmark[2], required 0.41 CPU seconds for r300.5, 2.52 CPU seconds for r400.5 and 9.71 CPU seconds for r500.5. In the following, unless explicitly stated otherwise, all CPU times refer to the reference machine.

The performance results for PLS on Benchmark–1 are shown in Table 1. To generate these results, 100 independent trials were performed for each instance using target clique sizes corresponding to those obtained in [4]. As shown, PLS achieved a 100% success rate on all Benchmark-1 instances while using considerably less processor time than that required in [4].

The performance of PLS for Benchmark–2 is shown in Table 2. To generate these results, an extensive trial was first performed to identify the putative maximum clique size for each benchmark instance. Using the putative maximum clique size obtained for each instance, 100 independent trials of PLS were performed using a $maxSelections$ of 100 000 000 to obtain the results shown in Table 2.

Figure 1 is an undirected graph showing the 0–1 contact maps for the 1KDI and 1PLA proteins and also the alignments (dotted lines) obtained by locating the maximum clique within the 1KDI–1PLA correspondence graph. The consistency of the alignments can be verified by the observation that there are no intersections between any alignment lines.

4 Conclusions and Future Work

The overall performance of PLS on the CMO instances reported here suggests that the underlying dynamic local search method has substantial potential to provide the basis for high-performance algorithms for other optimisation problems.

References

1. Lesk, A.M.: Introduction to Protein Architecture. Oxford University Press, Oxford, UK (2001)
2. Godzik, A., Skolnick, J., Kolinski, A.: Regularities in interaction patterns of globular proteins. Protein Eng. 6, 801–810 (1993)
3. Goldman, D., Istrail, S., Papadimitriou, C.: Algorithmic aspects of protein structure similarity. In: Proc. 40th Annual IEEE Sympos. Foundations Comput. Sci., pp. 512–522. IEEE Computer Society, Los Alamitos (1999)
4. Strickland, D.M., Barnes, E., Sokol, J.S.: Optimal protein structure alignment using maximum cliques. Operations Research 53, 389–402 (2005)
5. Garey, M.R., Johnson, D.S.: Computer Intractability: A Guide to the Theory of $\mathcal{NP}$-Completeness (1979)
6. Pullan, W.J.: Phased local search for the maximum clique problem. Journal of Combinatorial Optimization 12(3), 303–323 (2006)
7. Berman, H.M., Westbrook, J., Feng, Z., Gilliland, G., Bhat, T.N., Weissing, H., Shindyalov, I.N., Bourne, P.E.: The protein data bank. Nucleic Acids Research 28, 235–242 (2000)
8. Krasnogor, N., Pelta, D.A.: Measuring the similarity of protein structures by means of the universal similarity metric. Bioinformatics 20, 1015–1021 (2004)

[2] *Dmclique*, ftp://dimacs.rutgers.edu in directory/pub/dsj/clique

SMART: Structured Memory for Abstract Reasoning and Thinking

Umair Rafique and Shell Ying Huang

School of Computer Engineering,
Nanyang Technological University, Singapore
{umai0001,assyhuang}@ntu.edu.sg
http://www.ntu.edu.sg

Abstract. An important skill considered part of intelligent behavior is Abstract Thinking and Decision Making which includes thinking about a problem and reaching a decision after reasoning about its different aspects. This aspect of intelligent behavior has not recieved much attention and most of the cognitive architectures present today either focus more on the perceptual-motoric aspects of human brain or delve into the psycological, common behavioral and common sense issues. Here we present a cognitive architecture which addresses the issue of Abstract Thinking and Decision Making by using a novel representation for knowledge in the memory of an agent. The memory of an agent consists of four components, Concept Net, Working Memory, Perceptions and Possessions. Concept Net is a multi-layered net representing various concepts and their relationships with one another. We contend that this way of knowledge representation supports the process of decision making.

Keywords: Cognitive Architectures, Abstract Thinking, Decision Making, Knowledge Representation.

1 Introduction

Intelligence is an umbrella activity encompassing many different types of skills. Abstract Thinking and general purpose Decision Making is one such skill and covers thinking and reasoning about a problem and reaching a conclusion after considering all aspects and possibilities related to the problem. For example making a decision by a robotic house worker, "Shall I go to market for buying groceries". To simulate intelligent behaviours, many different types of cognitive architectures have been proposed and different representations for knowledge have been used in different models. Generally, the two broad categories can be recognized as symbolic and connectionist.

Although symbolic representations are very expressive in nature, encoding big domains using such kind of representation is very difficult since for every action and its preconditions and effects, the programmer has to code the details. Such representations are quite useful for addressing perceptual-motoric issues of human intelligence or for building such special purpose systems which naturaly create a search space for solution (solving puzzles, playing chess, etc) but for the task of abstract thinking and decision making, they are not very appropriate.

In connectionist way of representing knowledge, the emphasize is more on "connecting" nodes in different ways. Different architectures use different type of nodes and connect them in different ways but generally one node represents one piece of knowledge and its connections to its neighbors determine how often these nodes occur together and the weight of the link determines the strength. In our view there are other possible ways of connecting different nodes as well and we discuss them in later sections. Most of the architectures proposed to date using this way of representation address issues of general human behavior and common sense.

There can be found examples combining Symbolic and Connectionist approaches (e.g. ACT-R [1]). Others have suggested that we should use different kind of representations for different tasks, e.g. Society of Mind [5]. Architectures focusing more on providing perceptual-motoric skills (Soar [4], Icarus [3] and PRODIGY [8]) prefer symbolic representations and productions. Others using connectionist approach (Dual [2], Society of Mind, Push singh [6] etc) focus more on psychological aspects, social behavior and common sense .

CLARION [7] is very similar to ACT-R but focuses more on rule learning and rule extraction from its bottom layer which consists of a network of nodes. DUAL also combines the two approaches, symbolic and connectionist. Its nodes, which are called DUAL agents, have symbolic representations of knowledge and then it connects them at different levels in the form of a network.

Our focus in this paper is abstract thinking and decision making which the above mentioned architectures either have ignored or it is not their main focus and hence the representations favor the senso-motoric type of problems. We propose a new cognitive architecture, SMART (Structured Memory for Abstract Reasoning and Thinking), which uses a novel way of representing information. Although we use nodes and connect them to form a network, it is totally different from the mentioned connectionst architectures because here each node represents either a noun or an action while for others each node is normally a small processing unit. The links between different entities (nodes) in in our network represent semantic relationships. Our work can be described as a way of representing symbolic knowledge in the form of a network. The strength of our architecture is that we store knowledge about how different real world entities relate with each other, with each entity being in the network only at one place.

2 Architecture

2.1 Types of Nodes

In SMART, there are two different types of nodes named Noun Node (NN) and Action Node (AN). NN represents a real world noun entity, e.g. water, cloths, ball etc while an AN represents a real world action entity e.g. play, watch, go etc. We store the relationship among different entities in the form of a network which we call Concept Net (CN). CN is formed by joining all the entities which

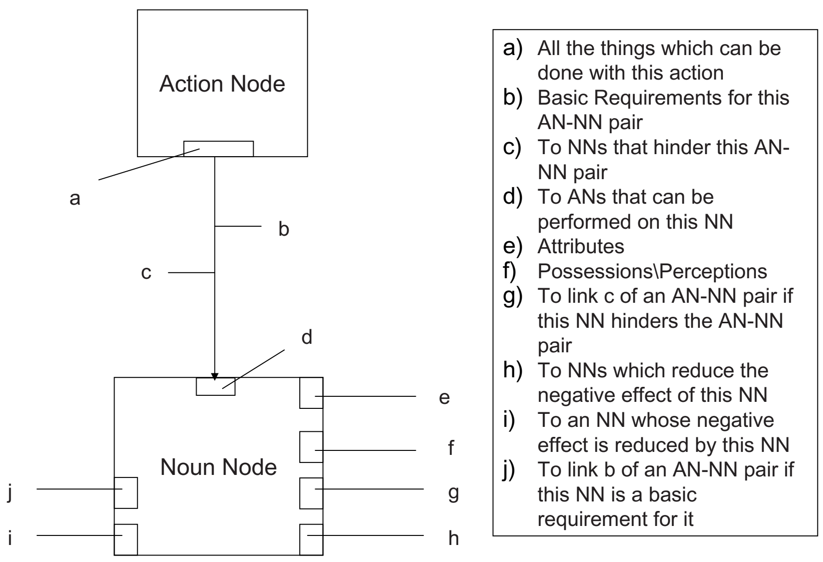

Action Node
Noun Node
a
b
c
d
e
f
g
h
i
j
a) All the things which can be done with this action
b) Basic Requirements for this AN-NN pair
c) To NNs that hinder this AN-NN pair
d) To ANs that can be performed on this NN
e) Attributes
f) Possessions\Perceptions
g) To link c of an AN-NN pair if this NN hinders the AN-NN pair
h) To NNs which reduce the negative effect of this NN
i) To an NN whose negative effect is reduced by this NN
j) To link b of an AN-NN pair if this NN is a basic requirement for it

Fig. 1. SMART at micro level, Action Nodes and Noun Nodes and their connections

are related to each other in a way that each connection represents a relationship. Figure 1, shows the way of joining the two types of nodes. For the time being we are focusing only on simple kind of relationships.

An AN has only one type of connection to one or more NNs. Since an action may be applied to many nouns (you can play any game, can watch anything, can go anywhere etc), the link from an AN can connect to anything that is linked with the action this AN represents.

The link between one pair of AN and NN represents one action-noun concept. For each action-noun concept, basic requirements (represented by link b in Figure 1) are the prerequisites for this action to take place (e.g. if you want to play a game, you must have at least the equipment that is required for playing that game). The other piece of knowledge for the action-noun concept (represented by link c) is the list of entities which have a hindering effect on it. This is one of the key features of our architecture and the base of any type of decision making. Whenever we have to make a decision about something, we immediately look for the basic requirements and examine what hinder us from achieving our goal. Then we make a tradeoff or find something that can help us reduce the negative effects of those hindering things, we make our decisions. Our architecture supports this reasoning process by the connections in the Concept Net.

An NN can have up to seven types of connections with other ANs and NNs. Figure 1 shows a generic NN. For a particular noun, some of the connections may not be applicable and will just be null. A NN may connect to one or more ANs which represent all the actions that can be done on this noun (represented by link d). Attributes are the main attributes of the noun and represent the properties the noun has (link e). Possession or perception link (link f) tells where to look for the availability of this object, in possessions or through perceptions (see next section).

As discussed earlier, link c connects to all the NNs which hinder an action-noun pair. Such NNs will be connected to link c of an action-noun pair using their link g. When a noun entity hinders some action-noun pair, different options that can reduce/eliminate negative effects of this noun entity can be found through link h. For example rain hinders our action-noun pair "go-market". So the c link of "go-market" leads to the noun node rain. From the h link of NN "rain" we know that an umbrella can save us from rain. So if it is available (based on search in possessions or perceptions), we can go out without being affected by rain. On the helping entity's side, it will end at link i. So rain's h link will connect to umbrella's i link. The last link j of a NN connects this noun entity to the b link of an action-noun pair if this noun entity is a basic requirement for the pair.

2.2 Memory Structure and Situation Monitoring

SMART's memory structure consists of four basic components. The first component, Concept Net, is composed of many layers of nodes which are connected with nodes in the same layer as well as with nodes in the upper layer (See Figure 2). Higher layers are for abstract concepts which are abstractions of concrete concepts in the lower layer. The number of vertical layers is not fixed and it depends on the abstraction of concepts. The nodes in one layer are connected as described in the last section. The bottom layer of the Concept Net stores all the concrete concepts an agent knows about the real world.

During the decision making process, only a small subset of nodes are relevant for a particular problem. These nodes will be copied to the second component, the Working Memory. In this way SMART can also monitor the atribute values and see if changes in environment will affect the current decision made. Third component will store perceptions about the enviorment and will work together with the Working Memory for situation monitoring. Fourth and the last component is a store for posessions. It consists of all the things an agent posseses. For example if an agent bought an umbrella then as long as it has it, it will be there in Posessions.

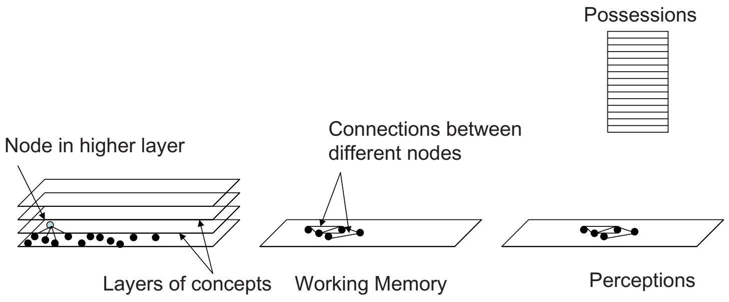


Fig. 2. Memory Structure

3 Discussion

We want to extend this theoretical concept to a fully implemented tool which can be used for abstract thinking and decision making in different domains. Another important development will be to investigate other new types of relationships between different nodes and to incorporate them into the architecture so that it can work for more complex problems.

Knowledge stored in SMART can be obtained from text documents e.g. newspapers, books, articles etc, as knowledge here is just a basic description of what an entity is and how it is related to others. Using an electronic dictionary is another possible way. We are working on writting such a parser which by parsing different documents and dictionaries can create a Concept Net that can be used in SMART. The parser should be able to find all the entities, i.e. nouns and verbs, which will constitute the nodes of the concept net and then it can establish the links among the relevant nodes. This knowledge can be presented to a trainer for updates and corrections if required.

References

1. Anderson, J.R., Byrne, M.D., Douglass, S., Lebiere, C., Qin, Y.: An integrated theory of the mind. Psychological Review 111(4), 1036–1050 (2004)
2. Kokinov, B.N.: The DUAL cognitive architecture: A hybrid multi-agent approach. In: European Conference on Artificial Intelligence, pp. 203–207 (1994)
3. Langley, P., Choi, D.: A unified cognitive architecture for physical agents. In: Proceedings of the Twenty-First National Conference on Artificial Intelligence (2006)
4. Lehman, J.F., Laird, J., Rosenbloom, P.: A gentle introduction to soar, an architecture for human cognition (2006)
5. Minsky, M.: The society of mind. Simon & Schuster, New York (1985)
6. Singh, P., Minsky, M.: An architecture for cognitive diversity. Visions of Mind (2005)
7. Sun, R. (ed.): An agent architecture for on-line learning of procedural and declarative knowledge, pp. 766–769. Springer, Heidelberg (1997)
8. Veloso, M., Carbonell, J., Pérez, A., Borrajo, D., Fink, E., Blythe, J.: Integrating planning and learning: The PRODIGY architecture. Journal of Experimental and Theoretical Artificial Intelligence 7(1), 81–120 (1995)

An Efficient Segmentation Technique for Known Touching Objects Using a Genetic Algorithm Approach

Edgar Scavino, Dzuraidah Abdul Wahab, Hassan Basri, Mohd Marzuki Mustafa, and Aini Hussain

Faculty of Engineering, Universiti Kebangsaan Malaysia, 43600 Bangi, Selangor DE, Malaysia
{scavino, dzuraida, drhb, marzuki, aini}@eng.ukm.my

Abstract. This paper presents a genetic algorithm (GA) based segmentation technique that can separate two touching objects intended for an automatic recognition of plastic bottles moving on a conveyor belt. The proposed method is based on the possibility to separate the two objects by means of a straight line, whose position is determined by a GA. Extensive testing shows that the proposed method is fast and yields high success rate of correct segmentation with only a limited number of both chromosomes and iterations.

Keywords: segmentation, genetic algorithm (GA).

1 Introduction

In computer vision based automated product inspection [1], image segmentation [2]-[3] is a fundamental step in the process of extracting meaningful features of objects from an input image. One common situation arises when the objects in an input image are in reciprocal contact. The segmentation algorithm must be able to identify the objects and draw the internal borders and overcome the complexity in shape and colour, instead of identifying uniform regions. This paper will focus on a common problem in the recycling process of waste plastic bottles. Typically, bottles are sorted according to the plastic type, which is usually recognized by means of IR, UV, X-ray detectors, or manually [4]. An automated, fast and inexpensive technique based on shape recognition of plastic bottles is presented in [5], however it requires a large database of image features. The bottles' random position, orientation and time of arrival in the camera observation add complexity to the recognition process.

The Hough, Euclidean distance and watershed transforms [6] have been attempted to separate two or more objects in contact, but all proved to be excessively sensitive to noise pixels and to the complex shape of some bottles. The watershed transform showed its usefulness in cases of short contact lengths, but in many cases it was not possible to identify two major minima in the distance transform and no segmentation was obtained. Also the colour difference was not chosen as a segmentation criterion as many bottles in the database are very similar in colour and some present contrasting internal patterns. The aim of this paper is to present a genetically optimized technique to separate two plastic bottles in contact under the following assumptions: a) the cluster is made of two touching bottles, whose dimensions, aspect ratios, momentums are stored in a database and b) the bottles' shapes are convex so that a meaningful segmentation may be done using a straight line. Particular attention has been given to the computation time of the algorithm for possible real-time applications.

2 Genetic Algorithm

In this work, a GA technique [7]-[9] is used due to its generality and capability to heuristically overcome situations where an exhaustive solution would be too computationally demanding. Its goal is to find a satisfying straight line which correctly separates a binarized cluster of two touching bottles into two subsets, each matching one of the bottles.

The *population P* of individuals consists of straight lines, connecting randomly chosen couples of perimeter pixels. To speed up the process, a small number of individuals were selected out of an initial larger population, as shown in Fig. 1 (*left*). Typical values, for a perimeter of 1000 pixels, are an initial population of 50 lines and subsequent ones of 10 lines. Individuals are coded as couples of indexes $P_k \equiv \{n_{1k}, n_{2k}\}$ which identify the pixels on the perimeter of the object. These are sorted and indexed in anticlockwise order so nearby pixels have near indexes.

The *cross-over* is performed on randomly selected couples of individuals by averaging their indexes:

$$\{n_{1k}, n_{2k}\}, \{n_{1k'}, n_{2k'}\} \rightarrow \{(n_{1k} + n_{1k'})/2, (n_{2k} + n_{2k'})/2\} \ . \tag{1}$$

This operation generates a new offspring whose vertices are approximately halfway between the vertices of the parents. The diversification arising from this crossover procedure does not guarantee that new individuals inherit or improve the fitness value of their parents. For this reason the two best individuals of a population are transmitted to the new generation without changes, in a process known as *elitism*.

The *mutation* is coded as random changes Δ_1, Δ_2 in the values of the pixels' indexes, up to a maximum predefined extent equal to 2% of the total number of pixels:

$$\{n_{1k}, n_{2k}\} \rightarrow \{n_{1k} + \Delta_1, n_{2k} + \Delta_2\} \ . \tag{2}$$

This mutation is applied to all new individuals in order to explore the vicinity of groups of similar individuals rather than explore different and faraway segmentations.

The *fitness* of a line has to be somehow related to its effectiveness in cutting the original binarized image into two fragments, both very similar to two of the bottle images in the database. For each line k, the length Δl and width Δh of both fragments are measured along their principal axes, as shown in Fig. 1 (*right*). The dividing line is added to both parts to avoid the displacement of the centroid with respect to the centroid position of the complete perimeter. Then, for each possible couple of bottles t, a fitness value is calculated as the sum of exponentials of the absolute errors:

$$\begin{aligned} f_{kt} &= \exp(-|\Delta l_{k1\to t1}| - |\Delta h_{k1\to t1}|) + \exp(-|\Delta l_{k2\to t1}| - |\Delta h_{k2\to t1}|) + \\ &+ \exp(-|\Delta l_{k1\to t2}| - |\Delta h_{k1\to t2}|) + \exp(-|\Delta l_{k2\to t2}| - |\Delta h_{k2\to t2}|) \end{aligned} . \tag{3}$$

in which the indexes *k1*, *k2* refer to the two parts identified by the cutting line, *t1*, *t2* refer to the two bottles of the couple *t*. The total fitness of a line *k* is $f_k = \max(f_{kt})$. Tests have been performed with other features in addition, or in substitution, to the length and width of the two fragments. The use of the area of each part appeared to be too slow for fast applications, while using the length or the square momentum of the perimeter resulted in totally wrong cuts associated with high fitness values.

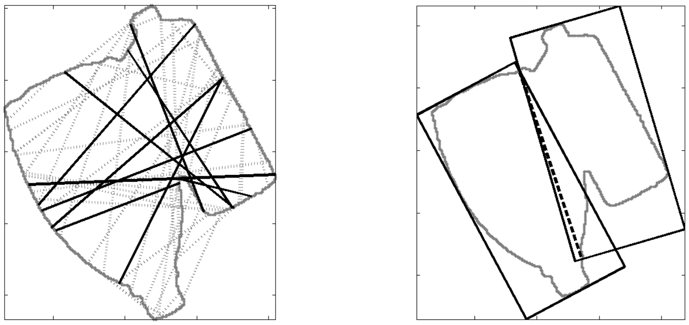


Fig. 1. (*Left*) Initial population of 50 lines, connecting random couples of perimeter pixels. The 10 fittest individuals (*black*) are kept as parents to produce new generations. (*Right*) The dashed line splits the object in two parts. Each fragment is measured along its principal axes then the width and length of the rectangles are used to evaluate the fitness of the cutting line.

3 Experimental Results

The GA-based segmentation technique was tested on images of touching bottles. We performed a visual inspection of the final segmentations to establish its quality and to rule out possible false positives, in which a high fitness level would correspond to wrong cuts. Fig. 2 shows two examples of successful segmentations, obtained with an initial population of 50 lines which were reduced to 10 prior to implementation of the actual GA. Twenty iterations were performed in both examples and fitness values of f=3.69 and f=3.66 were obtained. It was also observed that a line may have a high fitness level and provide a correct segmentation even without having its vertices lying exactly on the corner points common to both bottles.

The efficiency and the speed of this genetic algorithm were tested on a set of about 150 images depicting couples of bottles in contact. Fifty distinct couples were formed by choosing randomly out of a set of 50 bottles, whose features were stored in the database. For each couple, 3 - 4 separate images were taken in different contact patterns.

A selection of the candidate couples was performed by calculating the total area of the binarized image of touching bottles. Extensive testing showed that in all cases this area lay in the range 98.9% - 99.5% of the sum of the areas of the two bottles alone, as per values recorded in the database. This criterion, together with a check of all possible couples, showed a single worst case scenario in which 8 couples had to be taken into consideration. In most cases 3 to 5 couples would be required as candidates. During the test the touching bottles were put on the conveyor belt, both in motion and at rest. The images were taken with a standard webcam; its exposure time and the surrounding lighting were adapted to reduce blurring due to the conveyor belt movement.

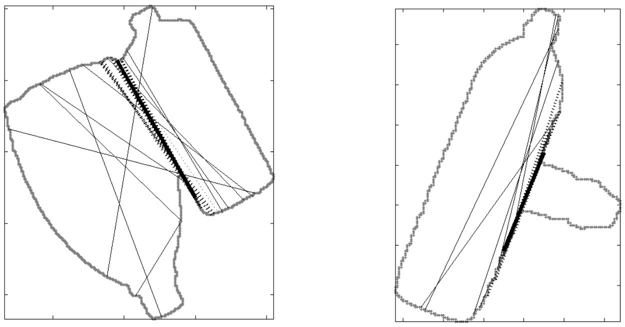


Fig. 2. Final result of the genetic algorithm segmentation in two examples of touching bottles. The initial population of 50 lines was reduced to 10 after the first screening. The solid lines represent the population after the first selection, the dotted lines are the final population and the thick black line is the best-fitting element.

The rate of successful segmentation is shown in Fig. 3. Each configuration was tested 100 times, with 20, 50 and 200 generations in the GA. Each result was visually inspected for false positives or correct recognitions associated to bad segmentations. High fitness values (f >3.6) correspond to good segmentations and correct identification (i.e., no false positives were recorded), while low ones (f <3.2) were associated with meaningless segmentations and random identification. Intermediate values of the fitness would lead to a correct identification but approximate segmentation. With just 50 iterations the average success rate climbs over 85% while more iterations provide only a marginal improvement. For the case when the parts in contact are long, success rates of more than 97% are obtained within just 20 generations.

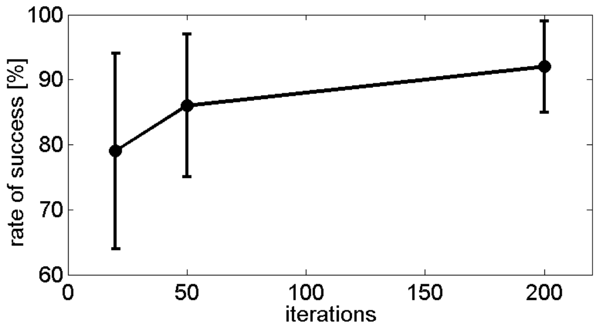


Fig. 3. The GA success rate as a function of the number of generations

Emphasis on the computing time performance of the algorithm was also considered to ensure possible real-time applications. The computing time was found to be roughly proportional to the number of candidate couples, as the initial line selection was much faster than the GA itself. When using 50 iterations and 10 lines for the GA, the average time was found to be around 80ms per candidate couple, in Matlab environment. In real-time applications involving the recognition of bottles on a conveyor belt, this procedure may be considered to be sufficiently fast and unlikely to be the bottleneck of the system. If a faster version is needed, a compiled version of the algorithm may reduce the computing time.

4 Conclusions

In this work, we have successfully implemented a GA based segmentation technique to separate touching bottles images of a waste sorting system that facilitates the identification of plastic bottle type. Typical issues in GA, such as selection of initial line population, parameters for fitness calculation as well as the cross-over and mutation optimisation were addressed. Results show that our GA implementation could provide a success rate above 85% with reasonable computing time for possible real-time applications. Using just 50 initial chromosomes, 10 selected chromosomes and 50 generations achieve the goal of 80ms per candidate couple of bottles.

References

1. Pham, D., Alcock, R.: Smart Inspection Systems. Academic Press, San Diego (2003)
2. Haralick, R.M., Shapiro, L.G.: Survey: Image Segmentation Techniques. Comp. Vision Graph. Image Process. 29, 100–132 (1985)
3. Melkemi, K.E., et al.: A Multiagent System Approach for Image Segmentation Using GA Algorithms and External Optimization Heuristics. Pattern Recognit. Lett. 27, 1230–1238 (2006)
4. Mital, A., et al.: A Comparison Between Manual and Hybrid Methods in Parts Inspection. Integrated Manufacturing Systems 9, 344–349 (1998)
5. Wahab, D.A., et al.: Development of a Prototype Automated Sorting System for Plastic R cycling. Am. J. Appl. Sci. 3(7), 1924–1928 (2006)
6. Haris, K., et al.: Hybrid Image Segmentation Using Watersheds and Fast Region Merging. IEEE Trans. Image Process. 7, 1684–1699 (1998)
7. Goldberg, D.E.: Genetic Algorithm. In: Search, Optimization and Machine Learning, pp. 28–60. Addison-Wesley, Reading MA (1989)
8. Koza, J.R.: Survey of Genetic Algorithms and Genetic Programming. In: Proc. of the We con95-Conf. Record: Microelectronics, Communications Technology, Producing Quality Products, Mobile and Portable Power, Emerging Technologies, San Francisco CA (1995)
9. Yoshimura, M., Oe, S.: Evolutionary Segmentation of Texture Image Using Genetic Algorithms towards Automatic Decision of Optimum Number of Segmentation Areas. Pattern Recognit. 32, 2051–2054 (1999)

Elements of a Learning Interface for Genre Qualified Search

Andrea Stubbe[1], Christoph Ringlstetter[2], and Randy Goebel[2]

[1] CIS, University of Munich
[2] AICML, University of Alberta

Abstract. Even prior to content, the genre of a web document leads to a first coarse binary classification of the recall space in relevant and non-relevant documents. Thinking of a genre search engine, massive data will be available via explicit or implicit user feedback. These data can be used to improve and to customize the underlying classifiers. A taxonomy of user behaviors is applied to model different scenarios of information gain. Elements of such a learning interface, as for example the implications of the *lingering time* and the *snippet genre recognition factor*, are discussed.

1 Introduction

Given a web user's information need, even prior to content, the genre of a web page leads to a first coarse binary classification of the recall space in immediately rejected documents and such that require further processing. A search engine that provides automatic classification into genres, as for example, *shopping portals*, *scientific papers* or *personal web pages*, could make a big difference in regards to the number of documents that have to be checked for relevancy.

If such an interface is available in public, a steady stream of user events will arise. These behavioral observations have to be turned into meaningful data either to improve the performance of the initial classifiers or to adapt to genre shift. All attempts to aquire such data from a running system have to consider the user's level of explicitness and cooperativness. Discussed in detail are the aspects of a silent genre interface where the user's statements on the genre of a document are only provided implicitly. In that connection, two qualities play a decisive role: first, knowledge about the implications of the *lingering time*, the time a user spends with a certain web page, will help to improve the precision of the genre classifiers; second, the *snippet genre recognition factor*, the percentage of documents whose genre a user can identify by only referring to the snippet, influences possible improvements of recall. To investigate the adaptability of genre classifiers for different scenarios, we simulate user feedback on genre labeled result sets by annotated corpus data.

2 Genre Qualified Search

To enable genre qualified search we need a classification schema, features, classifiers, and a search interface.

Document genres. The technical term "genre" refers to the partition of documents into distinct classes of texts with similar function and form. In [1] we introduced a genre hierarchy consisting of 8 container 32 leaf classes that tries to meet the demands of genre focused search. With regards to the main purpose of this study, the adaption of classifiers by user data, we exemplified results by five genres, three rather distinct ones: *blog, catalog, faq*, and the two journalistic genres *news* and *interview*.

Features. Feature selection was organized on training corpora comprising 20 prototype documents for each genre. Together over 200 different features were considered to organize the 32 leaf genres, including form, vocabulary, and parts of speech, complex patterns, and combinations of all these [1]. Examples of features are content-to-code-ratio, average line length, number of names, positive adjectives, dates, or bibliographic references. An example of a high level structure is a *casual style of writing* that can be recognized by the number of contractions (e.g. "won't") and the use of vague, informal, and generalizing words.

Specialized classifiers. To reach minimal description we introduced specialized features for each single genre classifier [1]. The features were arranged into a conjunction of single rules, applying a human supervised selection process that prevents overfitting by statistical coincidence on small training samples.[1] In Table 1 we show a cross-section of the specialized classifier of the genre *news*.

Table 1. Cross-section of the rule based classifier for the *news* genre

textlength in characters, forms
$length > 200 \wedge length < 6500 \wedge headlines < 3 \wedge sent > 1$
part of speech
$verbs \geq 5 \wedge adjectives < 20 \wedge adjectivesPositivNegativ < 0.4$
spoken/written text
$(contractions < 0.4 \vee dirSpeech > 0) \wedge contractions/dirSpeech < 0.2$
tense
$verbsPastTense < 0.18 \wedge verbsPastTense > verbsPresentTense \wedge$
$verbsIngForms > verbsPresentTense$

Search interface. To enable genre qualified search, an explicit interface has to provide functionalities for genre input and user feedback. Since "most users are reluctant to additional work"[2], a more realistic variant is a *silent interface* that minimizes the cognitive load of the user. Desired genres have then to be deduced from the query combined with locally or globally aggregated knowledge about the user. The feedback of the user is derived from his observable navigation on the result set.

3 Adaption of the Specialized Genre Classifiers

To make our static classifiers adaptive we had to rewrite them in disjunctive normal form (DNF). Alternative rules are combined by a logical *OR*; within the disjunctive

[1] For research purposes the corpora, features, and classifiers are available at http://www.cis.uni-muenchen.de/˜andrea/genre/.

elements only connections by logical *AND* are allowed. Lower *and* upper bounds of the features' numerical ranges have to be explicit. The adaptions of the classifier's ranges are normalized to the interval $[0..1]$. Two different situations have to be distinguished:

(1) False negative: A document of genre N_i has not been recognized as N_i. For every disjunctive element of the classifier in DNF form, we compute the sum of the required range adaptions to achieve a correct classification of the input document. The element with the minimum sum is selected and its ranges are temporarily adapted if the sum does not exceed a maximum threshold that prevents adaption to outliers or deliberately wrong feedback.
Constraint: The performance of this temporarily adapted classifier is then computed for all documents seen so far to find out whether the changes lead to an overall improvement. Generally, the files that are classified correctly attendant on the classifier adaption (new correct positives) have to outnumber the files that are now falsely classified (new false positives). In particular, for the purpose of modeling *genre-shift*, i.e. the modification of feature-value sets that determine a genre, a temporal discount factor can be introduced.

(2) False positive: A document of genre N_j has been falsely recognized as genre N_i. We identify elements of the disjunction that have confirmed the input document as N_i. Within the elements, we look for the smallest sum of adaptions that prevent the positive classification of the document.if the sum does not fall bellow a minimum threshold that prevents adaption to outliers or deliberately wrong feedback.
Constraint: Generally, the number of files for the *relevant* history that are classified correctly attendant on the classifier adaption (*new correct negatives*) has to be larger than the number of files that are now falsely classified (*new false negatives*).

4 User Behavior and Information Gain

Given different feedback events, the question arises of how information can be derived under the conditions of an increasingly realistic model of user behavior. Two major problems have to be faced: the loss of information, and the introduction of noise.

Fully cooperative User. The *fully cooperative user* provides the interface with complete information about the binary classification of the presented data. The system obtains a completely labeled additional dataset.

Cooperative User. A rational cooperative user will retrieve pages of the desired genre and will give feedback as to whether they were correctly classified. If not enough positively labeled pages are available, she will try to identify documents of the desired genre by the snippet information (s.b.), retrieve pages, and give feedback on the genre. According to studies of standard search engines [3], the average number of visited pages per search session is less than two and in most cases these two pages are retrieved from the first 20 hits of the search results. Given a fair amount of genre labeled documents, an average number of only two retrieved pages per turn leads to a strong preference of events that can help to improve precision.

Uncooperative User. With uncooperative user behavior, only the *lingering time*, an implicit source of information, is available to generate user statements. The lingering time of the user on a retrieved result page, depending on genre, topic, and model exogenous factors, is transformed into a binary signal: if it falls short a certain threshold τ, a negative relevancy signal for the document is assumed. If we abstract from model exogenous events, a document is irrelevant either because of wrong topic or because of wrong genre. Unfortunately, in a realistic scenario the precision of a search engine with regards to topic seems to be by far from perfect. This prevents us from gathering reliable data on the correctness of the genre classification via a negative relevancy signal. This leaves the case where the lingering time exceeds the threshold and a positive relevancy signal is hypothesized. This hypothesis is incorrect if the user stays on the web page because of exogenous factors. The two probabilities, $P(relevant(x)|time(y) > \tau)$ and $P(\neg relevant(x)|time(y) > \tau)$, can be estimated by frequency counts during a controlled user study. If a document is actually relevant, this case again can be further divided into relevancy of topic with and without the document being of the desired genre, $P(c(x) = label(x)|relevant(x)), P(c(x) \neq label(x)|relevant(x))$. For a rational user of the genre search interface, we expect the relevant cases that come with the wrong genre to be much rarer than those with the correct genre, $P(c(x) = label(x)|relevant(x)) >> P(c(x) \neq label(x)|relevant(x))$. After collecting data for the estimation of these basic probabilities, the problem of data loss and introduction of noise for the four a posteriori events of genre recognition can be modeled:

(1) Document of desired genre. The case of a document x actually being of desired genre, $c(x) = N^{desired}$ is subdivided into correct positives, $c(x) = label(x)$, and false negatives, $c(x) \neq label(x)$.

(1.1) Correct positive. To get a positive relevancy signal for cases where the correct genre has been recognized the topic must be relevant. Insofar, we have to expect data loss with a factor of $1 - precision(topic(x))$ and a small data gain via accidental confirmations by an exogenous event.

(1.2) False negative. Data gained for this case can improve the recall of the classifiers. As to the confirmation by exogenous events the probability is the same as for case (1.1). A difference concerns the data loss. Here not only is data lost by irrelevant topic but also by the user not recognizing the document as being of the desired genre. The problem lies in the indirect access to the document only enabled via the document's snippet. Data loss is additional in the size of the *snippet recognition factor* (s.b.). For both cases (1.1,1.2), we get no introduction of noise since noise could only be introduced by a negative lingering signal. However, as mentioned, negative signals are not reliable and because of that left out of consideration.

(2) Document not of desired genre. We have false positives, $c(x) \neq label(x)$ and correct negatives $c(x) = label(x)$.

(2.1) False positive. The problem with the data gain for false positives is that they can be amplified by a positive lingering signal. For the relevant documents, this is the portion where the topic is relevant and the genre is not, $P(relevant(x)|time(x) > \tau)P(c(x) \neq label(x)|relevant(x))$. For the non-relevant documents where the genre was falsely identified as the desired this the portion that is amplified by an exogenous event. $P(\neg relevant(x)|time(x) > \tau)P(c(x) \neq label(x)|\neg relevant(x))$.

(2.2) Correct negative. For the last case, the correct negatives, in a rational environment where the user only retrieves documents that he assumes to be of the wanted genre, the introduction of noise depends on *the snippet recognition factor*. If a document is retrieved via missrecognition of the snippet, it can be wrongly confirmed by an exogenous event.

5 Experiments on the Snippet Genre Recognition Factor

If a user retrieves a negatively labeled document, he concludes from the snippet it does fall among the desired genre. Since the document's snippet is the communicative act of the search engine to feature query results, it is fundamental for the implicit user feedback how well the user performs in deriving the genre of a document from its snippet. To give first experiments on the *snippet recognition factor*, we used our annotated genre corpus [1]: we selected a document and set up a query to a search engine (Google). The query was a combination of several keywords that the engine would use to construct the snippet and a defining N-Gram. In experiment (1) we retrieved five snippets for each of the eight journalistic genres of our hierarchy. These 40 snippets were presented to five users with the request to classify them within a time range of < 15 sec each. Table 2 shows the results. With an overall precision of 54.24% and a recall of 45.50%, the results point to a high amount of data loss. The low recognition rate could also lead to some amount of noise introduced by a combination of falsely retrieved documents and exogenous induced lingering. On the other hand, the genres *interview* and *portrait* seem to be identifiable with high accuracy. The variations are caused by the differences in the communication of the document genre by the snippet. For experiment (2) we used 20 snippets of each of the more distinct genres: *blog,catalog, faq* and 10 *news* documents. These more distinct genres seem to be much easier to distinguish. Only for *blog* and *news* a higher number of documents are confused (cf. Table 3).

Table 2. Users' recognition of journalistic genres by snippets. Precision and recall in percent.

Genre	*Precision*	*Recall*	*Genre*	*Precision*	*Recall*
A.1 commentary	42.86	48.00	A.5 interview	90.90	40.00
A.2 review	68.42	52.00	A.6 news	32.35	44.00
A.3 portrait	84.21	64.00	A.7 feature	34.48	40.00
A.4 marginal note	45.00	36.00	A.8 reportage	35.71	40.00

Table 3. Users' recognition of the genres blog, catalog and faq by snippets. Precision and recall in percent.

Genre	*Precision*	*Recall*	*Genre*	*Precision*	*Recall*
E.2 catalog	90.57	87.27	F.3 blog	62.50	90.90
C.4 faq	98.67	82.22	A.6 news	77.65	69.47

6 Experiments on Classifier Adaption

For our experiments on classifier adaption, we set, on average, a number of two retrieved pages per turn [3]. The rational user prefers to visit the positively labeled documents and he is assumed to be able to derive the desired genre with a certain accuracy from the snippet (*snippet genre recognition factor*). To conduct the experiments, we used annotated genre data: for the three example classifiers, *blog, catalog,* and *faq*, the corpora provided by Marina Santini [4], each split into 160 documents for training and 40 documents for measuring recall. For the training/testing with negative examples we used 620/980 documents of a balanced genre corpus. We randomly generated 48 result sets to simulate the user behavior. Each set consisted of 20 documents, containing on average 3 documents of the desired genre. For the second experiment we collected 300 documents for *interview* and 100 for *news*. For *news* we worked with a holdout of 20 positive and 200 negative documents. The results of the adaption process of the rule based classifiers are shown in Table 4. In addition Table 5 shows experiments on the performance of an SVM-classifier provided with an extended training set [5].[2]

Fully cooperative user. Summarized, even assuming a static feature space, a significant improvement of the classification can be achieved by using fully labeled data. For the *news* classifier with an extremely bad start configuration, it seems that further gain can only be expected from a dynamic feature set.

Table 4. Results for Recall and Fallout (in percent) of the adapted and the original classifiers (in parentheses) for the fully cooperative, the cooperative, and the uncooperative case. Test set for the first three genres, homogeneous with the data of the adaption process (Santini corpus).

Genre	$Recall^{Rule}$	$Fallout^{Rule}$	$Recall^{Rule}$	$Fallout^{Rule}$	$Recall^{Rule}$	$Fallout^{Rule}$
	Fully Cooperative Case		Cooperative Case		Uncooperative Case	
blog	72.50(57.50)	1.85(0.13)	83.40(57.50)	6.36(0.13)	72.50(57.50)	2.26(0.13)
catalog	52.50(40.00)	1.19(0.27)	52.50(40.00)	1.06(0.27)	52.50(40.00)	0.97(0.27)
faq	77.50(52.50)	4.29(1.20)	75.00(52.50)	1.91(1.20)	67.50(52.50)	1.91(1.20)
interview	67.50(55.00)	2.26(1.61)	65.00(55.00)	1.93(1.61)	60.00(55.00)	1.77(1.61)
news	30.00(5.00)	12.00(1.50)	25.00(5.00)	8.00(1.50)	10.00(5.00)	4.50(1.50)

Cooperative User. For labeled pages of the desired genre the user gives feedback whether or not they were correctly classified. In lack of enough documents he will try to derive the missing label from the snippets and give feedback on the genre. In an experimental run for the *faq* corpus, out of the 48 result sets a feedback of 60 correct positives, 28 false positives, 6 false negatives, and 2 correct negatives emerged. Despite of that only these few interesting examples are enough to adapt the borders of a classifier [6]. Also for the two converging SVM classifiers the small amount of selected training examples led to an improvement of F1 values.

Uncooperative user. We deliberately introduced the basic probabilities of 0.9 for the lingering time exceeding τ with given document relevancy and 0.95 for a relevant document being of relevant topic *and* relevant genre. The topic precision was set to 0.5. Those values lead to a data loss of 45% for the correct positives and the false negatives. By the exogenous factors we get a 12% introduction of noise, wrong positive amplification of documents that are not of the desired genre, for the chosen correct negatives and the false positives. For the experiment with *faq* we received 0 feedback examples

[2] To reach comparability for each genre, we used only the aggressively prunned feature set of the specialized classifiers.

Table 5. Results for Recall and Fallout (in percent) of SVM classifiers trained on the extended and the original datasets for the fully cooperative, the cooperative, and the uncooperative case

Genre	$Recall^{SVM}$	$Fallout^{SVM}$	$Recall^{SVM}$	$Fallout^{SVM}$	$Recall^{SVM}$	$Fallout^{SVM}$
	Fully Cooperative Case		Cooperative Case		Uncooperative Case	
blog	72.50(65.00)	2.14(1.07)	72.50(65.00)	2.14(1.07)	57.50(65.00)	2.14(1.07)
catalog	47.50(42.50)	1.37(0.31)	45.00(42.50)	1.98(0.31)	45.00(42.50)	0.92(0.31)

for false positives, 40 for correct positives, 6 for false negatives, 0 for correct negatives, 1 noisy example for correct positives and 7 noisy examples for false negatives. Both classifier types showed fairly robust improvements despite the data loss and the introduction of noise.

7 Conclusion

We introduced elements for the improvement of genre classification by user data. For experiments with corpus based simulated user events, we could achieve improvements of the original classifier setup. These improvements showed a remarkable stability against noise and data loss caused by misscategorized user events for more realistic, less cooperative user models.

With regards to the snippet recognition factor, we infer from first experiments that the ability to identify the genre of a document by its snippet varies significantly between the genres. Overall, the recognition accuracy seems high enough to derive data from events where the user chooses a document that was not classified as the desired genre.

Our future goals are to provide a prototype of a genre interface to collect data for the estimation of currently assumed probabilities, as, for example, the correlation between lingering time and the correctness of genre classification by snippets, and to extend the classifier adaption from static to dynamic feature space.

References

1. Stubbe, A., Ringlstetter, C., Schulz, K.U.: Genre as noise: Noise in genre. International Journal on Document Analysis and Recognition (to appear, 2007)
2. Henzinger, M.: Search technologies for the internet. Science 317, 468–471 (2007)
3. Silverstein, C., Marais, H., Henzinger, M., Moricz, M.: Analysis of a very large web search engine query log. SIGIR Forum 33(1), 6–12 (1999)
4. Santini, M.: Common criteria for genre classification: Annotation and granularity. In: Workshop on Text-based Information Retrieval (TIR 2006), Riva del Garda, Italy (2006)
5. Joachims, T.: A statistical learning learning model of text classification for support vector machines. In: SIGIR 2001. Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval, pp. 128–136. ACM Press, New York, NY, USA (2001)
6. Schohn, G., Cohn, D.: Less is more: Active learning with support vector machines. In: ICML 2000: Proceedings of the Seventeenth International Conference on Machine Learning, pp. 839–846. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (2000)

A System for Acquisition of Noun Concepts from Utterances for Images Using the Label Acquisition Rules

Yuzu Uchida and Kenji Araki

Graduate School of Information Science and Technology, Hokkaido University,
Kita-14, Nishi-9, Kita-ku, Sapporo, 060-0814 Japan
{yuzu, araki}@media.eng.hokudai.ac.jp

Abstract. This paper presents a system which acquires noun concepts (labels for images) based on infant vocabulary acquisition models. In order to improve its performance, we introduced "label acquisition rules" into this system.

1 Introduction

Various types of robots to entertain people have been developed in recent years [1][2][3][4]. These robots often have a dialogue capability, but people are not satisfied at their communication skills. The robots should be capable of learning new knowledge and have a higher capability to keep people interested over time.

We developed a **S**ystem for **N**oun **C**oncepts **A**cquisition from utterances for **I**mage, called SINCA in short (which means "evolution" in Japanese). It is a language acquisition system without knowledge of grammar and vocabulary, which learns noun concepts from user inputs. The system was not sufficient in terms of usage of knowledge from experience, therefore we introduced a "label acquisition rule" into it. By doing so, SINCA can learn phrases that tend to contain noun concepts.

Currently, SINCA is for Japanese only. The language acquisition method of this system is very simple. The method is independent of language rules. Therefore SINCA is expected to work successfully while using any language.

2 Algorithm of SINCA

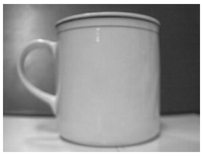
2.1 Input

A user input consists of an image paired with a spoken utterance. Figure 1 shows some examples of inputs.[1]

First, a user chooses an object O which he or she likes and captures its image with a web camera. A user has to try to capture the whole object O in the image.

[1] Strings indicated in boldface are labels and in italic are Japanese.

Next, a user imagines an utterance that an infant might be exposed to when listening to caregivers while gazing at the object O in the environment. A user enters the utterance on the keyboard as a linguistic input. SINCA does not carry out morphological analysis for the linguistic input, because we think infant capacity for word segmentation is not perfect[5].



Kore-ha ***KAPPU****-tte iu-n-da-yo.*
(This is a cup.)
KAPPU*-ni gyūnyū ireyōka.*
(Let's pour some milk into the cup.)

Fig. 1. Examples of input data

2.2 Image Processing

We use the "ERSP vision" included in the ERSP 3.1 Software Development Kit[6] for image processing. The ERSP Vision enables a robot or device to recognize 2D and 3D objects in real world settings where lighting and placement are not controlled. It tells whether the object in the present input image appears in the previous input images or not.

2.3 Common Parts and Basic Score

When a user inputs an image of the object O and an utterance, the system extracts all sections of a string that matches with the previous input utterances with the image of the same object O. We call the strings common parts. After this process, the system deals with them as candidates of a label for the object O.

The system provides every common part with a "basic score". The basic score is based on frequency of appearance and number of characters, and means how proper for the label the common part is. The basic score is defined as follows:

$$SCORE = \alpha \times \frac{F}{PN} \times \sqrt{L} \tag{1}$$

where, α is a coefficient which reduces the basic score if the common part has appeared with other object than O, F is frequency of appearance of the common part with the images of O, PN is the number of times that the user inputs with the images of O, and L is the number of characters of the common part.

2.4 Output

If the system finds a common part which basic score exceeds a threshold, outputs it as text. The reason for doing this comes from the belief that there is a high possibility that such common parts are proper for a label.

A user evaluates an output by choosing one of the following keywords:

– Good : It is proper for the label.
– Almost : It makes some sense but is not proper for the label.
– Bad : It makes no sense.

The system recalculates the basic score based on the keyword chosen by the user.

2.5 Acquisition of the Noun Concepts

After repeating these processes, if there is a common part which score is more than 30.0 and which has been rated as "Good", the system acquires the common part as the label for *O*.

2.6 Label Acquisition Rules

Humans can put their newfound knowledge to learn their native language effectively. This system imitates humans' way with "label acquisition rules".

A label acquisition rule is like a template, which enables recursive learning for acquisition of noun concepts. The system generates the label acquisition rules after acquisition of a label. When the system acquires a string S as a label for an object, the system picks up the previous linguistic inputs with the images of the object which contain the string S. Then, the system replaces the string S in the linguistic inputs with a variable "@". These abstracted sentences are called the label acquisition rules. To give an actual example, if an acquired label is "***WAN-CHAN*** (a doggy)" and a previous input is "*Atchi-ni* ***WAN-CHAN****-ga iru-yo.* (There is a doggy over there.)", then a label acquisition rule is "*Atchi-ni* ***@****-ga iru-yo.* (There is @ over there.)".

If there are matching rules with linguistic inputs, the system extracts strings which are proper for variable "@". The scores of extracted strings are increased.

3 Evaluation Experiment of SINCA with Label Acquisition Rules

3.1 Experimental Procedure

We compare a system using label acquisition rules with the one not using them. A user inputs the same data into both systems until they acquire 30 labels. The linguistic input data were collected by a questionnaire from 31 people. 1001 sentences were collected. We use 396 containing words representing labels for objects.

3.2 Experimental Results and Considerations

The system acquires 85 types of rules until it acquires 30 labels. Of them, only 13 rules are applied of 74 times in total. The system with the rules needs 5.6

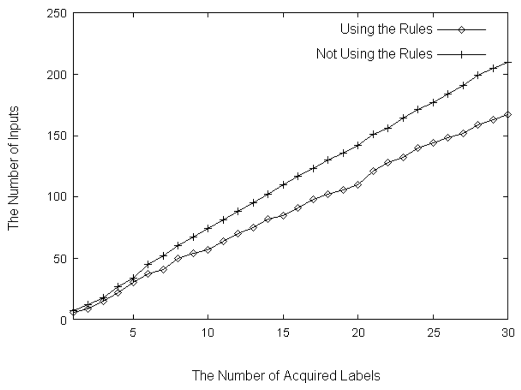


Fig. 2. The relation between number of inputs and acquisition of labels

inputs on average to acquire one label. In contrast, the system without the rules needs 7.0 inputs on an average. Figure [2] shows how many inputs the systems need to acquire labels.

This result indicates that using them makes the acquisition of labels more efficient. A lot of rules are acquired, but only 15.3% of those are applied. This means that a small proportion of the highly capable rules are applied. We should therefore consider deleting unnecessary rules.

4 Experiment to Evaluate the Generality of Label Acquisition Rules

Infants do not hear only infant-directed speech but also a TV set playing, conversations between adults, and so on. On the contrary, the user inputs for SINCA are always utterances that adults direct to it. SINCA uses web-based information to make up for this difference.

We get generalities of the label acquisition rules based on the results of Google search[7]. By these generalities we confirm if deleting unnecessary rules is possible.

4.1 Experimental Procedure

We search for every rule on Google. Variables in the rules are replaced by wildcards. To give an actual example, if an acquired rule is "*Kore-ha @ da-yo* (This is a @.)", a search query is "*Kore-ha * da-yo* (This is a *)".

4.2 Experimental Results and Considerations

Figure [3] shows relationship between the results of Google search and the number of rules. There are 41 rules where Google search returns 0 hits. It is nearly half

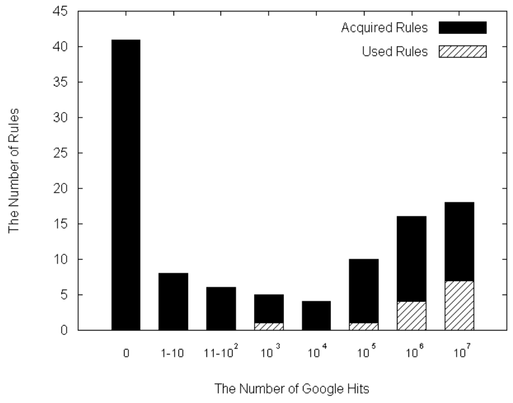


Fig. 3. The relation between label acquisition rules and Google search results

of all rules. It appears that they are too specific to use in SINCA. They were decided to be deleted.

It would appear that it is appropriate to select the rule based on Google search results. The results of this experiment show that rules which get less than 10,000 hits on Google can be deleted.

5 Conclusions and Future Work

In this paper, we described the algorithm of SINCA. It can acquire labels for images without ready-made linguistic resources, lexical information, and syntactic rules. We introduced label acquisition rules to improve the system.

In the next step, we will focus on acquisition of adjectives. If it proves successful, SINCA can have a greater diversity of expression.

References

1. Sony Corporation.: Aibo (2006), http://www.sony.net/Products/aibo/
2. NEC Personal Robot Research Center.: Papero (2005), http://www.incx.nec.co.jp/robot/english/robotcenter_e.html
3. Wow Wee Ltd.: Robosapien, http://www.wowwee.com/robosapien/robo1/robomain.html
4. Speecys.: ITR (2006), http://speecys.com/english/
5. Jusczyk, P.W., Houston, D.M., Newsome, M.: The beginnings of word segmentation in english-learning infants. Cognitive Psychology 39, 159–207 (1999)
6. Evolution Robotics, Inc.: ERSP 3.1 Robotic Development Platform OEM Software by Evolution Robotics, http://www.evolution.com/products/ersp/
7. Google.: http://www.google.com/

Branching Rules for Satisfiability Analysed with Factor Analysis*

Richard J. Wallace[1] and Stuart Bain[2]

[1] Cork Constraint Computation Centre and Department of Computer Science
University College Cork, Cork, Ireland
[2] Institute of Transport and Logistics Studies
Faculty of Economics and Business
University of Sydney, Sydney, NSW 2006, Australia
r.wallace@4c.ucc.ie, stuartb@itls.usyd.edu.au

Abstract. Factor analysis is a statistical technique for reducing the number of factors responsible for a matrix of correlations to a smaller number of factors that may reflect underlying variables. Earlier experiments with constraint satisfaction problems (CSPs) using factor analysis suggested that there are only a few distinct principles of heuristic action. Here, this work is extended to the analysis of branching rules for SAT problems using the Davis-Putnam algorithm. These experiments show that just as with CSPs, there seem to be two basic actions that distinguish heuristics, characterised as building up of contention and propagation of effects to the uninstantiated portion of the problem.

1 Introduction

In spite of the many additional features that are now incorporated into modern SAT solvers, most are still based on the Davis-Putnam procedure. The general operation of this procedure is that at each node in the search tree, search will choose a variable to branch on and instantiate it with one of its two possible boolean values. Should search ever return to this node, it must explore the corresponding subtree that results from instantiating the other value. Such a procedure requires a branching heuristic to select the variable at each node.

There are many branching rule heuristics for the satisfiability problem. Existing analyses of SAT variable ordering heuristics have suggested a number of hypotheses to explain their performance. One is the "satisfaction" hypothesis [1], which attributes the performance of a heuristic to its ability to branch into a satisfiable subproblem. Although the satisfaction hypothesis assisted in the development of some good branching rules, the underlying assumption that it is "satisfaction" alone that leads to good performance was later discredited [2]. An alternative hypothesis [2] was that heuristic performance could be explained by the amount of "simplification" engendered by a variable selection. This is because branching into an easily refutable subproblem quickly allows search to return to exploring a hopefully satisfiable, and simplified, subproblem.

The present work is based on a new approach to heuristic classification that considers the pattern of variation in search efficiency across problems, since similar patterns of

* This was supported by Science Foundation Ireland under Grant 00/PI.1/C075.

variation must reflect similar decisions, given the same basic algorithm. In this work, these patterns are analysed using the well-known statistical technique of *factor analysis*. This approach ensures that the maximum amount of the variance is accounted for by the smallest number of factors. By using the latter criterion, factor analysis can often extract orderly relations that are not always obvious in the original correlations, and gives results that are superior to simpler techniques for clustering.

Prior results using this approach suggest that there are just two underlying actions or behaviours that affect CSP heuristic performance [3], the effects of which might be generally described as "look-back" (building up contention to produce failure) and "look-ahead" (propagating effects to unincorporated parts of the problem). As SAT problems are a subclass of CSPs, it might be supposed that SAT heuristics can be broadly divisioned according to these two behaviours as well.

2 Background: Factor Analysis

"Factor analysis" refers to a collection of methods for reducing a set of variables to a smaller set that is equivalent in the sense that the set of measurements associated with the original variables can be derived from linear combinations of the new variables. This allows the investigator to re-interpret the original measurements in terms of a smaller set of possibly more basic variables. For details on these procedures see [4] [5].

The basic factor analysis model can be written as a set of linear relations of the form (taken from [4]):

$$z_{ij} = a_{j1}F_{i1} + a_{j2}F_{i2} + \ldots + a_{jm}F_{im} + d_jU_{ij} \quad (j = 1, 2, \ldots, n)$$

for the ith individual on the jth measure, where the F_i are *common factors*, i.e. factors that are common to some of the original measures, and U_j is a *unique factor* associated with measure j. Usually $m << n$. The coefficients a_{jk} are often referred to as "loadings", while the coefficient of U_j is called the unique factor loading.

Factor analysis is based on correlation coefficients and, thus, on measures of variance and covariance. The process begins with a matrix of correlations, derived from a sample of n values for each variable. In the current study, correlations between heuristics are based on runs on n different instances of each problem class. Then, a factor extraction process is applied, which extracts a set of *uncorrelated* factors which together account for a maximal amount of the variance in the original matrix. In this case, each a_{jk} above is equal to the correlation coefficient between z_j and F_k [4].

3 Methods for Experiments with Random k-SAT Problems

3.1 Factor Analysis

The R system was used for these analyses, downloaded from http://www.r-project.org. The factanal() function was used for the factor analysis. This function uses the maximum likelihood method for finding factors [5]. In addition, varimax rotation was used to simplify the pattern of loadings.

To ensure stability of results, samples of at least 100 problem instances were used. In comparisons of samples of 100 and 1000 problems with the same characteristics, the proportion of variance accounted for by successive factors and the factor loadings were similar. To minimise effects due to the skewed character of performance distributions in this domain, logarithms of the original performance scores were used in the analyses.

3.2 Solvers and Problems

Analyses were made with two separate solvers, written by each of the authors independently (indicated by their initials). The RJW solver was written in Common Lisp and run using Xlisp on a Dell Work Station PWS 330 running at 1.8 GHz. The SB solver was written in C++ and executed under Cygwin on a Dell Latitude D510 running at 1.73 GHz. A difference between the solvers is that in one case (RJW), unit-clause variables are instantiated one by one, with propagation after each instantiation, while for the other (SB) all unit-clause variables are instantiated together, followed by propagation. (In both solvers, each such instantiation is counted as a search node.) Because of this, in the RJW solver a distinction could be made between ‘simple’ unit propagation (which was done in lexical order within the set of unit-clause variables) and instantiation of unit-clause variables associated with the most unit clauses. Because these orderings together give more information about the nature of variable ordering heuristics, they are distinguished in the descriptions and discussions that follow. Since both solvers gave similar results, only data from the RJW solver are presented below.

The present results were based on fixed-length random SAT problems. 3-SAT problems were used, having 50 logical variables and 218 clauses (SATLIB uf50-218 and uuf50-218 problems). These results were replicated in experiments on random k-SAT problems generated using a generator written by B. Selman (mwff).

Later tests included SAT encodings of quasi-group problems based on the encoding of [6] and SAT encodings of k-coloring problems. Despite the marked differences in problem structure, the results for SAT encodings of quasigroup problems were quite similar to those presented here. Although results for k-colouring problems differ from random 3-SAT, patterns of loadings observed with random 3-SAT were still discernible.

3.3 Heuristics and Basic Algorithm

For purposes of comparison and experimental control, the order of selecting values was constant across heuristics. In some cases, branching rules therefore differ somewhat from the original, since some SAT heuristics select literals rather than variables (cf. [2]).

Ten branching rules were used in these experiments, plus an elementary version of the Davis-Putnam algorithm (lex) in which variables are simply chosen according to their lexical order. The first seven heuristics in the list below are dynamic; the last four are static (except for variable selection during unit propagation).

- *mostun*: Choose the variable in most unit clauses first, lexical otherwise.
- *moms*: Choose the variable that participates in the MOst clauses of Minimum Size.
- *JWf*: Jeroslow-Wang rule with fixed value ordering. Choose the variable maximising the sum of values $2^{-len_{c_i}}$, where len_{c_i} is the length of clause i, summed over the clauses that contain this variable.

- *JW2*: Two-sided Jeroslow-Wang rule. Maximise as with the basic Jeroslow-Wang rule but over variables instead of literals.
- *JWf1st*: "First-order probability rule" of Hooker and Vinay. Maximise the sum JW(L) − JW(¬L), an estimate of satisfaction probability [2].
- *fwddg*: Choose the variable with the maximum forward degree, i.e. according to the number of uninstantiated variables that appear in the same clauses.
- *bkwddg*: Choose the variable with the maximum backward degree, i.e. according to the number of instantiated variables that appear in the same clauses.
- *lex*: Select variables according to their lexical ordering in the problem (static).
- *clause*: Select variables in the most clauses first (static).
- *vardg*: Choose the variable with the greatest number of neighbouring variables first(static).
- *width*: Choose the first variable lexically, then choose variables that share the maximum number of clauses with previously selected variables (static).

In all cases, ties were broken lexically. For the RJW solver, unit-clause variables were also instantiated in lexical order. In the initial experiments value ordering was always F,T. The main tests were based on nodes visited during search and runtime. For a given set of problem instances, summary values of these measures varied in a similar manner across the different rules. Both measures produced similar patterns of factor loadings, so for brevity, only analyses based on search nodes are reported.

4 Results for Fixed-Length Random SAT Problems

Table 1 shows the average performance of each heuristic for soluble 3-SAT problems (uf50-218), as well as the factor analysis results for these problems and heuristics, using a two-factor model.

There is considerable similarity between these results and the corresponding factor analysis for CSPs. Heuristics that could be considered to "look ahead" tend to load more highly on Factor 1, while those that seem to rely more on building up contention load more highly on Factor 2. (Note in particular, the results for forward degree, backward degree and width.) It is of interest that heuristics that load heavily on the look-ahead factor tend to perform better on these problems. This is consistent with the simplification hypothesis.

The major differences are related to the inclusion of the in-most-unit-clause heuristic. When this was removed and the analysis rerun with the remaining data, the loading for lex on Factor 2 was 0.602 and that for width was 0.929, and bkwddg 0.974. This suggests that in the original analysis Factor 2 is reflecting different effects than in the CSP analysis. One hypothesis supported by later experiments is that in the present analysis, this factor is related to a lexical ordering of the variables.

For a straightforward experiment like this with CSPs, a lexical ordering does not load highly on either factor. This, however, may be explained by the observation that by interleaving unit propagation we create an ordering that is not strictly lexical. In fact, this procedure would be expected to increase the degree of contention with successive instantiations.

Table 1. Performance of DP with different branching rules

	Search Performance		Factor Analysis		
rule	nodes	time(s)	factor 1	factor 2	unique
mostun	2046	0.17	0.108	**0.992**	0.005
moms	318	0.04	**0.906**	0.175	0.148
JWf	633	0.13	**0.575**	0.158	**0.644**
JW2	433	0.08	**0.954**	0.160	0.064
JWf1st	3658	0.71	0.165	0.128	**0.956**
fwddg	990	0.11	**0.666**	0.133	**0.539**
bkwddg	904	0.10	0.185	**0.595**	**0.611**
lex	2927	0.25	0.107	**0.992**	0.005
clause	620	0.05	**0.873**	0.170	0.210
vardg	719	0.06	**0.702**	0.133	0.489
width	1010	0.11	0.183	**0.568**	**0.644**
cum var			0.353	0.608	

Notes. SATLIB uf50-218 problems. Performance values are means. Based on 1000 problems. Loadings ≥ 0.5 in bold.

There are other differences from the CSP case. This is especially true for the JW heuristics; although these tend to be associated with the “look ahead” factor, there are marked discrepancies, especially for JW1st.

Similar factor structures were found for insoluble problems. Hence, subsequent analysis is restricted to the soluble set. In addition, a similar pattern of loadings was found for samples of 100 drawn from this set, so it was possible to carry out further analyses, which sometimes included simplified heuristics that were relatively inefficient, using the first 100 uf50-218 problem instances.

More extensive analyses based on 3- and 4-factor models saw the backward degree heuristics separate from the simple unit-propagation algorithms (3-factor model), and the variable-degree heuristics separate from the other “look-ahead” heuristics (4-factor model). At the same time, the factors observed with the 2-factor model were still recognized, and for both sets of separated heuristics there were still moderate loadings on the factors that they were originally associated with.

To better understand the nature of these factors, heuristics were added that were simplified by removing unit propagation. In these cases, therefore, the basic heuristic was the sole basis for the variable ordering. The simplified heuristics were the static orderings: max clause, max var-degree, max-width and a straight lexical ordering. For these heuristics, the mean number of search nodes was 814,443, 1,332,537, 456,376, and 35,521,435, respectively.

Results based on the four-factor model are shown in Table 2. In the first place, the factor structure is essentially the same as for the original heuristics alone, using a 4-factor model. As before, there is a separation of the forward and the backward-degree heuristics, reflected in their loadings on Factors 3 and 4. Secondly, the patterns of loadings for “pure” heuristic strategies are similar to the patterns for these heuristics used in combination with unit propagation - in spite of the enormous increase in search effort when unit propagation is omitted.

Table 2. Factor Analysis for Branching Rules for 3-SAT - Pure heuristics included

	factor 1	factor 2	factor 3	factor 4	unique
mostun	0.119	**0.924**	0.346		0.005
moms	**0.886**	0.170	0.130	-0.151	0.146
JWf	**0.591**		0.243	0.114	**0.573**
JW2	**0.938**	0.156		-0.226	0.039
JWf1st			0.158	0.243	**0.914**
fwddg	**0.796**			**0.508**	0.103
bkwddg	0.148	0.386	**0.837**	0.101	0.119
lex	0.123	**0.926**	0.338		0.005
clause	**0.866**	0.162			0.222
vardg	**0.784**			0.541	0.091
width	0.155	0.339	**0.895**	0.114	0.046
clause No UP	**0.694**		0.164	0.111	0.477
vardg No UP	**0.606**			0.447	0.423
width No UP	0.168	0.309	**0.773**	0.123	0.264
lex No UP	0.117	**0.755**	0.233		0.358
cum var	0.332	0.515	0.683	0.748	

Notes. First 100 uf50-218 problem instances. Other notes as in Table 1.

Table 3. Factor Analysis for Branching Rules after 3 lexical choices

	factor 1	factor 2	unique
mostun	0.461	**0.885**	0.005
moms	**0.829**	0.471	0.092
JWf	**0.793**	**0.535**	0.085
JW2	**0.611**	**0.588**	0.280
JWf1st	**0.858**	0.455	0.056
fwddg	**0.839**	0.453	0.091
bkwddg	**0.695**	**0.594**	0.165
lex	0.452	**0.890**	0.005
clause	**0.821**	0.490	0.086
vardg	**0.842**	0.481	0.060
width	**0.575**	**0.706**	0.171
lex No UP	**0.531**	**0.613**	0.343
cum var	0.502	0.880	

Notes. Based on first 100 uf50-218 problem instances.

Of particular significance is the observation that the factor that *mostun* and *lex* load highly on is also strongly related to a simple lexical ordering of the variables (*lex No UP*). The confirms the suggestion of the last section. This factor is, therefore, of limited interest for the study of basic heuristic actions since it reflects a specific arbitrary ordering that coincides with default selections.

A further experimental strategy was employed that consisted of making k lexical choices at the top of the search tree before switching to a given heuristic strategy. In this way, differences related to whether or not initial choices are made lexically are controlled. In these experiments, although the procedure detected unit clauses that occurred at level k or above, unit propagation was not carried out until level $k + 1$. This was done to ensure that the first k choices were indeed lexical regardless of the pattern of clause reduction.

Table 3 gives results for k=3. In this analysis, two factors account for almost 90% of the variance (compared with only 60% in the analysis of Table 1), and all heuristics show at least moderate loadings on both factors. (This indicates that the idiosyncratic behaviour of JWf1st is related to choices at the top of the search tree.) However, in most cases there is still a difference consistent with the factor structure observed in the earlier experiments. This allows us to conclude that the present factors are analogous to the two basic factors in the CSP analyses. Since unit propagation heuristics alone load more heavily on the factor favoured by static width (in contrast to simple lexical ordering), unit propagation seems associated with contention effects.

5 Conclusions

These results indicate that, in spite of complications and some unresolved sources of variation, there are two kinds of variation across problem instances that are related to strategies that "look back" (contention) and those that "look ahead" (propagation-of-effects). There is, therefore, a fundamental similarity to the situation that occurs with variable ordering heuristics for CSPs.

The propagation-of-effects action appears similar to the simplification hypothesis of Hooker and Vinay. However, the present results show that neither of the earlier hypotheses can account for all variability in performance due to branching rules. In the first place, the evidence clearly indicates that there is a second major factor that can be distinguished from propagation-of-effects and, therefore, from simplification. Secondly, there appear to be other significant sources of variation associated with choices near the top of the search tree, which are as yet unspecified.

Given the present state of knowledge about heuristics, it would, therefore, be impossible to characterise the action of these heuristics correctly without the information given by the factor analysis. This shows that this approach can provide information that is critical for the development of a systematic analysis of heuristic action.

References

1. Jeroslow, R., Wang, J.: Solving propositional satisfiability problems. Annals of Mathematics and AI 1, 167–187 (1990)
2. Hooker, J.N., Vinay, V.: Branching rules for satisfiability. Journal of Automated Reasoning 15, 359–383 (1995)
3. Wallace, R.J.: Factor analytic studies of CSP heuristics. In: van Beek, P. (ed.) CP 2005. LNCS, vol. 3709, pp. 712–726. Springer, Heidelberg (2005)
4. Harman, H.H.: Modern Factor Analysis, 2nd edn. University of Chicago, Chicago, London (1967)
5. Lawley, D.N., Maxwell, A.E.: Factor Analysis as a Statistical Method, 2nd edn. Butterworths, London (1971)
6. Zhang, H., Stickel, M.E.: Implementing the Davis-Putnam method. Journal of Automated Reasoning 24, 277–296 (2000)

Hybrid Methods to Select Informative Gene Sets in Microarray Data Classification

Pengyi Yang[1] and Zili Zhang[1,2]

[1] Intelligent Software and Software Engineering Laboratory
Faculty of Computer and Information Science
Southwest University, Chongqing 400715, China
[2] School of Engineering and Information Technology, Deakin University
Geelong, VIC 3217, Australia
zzhang@deakin.edu.au

Abstract. One of the key applications of microarray studies is to select and classify gene expression profiles of cancer and normal subjects. In this study, two hybrid approaches–genetic algorithm with decision tree (GADT) and genetic algorithm with neural network (GANN)–are utilized to select optimal gene sets which contribute to the highest classification accuracy. Two benchmark microarray datasets were tested, and the most significant disease related genes have been identified. Furthermore, the selected gene sets achieved comparably high sample classification accuracy (96.79% and 94.92% in colon cancer dataset, 98.67% and 98.05% in leukemia dataset) compared with those obtained by mRMR algorithm. The study results indicate that these two hybrid methods are able to select disease related genes and improve classification accuracy.

1 Introduction

One popular application of microarray studies is to extract and classify the gene expression profiles between biological types. However, relatively few samples with a large dimension of attribute space and a high signal-to-noise ratio [1] are the nature of microarray data. Thus, the information contained in the gene expression profiles needs to be extracted by robust computational strategies.

Decision tree and artificial neural networks (ANN) are nonlinear classifiers which are suited in microarray data classification. However, decision tree algorithm is deterministic and it only uses the top ranked gene to split the data. This leads to only one tree being created and it may be locally optimal. As to ANN, the challenge is that usually the dataset been analyzed contains so many genes that if all used as inputs, the computation complexity will increase significantly and more noise will be introduced to the process [2].

Many hybrid methods are also used in the microarray data analysis. For instance, Leping Li et al. combined genetic algorithm (GA) and K-Nearest Neighbor (KNN) to select a subset of predictive genes [3]. Keedwell K. and Narayanan A. proposed a neural-genetic approach for gene expression analysis [4]. However, when applying these hybrid approaches to microarray data analysis, there are

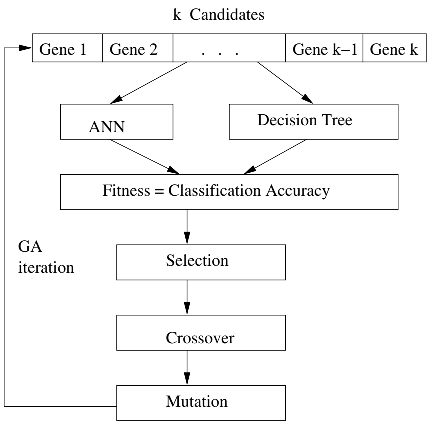


Fig. 1. Hybrid model of GADT & GANN

still some issues like too many genes are included in the result set and gene interaction information is ignored. To this end, we explore two other hybrid approaches namely GADT-hybrid of genetic algorithm and decision tree, and GANN-hybrid of genetic algorithm and artificial neural networks. These two hybrid methods were tested on two benchmark microarray datasets [5,6]. Furthermore, the results are compared with those obtained by mRMR algorithm, which is a feature selection algorithm based on mutual information theory [7].

2 Hybrid Approaches

GA and decision tree hybrid was first designed by J. Bala et al. [8] as a methodological tool for pattern classification. In this work, a standard GA is used to search the space of all possible subsets and invoke decision tree to classify data.

Our GANN algorithm is a variation of Neural-genetic approach proposed by Keedwell K. and Narayanan A. [4]. Different from Neural-genetic algorithm which used a step-function network, a back-propagation NN with a hidden layer was employed in our algorithm to capture nonlinear gene profiles.

The architecture of these two hybrid methods are similar in which they both use GA as the gene selector. As illustrated in Figure 1, GA is utilized to create various gene sets and select important combinations. ANN or decision tree were used as the classifiers in the process, and the fitness function of GA is defined as the classifiers' classification accuracy of different gene combinations.

3 Methods

3.1 Data Preprocessing

Two benchmark microarray datasets, which have been published by Alon et al. [5] and Golub et al. [6] respectively, have been used to evaluate our hybrid approaches. In data preprocessing, the expression values of genes in colon dataset were logarithmically transformed (with base e) and then normalized into 0-1. The tumor tissues were marked as 1 and the normal ones were marked as 0. For leukemia dataset, we adopted the preprocessing method proposed by Dudoit S et al. [9]. Furthermore, every gene was marked with an ID and the chromosome of GA was coded by a string of gene IDs, which specify a candidate gene profile.

3.2 Hybrid Model Construction and Evaluation

The starting population size of GA has been set to 500. The probability of crossover and probability of mutation have been assigned to 0.7 and 0.03 respectively. The single point mutation and crossover were used in genetic operation parts, and the binary tournament selection method was adopted. The termination condition is that GA reaches the 50th generation. For GADT, one of the most popular decision tree package C5.0 [10] was used to build trees and make data classification. A back propagation NN with 2-10 hidden nodes and 2-20 input nodes was utilized as the GANN's fitness calculator to evaluate various gene combinations. 1000 processing epoches, learning rate of 0.1-0.4 and momentum of 0.4 have been used in ANN training and evaluation.

Cross validation and multiple random validation are two common strategies of performance evaluation. In gene sets selection phase, multiple random validation was employed to re-divide data in every generation. In gene sets evaluation phase, 5-fold cross validation was used to evaluate the classification accuracy. The top five frequently selected gene combinations in both GANN and GADT were used to compare with previous studies [1,11] to find overlapped genes.

4 Results

Using GANN, the highest evaluation accuracy 94.92% for colon data and 98.05% for leukemia data are achieved by a 15-gene combination and a 5-gene combination respectively. The gene profiles of these two combinations are (U02020 M26481 T58861 H66786 H40108 X57351 M33210 R36977 L34657 T47645 R80400 H08393 R99907 T57882 U04241) and (M23197 M27891 D88270 D26156 U65011). With GADT, we obtained the highest classification accuracy 96.79% for colon data and 98.67% for leukemia data using two 5-gene combinations respectively, and the gene profiles selected by GADT are (L25851 X12671 M91463 M81840 T67897) and (M29960 M12959 M28827 L35263 X61587).

The results are compared with those obtained by mRMR algorithm [7]. Figure 3 summarized the comparison. With mRMR and ANN, the best results are 87.74% for colon data and 97.82% for leukemia data, using a 5-gene and a 20-gene combinations. When using mRMR with DT, the best results are 87.05% for colon data and 93.05% for leukemia data, with two 2-gene combinations. As can be seen from Figure 2, our hybrid approaches are superior in both cases.

The rank of the 10 highly significant colon tumor and leukemia relevant genes selected by GANN and GADT are shown in Table 1. The genes are ranked

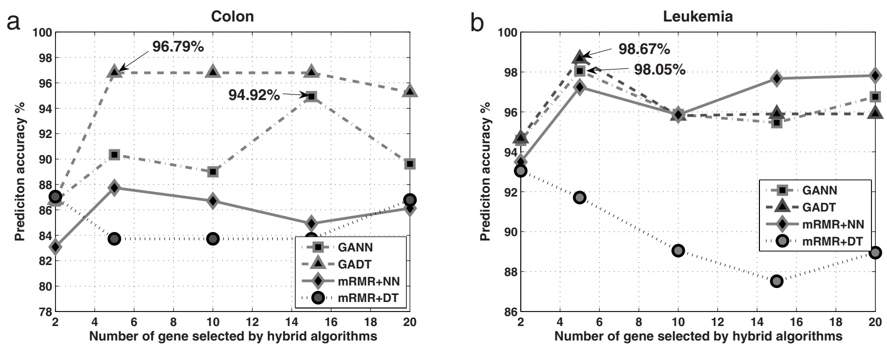


Fig. 2. Evaluation accuracy of the best gene profiles selected by GANN, GADT, mRMR+NN and mRMR+DT. (a) Classification accuracy comparison of colon dataset. (b) Classification accuracy comparison of leukemia dataset.

Table 1. The rank of the top 10 disease associated genes selected by GANN & GADT

GANN				GADT			
Colon	Selec. Freq.	Leukemia	Selec. Freq.	Colon	Selec. Freq.	Leukemia	Selec. Freq.
U14631	0.277	M23197	0.287	X12671	0.600	M27749	0.400
D00860	0.216	M27891	0.224	M91463	0.400	M27891	0.200
H08393	0.195	U46499	0.209	U31248	0.255	U46499	0.200
R36977	0.187	X95735	0.160	H08393	0.200	Y10807	0.200
L05485	0.183	M13690	0.140	T62947	0.200	Z49155	0.200
L27841	0.176	D88270	0.139	M26383	0.200	L07633	0.200
T57882	0.162	M83667	0.135	H11460	0.200	X61755	0.200
R80400	0.162	L09209	0.129	R46502	0.200	M92287	0.200
H66786	0.162	M63138	0.120	T60155	0.200	X95735	0.200
H40108	0.162	U70063	0.120	R99907	0.200	M12959	0.200

Table 2. Genes selected by GANN & GADT that overlapped with studies [1,11]

GANN				GADT			
Colon	Selec. Freq.	Leukemia	Selec. Freq.	Colon	Selec. Freq.	Leukemia	Selec. Freq.
H08393	0.195	M23197	0.287	H08393	0.200	U46499	0.200
T58861	0.139	M27891	0.224	M26383	0.200	M27891	0.200
U30825	0.139	L09209	0.129	T62947	0.200	L09209	0.200
M63391	0.053	U46499	0.209	H87465	0.200	X95735	0.200
Z24727	0.053	X95735	0.160	U37012	0.094	M12959	0.200
R87126	0.053	U70063	0.120	–	–	–	–
H87465	0.033	M83652	0.116	–	–	–	–
T62947	0.027	M12959	0.032	–	–	–	–

by their frequency of selection in GANN and GADT results. Furthermore, the selected colon tumor relevant genes and leukemia relevant genes which are overlapped with previous related studies [1,11] are presented in Table 2. Genes H08393, T62947 from colon dataset and M27891, U46499 from leukemia dataset, which have been considered as the most important disease related factors, have been identified.

5 Conclusion

In this study, two hybrid methods, GANN and GADT are employed to select optimal gene sets in microarray data classification. Appreciably high classification accuracy in two benchmark datasets were achieved. In addition, the most important disease related genes (Table 2) which have been reported by other alternative methods have also been identified by our hybrid methods. The results indicate that GANN and GADT are able to select and optimize informative gene profile in order to improve the classification accuracy of the microarray dataset.

References

1. Li, X., Rao, S.Q., Wang, Y.D., Gong, B.S.: Gene mining: a novel and powerful ensemble decision approach to hunting for disease genes using microarray expression profiling. Nucleic Acids Research 32(9), 2685–2694 (2004)
2. Yang, P.Y., Zhang, Z.L.: A Hybrid Approach to Selecting Susceptible Single Nucleotide Polymorphisms in Age-Related Macular Degeneration Diagnosis. submitted to Artificial Intelligence in Medicine
3. Li, L., Weinberg, C.R., Darden, T.A., Pedersen, L.G.: Gene selection for sample classification based on gene expression data: study of sensitivity to choice of parameters of the GA/KNN method. Bioinformatics 17, 1131–1142 (2001)
4. Keedwell, E., Narayanan, A.: Genetic Algorithms for Gene Expression Analysis. In: Raidl, G.R., Cagnoni, S., Cardalda, J.J.R., Corne, D.W., Gottlieb, J., Guillot, A., Hart, E., Johnson, C.G., Marchiori, E., Meyer, J.-A., Middendorf, M. (eds.) EvoBIO 2003. LNCS, vol. 2611, pp. 76–86. Springer, Heidelberg (2003)
5. Alon, U., Barkai, N., Notterman, D.A., Gish, K., Ybarra, S., Mack, D., Levine, A.J.: Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. Proc. Natl. Acad. Sci. 96, 6745–6750 (1999)
6. Golub, T.R., Slonim, D.K., Tamayo, P., Huard, C., Gaasenbeek, M., Mesirov, J.P., Coller, H., Loh, M.L., Downing, J.R., Caligiuri, M.A., Bloomfield, C.D., Lander, E.S.: Molecular Classification of Cancer: Class Discovery and Class Prediction by Gene Expression Monitoring. Science 286, 531–537 (1999)
7. Peng, H., Long, F., Ding, C.: Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy. IEEE transactions on pattern analysis and machine intelligence 27, 1226–1238 (2005)
8. Bala, J., Huang, J., Vafaie, H., DeJong, K., Wechsler, H.: Hybrid Learning Using Genetic Algorithms and Decision Trees for Pattern Classification. In: IJCAI conference, Montreal (August 19-25, 1995)
9. Dudoit, S., Fridlyand, J., Speed, T.P.: Comparison of Discrimination Methods for the Classification of Tumors Using Gene Expression Data. Journal of the the American Statistical Association 97, 77–87 (2002)
10. Quinlan, J.R.: C4.5: Programs for machine learning. Morgan Kaufmann Publishers, San Francisco (1993)
11. Cho, S.B., Won, H.H.: Machine Learning in DNA Microarray Analysis for Cancer Classification. In: Conferences in Research and Practice in Information Technology, vol. 19 (2003)

An EM Algorithm for Independent Component Analysis Using an AR-GGD Source Model

Yumin Yang[1,2,3], Chonghui Guo[1], and Zunquan Xia[2]

[1] Institute of Systems Engineering, Dalian University of Technology, Dalian 116024, P.R. China
[2] Department of Applied Mathematics, Dalian University of Technology, Dalian 116024, P.R. China
[3] Department of Mathematics, Anshan Normal University, Anshan 114001, P.R. China
guochonghui@tsinghua.org.cn

Abstract. A maximum likelihood blind source separation algorithm is developed. The temporal dependencies are explained by assuming that each source is an autoregressive (AR) process and the distribution of the associated i.i.d. innovations process is described by generalized Gaussian distribution (GGD), which can fit a broader range of statistical distributions by varying the value of the steepness parameter α. Unlike most maximum likelihood methods the proposed algorithm takes into account both spatial and temporal information. Optimization is performed using the Expectation-Maximization method, and the source model is learned alone with the demixing parameters.

1 Introduction

Blind source separation (BSS) is typically performed in a setting where the observed signals are instantaneous noise-free linear superpositions of underlying hidden source signals. Let us denote the m source signals by $s_1(t), \cdots, s_m(t)$, and the observed signals by $x_1(t), \cdots, x_m(t)$, where t is the time index. Let $\mathbf{A} \in \mathbb{R}^{m \times m}$ is the mixing matrix. Now the mixing can be expressed as

$$\mathbf{x}(t) = \mathbf{A}\mathbf{s}(t). \tag{1}$$

The problem of BSS is to estimate both the source signals $\mathbf{s}(t)$ and the mixing matrix $\mathbf{A}$ (or its inverse $\mathbf{W} = [\mathbf{w}_1, \mathbf{w}_2, \dots, \mathbf{w}_n]^T$, i.e., $\mathbf{W} = \mathbf{A}^{-1}$, which is called the separation matrix), based on the observations of $\mathbf{x}(t)$ alone [1,9].

In this paper, the temporal dependencies are explained by assuming that each source is an AR process and the distribution of the associated i.i.d. innovations process is described by generalized Gaussian distribution [2,3]. The approach presented here, differs from most ML methods [4-6] in two important ways. First, the proposed criterion makes use of both the spatial and temporal structure of the sources. Second, instead of assuming the target distributions are known, the proposed method learns the target distributions directly from the observations.

2 Generalized Gaussian Distributions

The generalized Gaussian distribution is used to model distributions that deviate from the standard normal. The general form of this distribution is

$$P(n|\mu,\sigma,\alpha) = \frac{\omega(\alpha)}{\sigma}\exp\Big[-c(\alpha)\Big|\frac{n-\mu}{\sigma}\Big|^{2/(1+\alpha)}\Big], -\infty < n < \infty, \alpha > -1 \quad (2)$$

where n is a stochastic variable, and

$$\omega(\alpha) = \frac{\Gamma\Big(\frac{3}{2}(1+\alpha)\Big)^{1/2}}{(1+\alpha)\Gamma\Big(\frac{1}{2}(1+\alpha)\Big)^{3/2}}, c(\alpha) = \Big[\frac{\Gamma\Big(\frac{3}{2}(1+\alpha)\Big)}{\Gamma\Big(\frac{1}{2}(1+\alpha)\Big)}\Big]^{1/(1+\alpha)}, \quad (3)$$

$$\Gamma(x) = \int_0^\infty t^{x-1}e^{-t}\,\mathrm{d}t, x > 0. \quad (4)$$

When $\mu = 0$ and $\sigma = 1$, the generalized Gaussian distribution then becomes a function which only has a single parameter α

$$P(n|\alpha) = \omega(\alpha)\exp\Big[-c(\alpha)\big|n\big|^{2/(1+\alpha)}\Big]. \quad (5)$$

The parameter α is a measure of steepness which controls the distribution's deviation from the standard normal. Fig.1 shows examples of the generalized densities for various value of α, with zero mean and unit variance.

3 Likelihood of Mixing Matrix

We model each source signal by an autoregressive model

$$s_i(t) = \sum_{\tau>0}\beta_i^{\tau}s_i(t-\tau) + n_i(t), i = 1, 2, \cdots, m. \quad (6)$$

where the innovation $n_i(t)$ is assumed to have zero-mean with a density function $q_i(n_i(t))$. In addition, innovation sequences $\{n_i(t)\}$ are assumed to be mutually independent white sequences. Without loss of generality, we can derive the likelihood of the model when the source is AR(1) model. Let us consider latent variables $s_i(t)$ over on N-point time block.

$$\mathbf{s}_i = \big(s_i(0), \cdots, s_i(N-1)\big)^T. \quad (7)$$

Then the joint probability density function of $\mathbf{s}_i$ can be written as

$$p_i(\mathbf{s}_i) = \prod_{t=0}^{N-1} p_i\big(s_i(t)|s_i(t-1)\big) = \prod_{t=0}^{N-1} q_i\big(n_i(t)\big) = \prod_{t=0}^{N-1} q_i\big(\tilde{y}_i(t)\big), \quad (8)$$

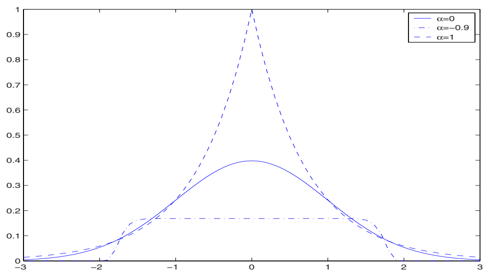


Fig. 1. Generalized Gaussian distributions for various values of α. The larger the value of α is, the steeper the distribution is.

where $s_i(t) = 0$ for $t < 0$, $\tilde{y}_i(t) = s_i(t) - \hat{\beta}_i s_i(t-1)$. Then the log-likelihood is expressed as

$$\frac{1}{N}\log(\mathbf{W}) = \log|\det \mathbf{W}| + \frac{1}{N}\sum_{t=0}^{N-1}\sum_{i=1}^{m}\log q_i\big(\tilde{y}_i(t)\big). \tag{9}$$

Noting that the ensuing algorithm easily becomes unstable, we prewhiten the data and constrain $\mathbf{W}$ to be orthogonal [1] and the estimate of latent variables is denoted by $\mathbf{y}(t) = \mathbf{W}\mathbf{x}(t)$. So the log-likelihood is simplified to

$$J(\mathbf{W}) \approx \sum_{i=1}^{m}\hat{E}\Big\{\log q_i\Big(\mathbf{w}_i^T\big(\mathbf{x}(t) - \sum_{\tau>0}\hat{\beta}_i^{\tau}(\mathbf{w}_i)\mathbf{x}(t-\tau)\big)\Big)\Big\}, \tag{10}$$

where $\tilde{y}_i = y_i(t) - \sum_{\tau>0}\hat{\beta}_i^{\tau}(\mathbf{w}_i)y_i(t-\tau)$.

4 EM Algorithm for AR-GGD

In this section we present an EM algorithm for inferring estimated innovations parameters α from the data and extracting independent sources. The prior $q(n_i(t))$ hereby can be assumed to be a generalized Gaussian distribution as the form in the equation (5) with parameter α_i, so (10) can be written as

$$J(\mathbf{W}, \alpha_i) \approx \sum_{i=1}^{m}\hat{E}\Big\{\log\omega(\alpha_i) - c(\alpha_i)\Big|\tilde{y}_i(t)\Big|^{2/(1+\alpha_i)}\Big\}. \tag{11}$$

We can assume that the estimations of the $\hat{\beta}_i^{\tau}(\mathbf{w}_i)$ are decoupled from the estimation of the $\mathbf{w}_i$ and $\hat{\alpha}_i$. The learning rule for α_i and $\mathbf{w}_i$ may be obtained via gradient ascent on $J(\mathbf{W}, \alpha_i)$ respectively.

$$\begin{aligned}\nabla\alpha_i J = \hat{E}\Big\{&\Big[\frac{3}{4}(\frac{u'}{u} - \frac{v'}{v}) - \frac{1}{1+\alpha_i}\Big] - \omega'\Big(\Big|\tilde{y}_i(t)\Big|^{2/(1+\alpha_i)}\Big) \\ &+ \frac{2\omega}{(1+\alpha_i)^2}\ln\Big|\tilde{y}_i(t)\Big|\Big|\tilde{y}_i(t)\Big|^{2/(1+\alpha_i)}\Big\},\end{aligned} \tag{12}$$

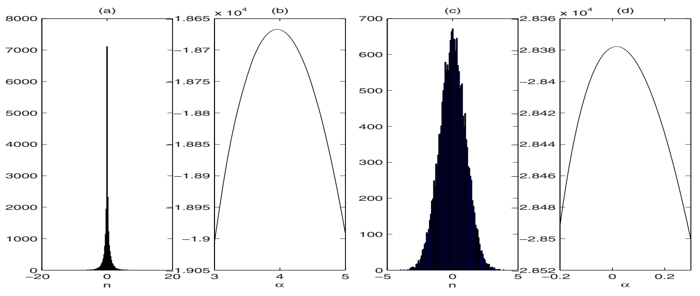


Fig. 2. (a) and (c) are the histograms of two random data (20000 samples) drawn from two different generalized Gaussian distributions, with $\alpha = 4$ and $\alpha = 0$ respectively. (b) and (d)the maximum estimators of these two log-likelihood functions are $\hat{\alpha} = 3.7486$ and$\hat{\alpha} = 0.0191$ respectively.

$$u = \Gamma\Big(\frac{3}{2}(1+\alpha_i)\Big), u' = \Gamma'\Big(\frac{3}{2}(1+\alpha_i)\Big), \tag{13}$$

$$v = \Gamma\Big(\frac{1}{2}(1+\alpha_i)\Big), v' = \Gamma'\Big(\frac{1}{2}(1+\alpha_i)\Big), \tag{14}$$

$$\omega' = \Bigg(\Bigg[\frac{\Gamma\Big(\frac{3}{2}(1+\alpha_i)\Big)}{\Gamma\Big(\frac{1}{2}(1+\alpha_i)\Big)}\Bigg]^{1/(1+\alpha_i)}\Bigg)' = -\omega \cdot \Big(\frac{\ln\left(\frac{u}{v}\right)}{(1+\alpha_i)^2} - \frac{(3u'v - uv')}{2(1+\alpha_i)\cdot uv}\Big). \tag{15}$$

$$\nabla \mathbf{w}_i J = -\hat{E}\Big\{\frac{2c(\alpha_i)}{1+\alpha_i}\cdot \mathrm{sign}\big(\tilde{\mathrm{y}}_\mathrm{i}(\mathrm{t})\big)\cdot \Big|\tilde{\mathrm{y}}_\mathrm{i}(\mathrm{t})\Big|^{\frac{(1-\alpha_\mathrm{i})}{(1+\alpha_\mathrm{i})}} \cdot \big(\mathbf{x}(\mathrm{t}) - \sum_{\tau>0} \hat{\beta}_\mathrm{i}^{\tau}(\mathbf{w}_\mathrm{i})\mathbf{x}(\mathrm{t}-\tau)\big)\Big\}. \tag{16}$$

To improve the convergence, it is quite useful to perform a projection of the gradient on the tangent surface of the set of orthogonal matrices [7-8]. This means replacing the gradient $\nabla_{\mathbf{W}}$ by

$$\nabla^{ort}_{\mathbf{W}} = \nabla_{\mathbf{W}} - \mathbf{W}\nabla^{T}_{\mathbf{W}} J\mathbf{W},\ \mathbf{W} \leftarrow (\mathbf{W}\mathbf{W}^T)^{-1/2}\mathbf{W}. \tag{17}$$

5 Simulations

We created five source signals using an AR(1) model. Signals 1, 2 and 3 were created with supergaussian innovations and signals 4 and 5 with gaussian innovations; all innovations were drawn from generalized Gaussian distributions with various α which were designed to be 4, 3, 2, 0 and 0, and all innovations had unit variance. The signals were mixed by a 5×5 random mixing matrix (denote by $\mathbf{A}$) as in ICA. Sample size T was 20000. The algorithm always converge after about 500 iterations depending on the initial conditions. Table 1 shows the inferred values for parameters α, the Kullback-Leibler divergence between the inferred density and actual innovation distribution, and the kurtosis of the inferred innovations.

Table 1. Estimated α and KL-divergence

Innovations	$n_1(t)$	$n_2(t)$	$n_3(t)$	$n_4(t)$	$n_5(t)$
Estimated α	3.7486	2.6599	1.9558	0.0191	0.0162
KL-distance	0.0083	0.0141	0.0095	0.0346	0.0252
Kurtosis	10.0206	5.1078	4.1405	0.0120	0.0121

6 Conclusions

This paper develops a BSS algorithm that is based on maximizing the data likelihood where each source is assumed to be an AR process and the innovations are described using a generalized Gaussian distributions. The EM algorithm is used as the optimization method and the algorithm performs well through simulations. Future work will focus on incorporating noise directly into the model in a manner similar to that used for independent component analysis method.

Acknowledgments. The work was supported by NSFC(10571018).

References

1. Hyvärinen, A., Karhunen, J., Oja, E.: Independent Component Analysis. John Wiley & Sons, New York (2001)
2. Lee, T.W., Lewicki, M.S.: The Generalized Gaussian Mixture Model Using ICA. In: International Workshop on Independent Component Analysis, Helsinki, pp. 239–244 (2000)
3. Lee, T.W., Lewicki, M.S., Sejnowski, T.J.: ICA Mixture Models for Unsupervised Classification of Non-Gaussian Sources and Automatic Context Switching in Blind Signal Separation. IEEE Transactions on Pattern Analysis and Machine Intelligence 22(10), 1078–1089 (2000)
4. Moulines, E., Cardoso, J.F., Gassiat, E.: Maximum Likelihood for blind source separation and deconvolution of noisy signals using mixture models. In: Intl. Conf. on Acoustics, Speech, and Signal Proc., vol. 5, pp. 3617–3620 (1997)
5. Pham, D.T., Garat, P.: Blind separation of mixture of independent sources through a quasi-maximum likelihood approach. IEEE Trans. on Signal Proc. 45, 1712–1725 (1997)
6. Pearlmutter, B.A., Parra, L.C.: Maximum likelihood blind source separation: A context-sensitive generalization of ICA. Advances in Neural Information Proc. Systems 9, 613–619 (1996)
7. Edelman, A., Arias, T.A., Smith, S.T.: The gemometry of algorithms with orthogonality constraints. SIAM J. Matrix Anal. Appl. 20(2), 303–353 (1998)
8. Xu, L.: One-bit-matching theorem for ICA, convex-concave programming on polyhedral set, and distribution approximation for combinatorics. Neural comput. 19(2), 546–569 (2007)
9. Jutten, C., Hérault, J.: Blind separation of sources, Part I: an adaptive algorithm based on neuromimetic architecture. Signal Process 24, 1–10 (1991)

Bagging Support Vector Machine for Classification of SELDI-ToF Mass Spectra of Ovarian Cancer Serum Samples

Bailing Zhang[1], Tuan D. Pham[2], and Yanchun Zhang[1]

[1] School of Computer Science and Mathematics
Victoria University, VIC 3011, Australia
bailing.zhang@vu.edu.au
[2] School of Information Technology
James Cook University, QLD 4811, Australia

Abstract. There has been much progresses recently about the identification of diagnostic proteomic signatures for different human cancers using surface-enhanced laser desorption ionization time-of-flight (SELDI-TOF) mass spectrometry. To identify proteomic patterns in serum to discriminate cancer patients from normal individuals, many classification methods have been experimented, often with successful results. Most of these earlier studies, however, are based on the direct application of original mass spectra, together with dimension reduction methods like PCA or feature selection methods like T-tests. Because only the peaks of MS data correspond to potential biomarkers, it is important to study classification methods using the detected peaks. This paper investigates ovarian cancer identification from the detected MS peaks by applying Bagging Support Vector Machine as a special strategy of bootstrap aggregating (Bagging). In bagging SVM, each individual SVM is trained independently, using randomly chosen training samples via a bootstrap technique. The trained individual SVMs are aggregated to make a collective decision in an appropriate way, for example, the majority voting. Bagged SVM demonstrated a 94% accuracy with 95% sensitivity and 92% specificity respectively by using the detected peaks. The efficiency can be further improved by applying PCA to reduce the dimension.

1 Introduction

In the last decade, mass spectrometry (MS) based technologies have impressively emerged as the method of primary choice toward the study of proteins, which is the main theme of proteomics as an integral part of the process of understanding biological functions and structures at the protein level. High-throughput proteomics techniques based on mass spectrometry hold great promise for bulk analysis of biological material to accurately identify bacteria. In the realm of clinical proteomics, they have been rapidly adapted to biomedical approaches, for example, cancer prediction on the basis of peptide/protein intensities [1-5, 6-7,13]. The research is very important because earlier detection of cancer is a critical issue for improving patient survival rates.

The aim of mass spectrometry is to provide information about the amount of all specific molecules in a probe, where each molecule is characterised by its mass. Mass spectrometry measures two properties of ion mixtures in the gas phase under the vacuum environment: the mass/charge ratio (m/z) of ionized proteins in the mixture and the number of ions present at different m/z values. The m/z values and the intensity measurement that indicates an (relative) abundance of the particle are represented on the horizontal axis and vertical axis, respectively.

While MS is increasingly used for protein profiles, significant challenges have arisen with regard to analyzing the data sets. The most addressed issue is the biomarkers detection. A biomarker is an identified protein, which is correlated with the state of a particular disease or condition [15]. For clinical applications, biomarkers discovery will specify which chemicals are involved in the disease process or which bio-molecules are affected by the disease process. This requires identification of those peaks in the MS spectrum. Hence extracting these relevant peaks should be one of the fundamental steps in preparing mass spectrometry data for any further analysis [7-8, 15]. Though some earlier studies applied classification algorithms directly based on the raw spectra, it will be of much greater interest for causality interpretability to build an efficient classification model based on the identified biologically relevant chemicals.

A common approach for spectral peaks detection involves feature extraction and quantification, in which the peaks are located and quantified. This requires dealing with several modeling issues, including calibration of the spectra, baseline correction, smoothing and normalization. Among the proposed methods, wavelet de-noising procedure has shown some advantages [15], by which the wavelet coefficients are shrunk toward zero so that only a limited number will be reserved for reconstructing the original signal.

Classification is often the ultimate step in MS data analysis. Extensive works on different classification methods for cancer identification with MS data have been published [3,6,9,13]. In this paper, we first compared several commonly applied classification algorithms based the selected peaks and then proposed to apply bagging strategy to enhance their performance. Bagging, which stands for bootstrap aggregation, is a way of manipulating training data for ensemble methods [11]. By bagging, unstable base classifiers whose behavior could be significantly changed by small fluctuations in the training dataset, are more likely to be stabilized after being trained with different input data and combined afterwards. By bagging SVMs, several SVMs are trained independently via a bootstrap method and then they are aggregated via an appropriate combination technique. Our experiments demonstrated that Bagging Support Vector Machine (SVM) is the best among the compared algorithms for SELDI-TOF data analysis.

This paper is organized as follows. The principle of peaks detection based on undecimated wavelet transform is briefly outlined in next section, followed by short introduction of Support Vector Machine (SVM) and Bagging SVM Ensemble. Section 4 delivered detailed experiments for the MS spectra classifications based on the detected peaks.

2 Pre-processing and Peaks Detetions for Mass Spectrometry Data

For cancer classification, the protein samples from cancer patients and non-cancer patients or from different cancer stages are analyzed through mass spectrometry instruments and the mass spectrometry patterns are then used to built a diagnostic classifier. However, protein mass spectra are inherently noisy and require substantial pre-processing before analysis. MS peak selection step is especially important in the pre-processing. Most peaks in the spectrum are associated with proteins or peptides and constitute important features. Many methods have been proposed so far for peak detection, among them wavelet shrinkage based method [15] has become popular. Under wavelet shrinkage principle, the wavelets are exploited as smoothing operators. Small wavelets coefficients that are not distinguishable from pure noise are eliminated by thresholding. The remained coefficients will be used for a better reconstruction of the signal.

In MS peak detection, wavelet shrinkage is often based on the so-called *undecimated discrete wavelet transform (UDWT)*, because there is a well-known problem with the usual DWT of being lack of translation invariance. This means that statistical estimators that rely on DWT are sensitive to choice of origin. The UDWT overcomes the problem by using a redundant basis in which there is the same number of coefficients at each level. We simply apply appropriate high and low pass filters to the data at each level to produce two sequences at the next level without decimation.

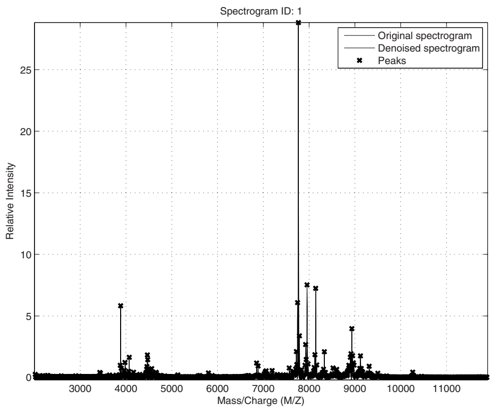


Fig. 1. Plots of SELDI-ToF sample with selected peaks based on wavelet

There are a number of options for thresholding. In the simple hard thresholding, all coefficients less than a threshold value are set to zero, while all coefficients greater than the threshold remain unchanged. The threshold is the product of a thresholding parameter η and a robust estimate of the noise, the median absolute deviation (MAD) divided by 0.67. Because most signals can be represented by a small number of wavelet coefficients, yet white noise is distributed equally among all wavelet coefficients; this approach denoises with minimal attenuation of the features of the signal.

3 Support Vector Machine (SVM) and Bagging SVM Ensemble

SVM is based on the structural risk minimization principle, which determines the classification decision function by minimizing the empirical risk:

$$R = \frac{1}{l}\sum_{i=1}^{l} \left| f(\mathbf{x}_\mathrm{i} - \mathrm{y}_\mathrm{i}) \right|, \tag{1}$$

where l and f represent the size of examples and the classification decision function respectively. Usually, the classification decision function in the linearly separable problem is represented by

$$f = \mathrm{sign}(\mathbf{w} \cdot \mathbf{x} + \mathrm{b}) \tag{2}$$

In SVM, the optimal separating hyperplane is determined by giving the largest margin of separation between different classes. This optimal hyperplane bisects the shortest line between the convex hulls of the two classes. The optimal hyperplane is required to satisfy the following constrained mimimization, as

$$\begin{aligned} &\text{Min:} \quad \frac{1}{2}\mathbf{w}^\mathrm{T}\mathbf{w}, \\ &y_i(\mathbf{w} \cdot \mathbf{x}_\mathrm{i} + \mathrm{b}) \geq 1 \end{aligned} \tag{3}$$

Many studies have shown that aggregating the prediction of multiple classifiers can improve the performance achieved by a single classifier [12]. Bagging [11] is a “bootstrap” ensemble method that creates individuals for its ensemble by training each classifier on a random redistribution of the training set. In [10] it has been shown that reducing the error of single SVMs would reduce the error of bagging of SVMs.

Given a common training set T, k training sets T_i as bootstrap replica (resampled with replacement) are generated. These bootstraped samples generate different training sets for the base SVM classifiers. On average, 63.2% of the total original training examples are retained in each bootstrapped set [9]. This procedure seeks to make training sets as different as possible in order to obtain higher improvement performance on the aggregation result.

Let $f(x, S_1^*)$, ..., $f(x, S_B*)$ be B SVMs trained on $S_1^*, \ldots, S_B^*$ that employ the same kernel function. For each test sample z_j, $\hat{y}_i = f(z_j, S_i^*)$, $i = 1, 2, \ldots, B$, is the estimated class label by each of the B SVMs. With the trained classifiers, their responses can be aggregated by majority voting or sum of responses.

4 Experiments on High Resolution Ovarian Mass Spectrometry Data

The publicly available High Resolution Ovarian MS dataset was downloaded from the website in [16] and used in our research. The dataset contains high resolution mass spectrometry data from a hybrid quadrupole time-of-flight spectrometer and consist of 121 ovarian cancer samples and 95 control samples. Description and analysis of these data are available in [7]. A characteristic of the data is that a number of peaks can be observed. Each peak represents a singly charged positive ion originating from a protein in the sample. We applied the UDWT peak detection methods, yielding 1500 peak list for each sample, as indicated in Fig. 1.

In our first experiment, we applied bagging strategy to several classification methods using the detected peaks, including kNN, decision tree and SVM, based on random divisions of the dataset into a training set and a test set. Each classifier's training set is generated by randomly drawing with replacement from original training set. For the decision tree classifier CART, the purity was adopted as the binary splitting criterion. For k-NN classifier, we simply chosen $k = 1$. For multiple layer perceptron (MLP), we experimented with a three-layer network with the number of inputs the same as the number of features, one hidden layer with 5 units and a single linear unit representing the class label. The support vector machine classifier was optimized by quadratic programming and we chosen the polynomials basis with constant $C = 1$. Each individual classifier in the ensemble is generated with a different random sampling of the training set. For the aggregation in the bagging, we adopted double-layer hierarchical combining, which means that the responses of a classifier's ensemble are aggregated by a same classifier.

The first holdout experiment results are presented in the top of Figure 2, which are from 20 random splitting of training/testing and the classification results on the testing set were recorded and averaged. The results show that (1) for the four classifiers implemented, bagging all brings improvement in classification accuracy, and (2) among the different bagged classifiers, bagged SVM is the best, which gives 94% overall accuracy with 95% sensitivity and 92% specificity. Sensitivity is the number of detected cancer pantients over the total number of patients, while specificity the number of detected controls over the total number of controls. The Figure 2(b) gives the 10-fold cross-validation results in the same experiment setting, with consistent, similar results.

The bagging SVM achieved not only the best average performance among all the different versions of the training sets, but it was also the one whose performance exhibited the smallest variance. The specificity from the bagged SVM is the highest among the tested classifiers with 92% and 95% from holdout and 10-fold CV respectively. The advantages of bagging for other classifiers are not consistent, however. While the overall accuracies all improved from bagging, the sensitivity become worse in the case of kNN. The performance of SVM is stable over the different versions of the datasets is a further indication that for MS data analysis we must closely examine more than one variables in the same time in order to make a classification of a sample.

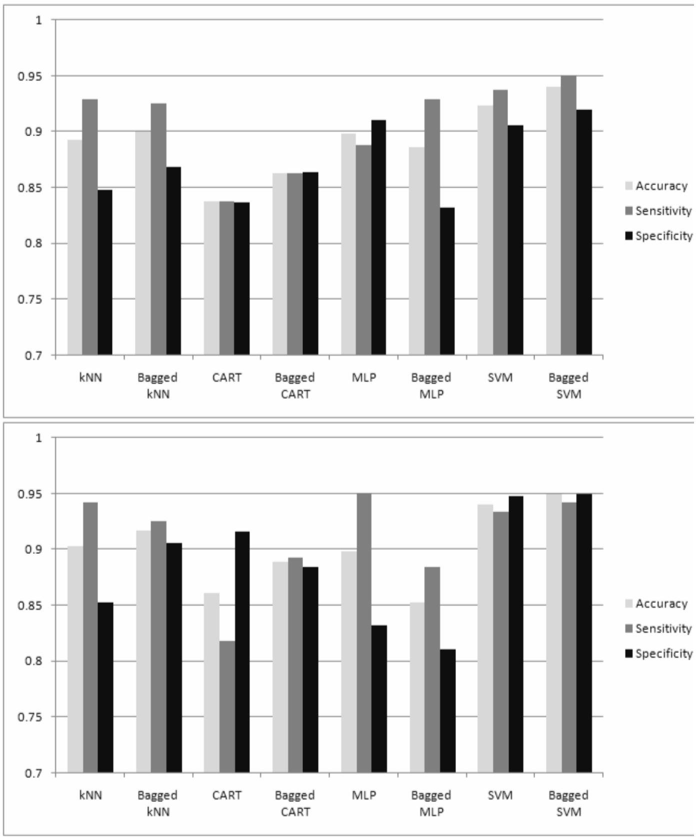


Fig. 2. Bar plots of classification performances from the different classifiers. (a) results from holdout experiment with 80% of data were used for training while the remaining for testing. The results were from the average of 20 tests. (B) results from 10-fold cross-validation.

5 Conclusion

The proteomic data obtained from mass spectrometry have attracted great interest for possible clinical applications. The peak detection step in the preprocessing of MS output is very crucial in finding biomarkers with diseases, which unfortunately has often been separated from classification procedures in the previous studies. Our preliminary experiments with the ovarian cancer MS data showed that the bagging SVM with UDWT-based peaks detection is a promising and practical scheme for the detection of cancer. Further dimension reduction for the peaks features have also proven to be efficient in the classification performance. The bagging support vector machine classification scheme based on the detected peaks worked very well and could be an effective way to make a highly reliable prognostication of patients possibly suffering from cancer.

References

1. Seibert, V., Ebert, M., Buschmann, T.: Advances in clinical cancer proteomics: SELDI-ToF-mass spectrometry and biomarker discovery. Briefings in Functional Genomics and Proteomics 4, 16–26 (2005)
2. Wu, B., Abbott, T., Fishman, D., McMurray, W., Mor, G., Stone, K., Ward, D., Williams, K., Zhao, H.: Comparison of statistical methods for classification of ovarian cancer using mass spectrometry data. Bioinformatics 19, 1636–1643 (2003)
3. Yu, J., Ongarello, S., Fiedler, R., Chen, X., Toffolo, G., Cobelli, C., Trajanoski, Z.: Ovarian cancer identification based on dimensionality reduction for high-throughput mass spectrometry data. Bioinformatics 21, 2200–2209 (2005)
4. Zhang, X., Wei, D., Yap, Y., Li, L., Guo, S., Chen, F.: Mass spectrometry-based "omics" technologies in cancer diagnostics. Mass Spectrometry Reviews 26, 403–431 (2007)
5. Lilien, R.H., Farid, H., Donald, B.R.: Probabilistic Disease Classification of Expression-Dependent Proteomic Data from Mass Spectrometry of Human Serum. Journal of Computational Biology 10, 925–946 (2003)
6. Conrads, T.P., et al.: High-resolution serum proteomic features for ovarian detection. Endocrine-Related Cancer 11, 163–178 (2004)
7. Petricoin, E.F., et al.: Use of proteomic patterns in serum to identify ovarian cancer. The Lancet 359, 572–577 (2002)
8. Diamandis, E.: Mass spectrometry as a diagnostic and a cancer biomarker discovery tool: opportunities and potential limitations. Mol. Cell Proteomics 3, 367–378 (2004)
9. Vlahou, A., Schorge, J., Gregory, B., Coleman, R.: Diagnosis of Ovarian Cancer Using Decision Tree Classification of Mass Spectral Data. Journal of Biomedicine and Biotechnology 2003, 308–314 (2003)
10. Valentini, G., Muselli, M., Ru1no, F.: Cancer recognition with bagged ensembles of Support Vector Machines. Neurocomputing 56, 461–466 (2004)
11. Breiman, L.: Bagging predictors. Machine Learning 24, 123–140 (1996)
12. Sutton, C.D.: Classification and Regression Trees, Bagging, and Boosting. Handbook of Statistics 24, 14–41 (2004)
13. Cruz, J., Wishart, D.: Applications of Machine Learning in Cancer Prediction and Prognosis. Cancer Informatics 2, 59–78 (2006)
14. Kim, H., Pang, S., Je, H., Kim, D., Bang, S.: Pattern Classification Using Support Vector Machine Ensemble. In: Proc. of ICPR 2002, pp. 20160–20163 (2002)
15. Morris, J., Coombes, K., Koomen, J., Baggerly, K., Kobayashi, R.: Feature extraction and quantification for mass spectrometry in biomedical applications using the mean spectrum. Bioinformatics 21, 1764–1775 (2005)
16. Clinical proteomics program databank, http://ncifdaproteomics.com/ppatterns.php

Class Association Rule Mining with Multiple Imbalanced Attributes

Huaifeng Zhang, Yanchang Zhao, Longbing Cao, and Chengqi Zhang

Faculty of IT, University of Technology, Sydney, Australia
PO Box 123, Broadway, 2007, NSW, Australia
{hfzhang, yczhao, lbcao, chengqi}@it.uts.edu.au

Abstract. In this paper, we propose a novel framework to deal with data imbalance in class association rule mining. In each class association rule, the right-hand is a target class while the left-hand may contain one or more attributes. This framework is focused on the multiple imbalanced attributes on the left-hand. In the proposed framework, the rules with and without imbalanced attributes are processed in parallel. The rules without imbalanced attributes are mined through standard algorithm while the rules with imbalanced attributes are mined based on new defined measurements. Through simple transformation, these measurements can be in a uniform space so that only a few parameters need to be specified by user. In the case study, the proposed algorithm is applied into social security field. Although some attributes are severely imbalanced, the rules with minority of the imbalanced attributes have been mined efficiently.

1 Introduction

Data imbalance is often encountered in data mining especially classification and association rule mining. As integration of classification and association rule, class association rule [3] always suffers from data imbalance problem. In class association rule, the right-hand side is a predefined target class while the left-hand side can be single or multiple attributes. Data imbalance on either side of the class association rule can cause severe problem.

Recently, there are some researchers working on the data imbalance in class association rule mining. In 2003, Gu et al. [2] proposed an algorithm to deal with imbalanced class distribution in association rule mining. They defined a set of criteria to measure the interestingness of the association rules. Arunasalam and Chawla [1] studied the data imbalance in association rule mining. These algorithms are focused on the data imbalance of target class to improve the performance of so-called associative classifier [3].

In 1999, Liu et al. proposed MSApriori algorithm [4] to deal with rare item problem. Rare item problem is essentially the data imbalance problem on transactional dataset. In MSApriori, the authors defined Minimum Item Support (MIS) to apply multiple minimum supports in the algorithm. However, MIS has to be assigned to every item by user. Moreover, the discovered rules vary depending on the MIS values. Yun et al. [5] proposed an algorithm to mine association rule on significant rare data. In their algorithm, a number of supports also have to be specified by user. The performance of the algorithm heavily depends on the specified supports.

In this paper, we propose a novel algorithm to mine class association rules on the dataset with multiple imbalanced attributes. The rules with and without imbalanced attributes are processed in parallel. With new defined interestingness measurements and simple transformation, the rules can be post-processed in a uniform space. In the case study, this algorithm is applied into social security area. Many more rules with minorities of the imbalanced attributes have been mined, which is very interesting to the business line.

The paper is organized as follows. Section 2 introduces class association rule. Section 3 proposes the outline and the new measurements of proposed algorithm. Section 4 presents a case study. Section 5 is the conclusion. The last section is the acknowledgement.

2 Class Association Rules

Let T be a set of tuples. Each tuple follows the schema $(A_1, A_2, \ldots, A_N, A_C)$, in which $(A_1, A_2, \ldots, A_N)$ are N attributes while A_C is a special attribute, the target class. The attributes may be either categorical or continuous ones. For continuous attributes, the value range is discretized into intervals. For the convenience of description, we call an attribute-value pair an *item*. Suppose itemset $U \subseteq A$, A is the itemset of any items with attributes $(A_1, A_2, \ldots, A_N)$, c is 1-itemset of class attribute, a class association rule can be represented as

$$U \Rightarrow c$$

Here, U may contain a single item or multiple items.

In this paper, we represent the class association rules as

$$X + I \Rightarrow c$$

where, $X \subseteq A$ is the itemset of balanced attributes while $I \subseteq A$ is the itemset of imbalanced attributes.

3 Mining Rules with Imbalanced Attributes

Outline of the Algorithm. In our algorithm, association rule mining is done through two parallel parts. In one part, no imbalanced attributes are involved, and standard Apriori algorithm is used to mine interesting rules. In the other part, the imbalanced attributes are mined on sub-datasets to achieve high efficiency. The following is the detailed procedure of the proposed framework.

1. Standard association rule mining on the balanced attributes. In the original dataset, the imbalanced attributes are excluded and all of the tuples are kept for association rule mining.
2. Filter the original dataset to obtain the tuples containing minority part of one imbalanced attribute. In this step, only a small portion of the dataset is kept.
3. Mine the association rules on the filtered dataset using predefined minimum *confidence* and minimum *conditional support*. For every imbalanced attribute, repeat Step 2 and Step 3.
4. Transform the measurements into a uniform space and put all of the mined rules together. The final class association rule list is selected based on a set of criteria.

Conditional Support. In order to mine the rules including imbalanced attributes, we extend the definition of *support* since the minority of an imbalanced attribute normally occurs in a small portion of the tuples. In this paper, *conditional support* is defined to measure the interestingness of the rules with imbalanced attributes. If a class association rule is $X + I_m \Rightarrow c$ and I_m is the minority part of one imbalanced attribute m, the *conditional support* of this rule is,

$$Supp_c = \frac{P(X \cup I_m \cup c)}{P(I_m)}$$

where $P()$ stands for the probability of the itemsets occurring, X is an itemset of balanced attributes, and I_m is a 1-itemset of imbalanced attribute m, c is a class ID. Note that X, I_m and c are all itemsets rather than tuple sets so that $X \cup I_m \cup c$ means X, I_m and c occur simultaniously.

Confidence and Lift. Suppose the original dataset is represented as T. The subset T_m consists of the tuples containing the minority of imbalanced attribute m. If an association rule is $X + I_m \Rightarrow c$, it is not difficult to prove that the confidence on original dataset T and subset T_m are same. The confidience of this rule is

$$Conf = Conf' = \frac{P(X \cup I_m \cup c)}{P(X \cup I_m)} \tag{1}$$

However, on T and T_m, the expected confidences are different, which are

$$Conf_E = P(c) \tag{2}$$

$$Conf'_E = P'(c) = \frac{P(I_m \cup c)}{P(I_m)} \tag{3}$$

Hence the lifts obtained on T and T_m are different. In order to use uniform criteria to select the rules, the lift on T_m has to be transformed. On the original dataset T, the expected confidence with respect to c is known, which is $P(c)$. On T_m, the confidence can also be obtained. So we may transform the lift obtained from subset T_m.

$$L_{new} = \frac{Conf'}{Conf_E} = \frac{Conf'}{P(c)} \tag{4}$$

So far the confidence, lift and the conditional support of all the rules are on the same base. We can use a uniform criteria to select the final rules. In this paper, minimum confidence, minimum lift and minimum conditional support are used to select the final class rule list.

4 Case Study

Our proposed algorithm has been tested with real-world data in Centrelink, Australia, which is an Australian government agency delivering a range of Commonwealth services to Australian community.

4.1 Problem Statement and Datasets

When a customer has received public monies to which he or she was not entitled, those funds may become a debt to be recovered. The purpose of data mining in debt recovery is to profile the customers according to the speed at which they pay off such debts. From a technical point of view, the objective is to mine the association rule with respect to the demographic attributes and debt information of a customer, the arrangement, and the target classes. Note that an arrangement is an agreement between a customer and Centrelink officer on the method, amount and frequency of repayments.

There are three kinds of datasets used for the association rule mining task: customer demographic data, debt data and repayment data. The class IDs, which are defined by business experts, are included in the repayment dataset. In the involved three datasets, there are three attributes having imbalanced distributions. For privacy reason, the three imbalanced attributes are denoted as “A”, “B” and “C” respectively. The majorities of these attributes are all more than 90%.

4.2 Experimental Results

In the experiments, all the customers are grouped based on the arrangement patterns. Thus each arrangement is associated with a group of customers. Since conditional support rather than conventional support is used to generate the frequent itemset, the association rules including minorities of imbalanced attributes are mined without exhaustive searching. In Table 1, “A:1”, “A:2”, “B:1”, “B:2” and “C:1” are all minorities of imbalanced attributes.

Table 1. Selected Rules with Imbalanced Attributes

Arrangement	Demographic Pattern	Class	$Conf_E$(%)	Conf(%)	$Supp_c$(%)	L_{new}	Count
W_W	Weekly:[\$200, \$400) & A:1 & GENDER:F	Class 1	39.0	48.6	6.7	1.2	52
C_A	MARITAL:SEP & A:1	Class 2	25.6	63.3	6.4	2.5	50
CI_A	Weekly:[\$400, \$600) & A:2	Class 3	35.4	64.9	6.4	1.8	50
WV_W	Children:0 & A:2 & MARITAL:SEP	Class 2	39.0	49.8	16.3	1.3	127
V_V	Weekly:0 & B:1 MARITAL:MAR &	Class 1	25.6	46.9	7.8	1.8	61
WV_WV	B:2 & GENDER:F	Class 3	25.6	49.7	11.4	1.9	89
WI_CW	Weekly:[\$200, \$400) & C:1 & GENDER:F	Class 3	39.0	45.7	18.8	1.2	147

5 Conclusions

This paper proposes an efficient algorithm to mine class association rules on the dataset with multiple imbalanced attributes. Unlike previous algorithms dealing with class imbalance, our algorithm processes the data imbalance on multiple attributes. Also different from the algorithms dealing with rare item problem, our algorithm employs a uniform selection criteria to discover the final combined association rule, which makes the algorithm more robust. The experimental results show the effectiveness of our proposed algorithm.

Acknowledgments

We would like to thank Mr. Fernando Figueiredo, Mrs. Leigh Galbrath, Mr. Shannon Marsh, Mr. Peter Newbigin, Mr. David Weldon, Ms. Michelle Holden and Mrs. Yvonne Morrow from Centrelink Australia for their support of domain knowledge and helpful comments.This work was supported by the Australian Research Council (ARC) Discovery Projects DP0449535, DP0667060 & DP0773412 and Linkage Project LP0775041.

References

1. Arunasalam, B., Chawla, S.: Cccs: a top-down associative classifier for imbalanced class distribution. In: KDD 2006, New York, NY, USA, pp. 517–522 (2006)
2. Gu, L., Li, J., He, H., Williams, G., Hawkins, S., Kelman, C.: Association rule discovery with unbalanced class distributions. In: Gedeon, T.D., Fung, L.C.C. (eds.) AI 2003. LNCS (LNAI), vol. 2903, pp. 221–232. Springer, Heidelberg (2003)
3. Liu, B., Hsu, W., Ma, Y.: Integrating classification and association rule mining. In: KDD 1998, New York, pp. 80–86 (August 27-31, 1998)
4. Liu, B., Hsu, W., Ma, Y.: Mining association rules with multiple minimum supports. In: KDD 1999, New York, USA, pp. 337–341 (1999)
5. Yun, H., Ha, D., Hwang, B., Ryu, K.H.: Mining association rules on significant rare data using relative support. Journal of Systems and Software 67(3), 181–191 (2003)

DBSC: A Dependency-Based Subspace Clustering Algorithm for High Dimensional Numerical Datasets

Xufei Wang and Chunping Li

School of Software,Tsinghua University
China MOE Key Laboratory for Information System Security
xf-wang05@mails.tsinghua.edu.cn,
cli@mail.tsinghua.edu.cn

Abstract. We present a novel algorithm called DBSC, which finds subspace clusters in numerical datasets based on the concept of "dependency". This algorithm uses a depth-first search strategy to find out the maximal subspaces: a new dimension is added to current k-subspace and its validity as a (k+1)-subspace is evaluated. The clusters within those maximal subspaces are mined in a similar fashion as maximal subspace mining does. With the experiments on synthetic and real datasets, our algorithm is shown to be both effective and efficient for high dimensional datasets.

1 Introduction

The development of genetic and spatial technologies creates high dimensional datasets whose useful information can't be easily captured by traditional clustering techniques in that clusters are hidden in a subset of the dimensions rather than in the full space. Subspace clustering is designed to explore the clusters for this purpose.

CLIQUE[1],MAFIA[2] and SCHISM[3] are density-based subspace clustering algorithms. CLIQUE partitions each dimension into equidistant intervals by a user specified parameter ξ, the hyper rectangle regions above a minimal *support* measured by density are kept for further clustering. It uses a bottom-up style to grow higher dimensional subspaces. To speed up this algorithm, Minimal Description Length(MDL) principal is used to prune relatively low coverage subspaces. MAFIA extends CLIQUE by using an adaptive grid and introducing parallelism calculation. SCHISM extends the CLIQUE algorithm in another way which is based on the notion of *Chernoff-Hoeffding* bound. CLICKS[4] is dedicated to find clusters in categorical datasets based on the search for k-partite maximal cliques. ENCLUS[5] does not measure density or coverage directly but entropy instead. It uses the information gain to evaluate the *interest* of a subspace. ENCLUS demonstrates entropy as a reasonable clustering criterion by emphasizing the relationship with coverage, density and correlation. It grows the higher dimensional subspace in a similar style as CLIQUE does: combining (k-1)-dimensional subspaces into k-dimensional subspaces. To reduce search space, two closure properties are used to prune those relatively uninteresting subspaces.

However, information gain is difficult to select for ENCLUS. It is desirable to propose a more natural measure in evaluating the *interest* of a subspace. In this paper, we developed the concept of *dependency* or in detail, we evaluate how a dimension Y can be determined by current k-subspace S_k. Another difference of the proposed approach with ENCLUS is that we mine both subspaces and clusters in those subspaces.

2 Theoretical Foundations for DBSC

Basic Concepts: We denote $A_1,\ldots,A_n$ as the *attributes* and $D_1,\ldots,D_n$ as the domain for each attribute. Let $S_j \in \{D_1,\ldots,D_n\}$, and $S_i \cap S_j = \phi$, a k-subspace is denoted as $S_1 \times \ldots \times S_k$, when k equals to n, this n-subspace is called a *full space*. Suppose that X $=X_1 \times \ldots \times X_m$, Y $=Y_1 \times \ldots \times Y_n$, X is *contained* in Y, denoted by $X \subseteq Y$, iff m $\leq$ n and for all $X_j \in$ Y. A *maximal subspace* M means that no other subspace M' contains it, i.e., M $\not\subset M'$. A maximal k-subspace S contains at least one *cluster*, which is represented by a *conjunctive normal form(CNF)* expression: $[A_{1low}, A_{1high}] \wedge \ldots \wedge [A_{klow}, A_{khigh}]$, where "low" and "high" are the lower and higher bound of this cluster in attribute A_k. *Cohesive Object* is a representation of a group of data points with high inner cohesion(similarity). It plays a central role in our proposed algorithm and its creation and separation are further explained in Section 3. It has a similar expression as the cluster does. When a *Cohesive Object* is partitioned into several ones, we call those new *Cohesive Objects* as *Child Cohesive Objects*.

Dependency as a Clustering Criterion: Let X, Y be two random variables, we refer to I(X;Y) as the mutual information between X and Y, and H(X) the entropy of X.

Definition 1. *Let k-dimensional subspace be $S_k(v_1,\ldots, v_k)$, we are interested in the probability that a new dimension v_{k+1} and k-subspace $S_k(v_1,\ldots, v_k)$ in constituting a greater (k+1)-subspace. We call this probability "dependency", which represents how much the distribution in dimension v_{k+1} is determined given k-subspace S_k. Two random variables are used in following formula: X for k-subspace S_k and Y for 1-subspace $S_{v_{k+1}}$.*

$$\begin{aligned} dependency(Y;X) &= I(X;Y)/H(Y) \\ &= 1 - \frac{\sum_{co \in S_k} \sum_{x \in X(co)} (p(x|co) \cdot \log_2 p(x|co)) \cdot p(co)}{\sum_{x \in X(co)} p(x) \cdot \log_2 p(x)} \end{aligned} \quad (1)$$

In Formula (1), S_k is k-subspace; *co* is the abbreviation for *Cohesive Object*, *X(co)* is the collection of all *Child Cohesive Objects* within *Cohesive Object co* which is partitioned by adding the new dimension Y to k-subspace S_k. With the definition of "dependency", we call an *interesting subspace* as its dependency is greater than a user defined threshold α, otherwise, we call it *uninteresting subspace*.

Lemma 1. (Dimension Correlation): *Let dimensions X, Y, Z be random variables, I(X;Y) = 0 iff X is independent with Y. If X, Y are independent with Z, then $I(X;YZ)$ = I(X;Y).* This indicates that adding an independent random variable has no impact for the information of a system. In other words, two independent dimensions don't constitute an interesting subspace, we thus seek to find the dimensions that are highly correlated.

Lemma 2. (Downward Closure) : *Let k-subspace S_k be an interesting subspace, then all its (k-1)-subspaces S_{k-1} are interesting subspaces.*

Proof: we give a proof by contradiction. Suppose that one of the (k-1)-subspaces S'_{k-1} is an *uninteresting subspace*, in other word, there is at least one dimension Y' in subspace S'_{k-1} satisfying $dependency(Y';S'_{k-2}) < \alpha$. When a new dimension Y is added into this (k-1)-subspace S'_{k-1}, the new dependence $dependency(Y';S'_{k-2}Y)$ is calculated by $\frac{I(Y';S'_{k-2};Y)}{H(Y')} = \frac{I(Y';S'_{k-2})-I(Y';S'_{k-2}|Y)}{H(Y')} \leq \frac{I(Y';S'_{k-2})}{H(Y')} < \alpha$, which contradicts the assumption that k-subspace S_k is an *interesting subspace*. ■

Definition 2. Growing direction is defined as the occurrence sequence between two dimensions. Let two dimensions be X and Y, if dependency(Y,X) > dependency(X,Y), the growing direction is from X to Y; otherwise the growing direction is from Y to X.

By Definition 1, *dependency* is inversely propositional to the entropy of Y: the higher the entropy of Y, the higher the uncertainty of Y, thus the lower the dependency of Y to X. Our algorithm always prefers the dimension with lower uncertainty as its growing direction.

3 Algorithm Implementation and Evaluation

DBSC is a depth-first algorithm which searches maximal subspaces starting from a single dimension, and then adding a new dimension to current k-subspace(parent subspace) in each step and testing the validity of the greater (k+1)-subspace(child subspace). This process stops until either the child *dependency* is below the threshold α or the parent's. This algorithm starts by adding all 1-subspaces into a set and the *Cohesive Objects* for each 1-subspace are created in this step. This mining process is guaranteed to be complete since each possible subspace is tested for its validity. Our algorithm contains four steps which will be discussed as below.

Cohesive Object Creation: Mundhenk[6] proposed a basic idea to find clusters with less assumptions such as the size and count of clusters. It first finds out the density and distance between different points in feature space and a parent-child hierarchy is built by assigning a point to its closest and higher density point, and then breaks this tree into several subtrees if the distance between two points are far away enough. We provide a tree breaking threshold β which satisfies $P\{\mid Dist(i,j) - Exp(Dist) \mid > \beta \cdot \sqrt{Var(Dist)}\} < \frac{1}{\beta^2}$, where Dist(i,j) is the Euclidean distance between two nodes i and j, *Exp(Dist)* is the mean value, i.e., $Exp(Dist) = \frac{1}{n}\sum Dist(i,j)$ and *Var(Dist)* is the variance of the distance sequence, i.e., $Var(Dist) = \frac{1}{n}\sum Dist^2(i,j) - (\frac{1}{n}\sum Dist(i,j))^2$. The subtrees which behave strong inner cohesion are called *Cohesive Objects*.

Pre-Prune Operation: Since the overall number of subspaces are $\sum_{1\leq k\leq n} P_n^k$ for a n-dimensional dataset, where P_n^k is the permutation of k from n, this makes the time cost for testing all subspaces unacceptable. We also capture the fact that many of the dimensions fail to constitute an *interesting subspace*. The pre-prune operation tests all 2-subspaces S and makes sure whether those two dimensions are making an *interesting subspace*. The reason for this strategy is natural when we notice that 2-subspaces are the most basic elements in making greater subspaces. The rules described in Definition **2** can be applied to efficiently prune half of the subspaces. The down closure property of *dependency* shows a k-subspace is an *uninteresting space* if any of its (k-1)-subspace

Table 1. Time Performance of ENCLUS and DBSC by Different Dimension and Dataset Size

dim	10	20	40	60	80	100	150	200	count(k)	10	30	50	100	150	250	300
ENCLUS	3	35	404	1564	3556	7470	-	-	ENCLUS	592	1847	2975	5802	8538	-	-
DBSC	7	11	16	25	47	66	191	390	DBSC	19	49	82	196	368	481	606

Table 2. Wine Dataset

class	C1	C2	C3	C4	C5
1	13			41	
2	3	55		4	2
3			11		17

Table 3. Breast Cancer

class	C1	C2	C12
Benign	64	90	26
Malignant	5	1	430

Table 4. PenDigits Dataset

Class	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12
'0'				1		180		39				
'1'	12						5		161	30	15	91
'2'	141				5		221			2	114	
'3'	538						2			24		
'4'	10			292			3			22	1	
'5'	71					46		127		28		
'6'		15	224		290		8			1	1	
'7'	1	338	96		1		3			1		2
'8'	1	14						70				
'9'	30					53				23		32

is an *uninteresting subspace*. Following this approach, most of the *uninteresting subspaces* are excluded from further processing.

Maximal Subspace Mining: The maximal subspaces are mined through a depth-first search strategy, to add a new dimension v_{k+1} into a k-subspace and to test its validity as a (k+1)-subspace. Each k-subspace maintains a local threshold identifying current level of *dependency*. If the *dependency* of (k+1)-subspace is greater than this local threshold, the local threshold is heuristically increased as the new dependency. A drop in this dependency indicates that the new added dimension makes the k-subspace uncertain. Adding a new dimension v_{k+1} makes a partition for each *Cohesive Object* in current subspace. This partition process is a *Cohesive Object Creation* process on the data points in each *Cohesive Object* and all data points in each *Cohesive Object* are distributed into its children ones. Then the dependency of the new dimension v_{k+1} to k-subspace S_k is computed by Definition (1). If the new *dependency* is either below the threshold α or the parent's, the subspace S_k is labeled as *uninteresting subspace* and its parent a *maximal subspace*. However, if both criterions hold, this subspace is registered as (k+1)-subspace. This above process continues until no new dimension can be added to current subspace.

Clustering in Maximal Subspace: After all *maximal subspaces* are figured out, DBSC turns to each subspace for all clusters embedded in it. Similar to **Maximal Subspace Mining**, it starts by one dimension, add one dimension each time to make a partition until all the dimensions in this subspace are covered. Following the definition of *Cohesive Object*, we get collections of data points with high inner cohesion. In other words, we get the clusters in subspaces. Clustering in other subspaces are the same.

Experimental Evaluations: The synthetic dataset is the same as the one used by ENCLUS [5]. Each dataset used in the following experiment contains 2 subspaces and each subspace has 2 and 3 clusters. In Table 1, DBSC outperforms ENCLUS both by different dimension(part 1) and different dataset size(part 2). DBSC has a time complexity of O(n^2+n) where n is the dimensionality of the dataset, and its time cost is linear to the dataset size. On above experiments, DBSC succeeds in finding out those two subspaces, however, ENCLUS(ω = 8.5, ϵ = 0.01) finds out three maximal subspaces but only one of them is correct, noticing that other parameters are also tried. DBSC also has the capacity of outlier tolerance because outliers do not show a strong statistical behavior and are removed from the *Cohesive Objects* in the subspace growing process, which eventually produces pure clusters. The selection of α and β are not critical in our algorithm. Various experiments suggests α is suitable to be set between 0.2 and 0.5 and β be set between 0.5 and 1.5.
We apply DBSC to three numerical real datasets: BreastCancer, Wine and PenDigits[1]. We only list part of the largest clusters in the following tables because of space limitation. Table 2, 3 lists the cluster distribution for Wine(α = 0.25, β=0.5) and BreastCancer(α = 0.3, β=0.5). We are achieving an accuracy of 95.0% and 93.2% respectively. As for PenDigits(α = 0.2, β=0.6), we get the confusion maxtrix in Table 4. Besides the pure clusters, we also capture some interesting information: some of the digits that show a similar structure, such as {2,3,9} and {0,5,8,9}, are grouped together in cluster C1, C6, C8. On the other hand, a same digit, such as 2, is distributed in different clusters C1, C5, C7, C10 and C11 because those handwritings differ in sampled locations.

4 Conclusion

In this paper, we present the algorithm DBSC for clustering high-dimensional numerical datsets based on the concept of "dependency", which originates from the fact that clusters are hidden in dimensions with high correlation. Our algorithm uses a depth-first strategy by starting from 1-subspaces to mine maximal subspaces. The downward closure and growing direction are used to prune the uninteresting subspaces. Extensive experiments show that DBSC is efficient, robust and scalable and it's effective to provide useful information for users.

Acknowledgements. We would like to thank the useful suggestions from reviewers. This work is funded by China 973 Research Project under Grant No.2002CB312006.

References

1. Agrawal, R., Gehrke, J., Gunopulos, D., et al.: Automatic Subspace Clustering of High Dimensional Data for Data Mining Applications. In: Proceedings of the ACM SIGMOD, pp. 94–105. ACM Press, New York (1998)

[1] We get dataset BreastCancer, Wine and PenDigits from UCI Machine Learning Repository. http://www.ics.uci.edu/ mlearn/MLRepository.html

2. Goil, S., Nagesh, H., Choudhary, A.: MAFIA: Efficient and Scalable Subspace Clustering for very Large Datasets.Technical Report CPDC-TR-9906-010, Northwestern University (June 1999)
3. Sequeira, K., Zaki, M.: SCHISM: A New Approach for Interesting Subspace Mining. In: The Proceedings of the Fourth IEEE Conference On Data Mining, pp. 186–193 (2004)
4. Zaki, M.J., Peters, M., et al.: CLICKS: an Effective Algorithm for Mining Subspace Clusters in Categorical Datasets. In: Proceeding of the eleventh ACM SIGKDD, pp. 736–742 (2005)
5. Cheng, C.H., Fu, A.W., Zhang, Y.: Entropy-based Subspace Clustering for Mining Numerical Data. In: Proceedings of ACM SIGKDD, pp. 84–93 (1999)
6. Mundhenk, T.N., Navalpakkam, V., Makaliwe, H., Vasudevan, S., Itti, L.: Biologically Inspired Feature Based Categorization of Objects. In: HVEI 2004. Proc. SPIE Human Vision and Electronic Imaging IX, San Jose, CA, Bellingham, vol. 5292 (January 2004)

Author Index