COMPUTER SCIENCE 2015

Proceedings of the 38th Australasian Computer Science Conference (ACSC 2015),
Sydney, Australia, 27 - 30 January 2015

David Parry, Eds.

Volume 159 in the Conferences in Research and Practice in Information Technology Series.
Published by the Australian Computer Society Inc.

Published in association with the ACM Digital Library.

Conferences in Research and Practice in Information Technology, Volume 159.

Copyright ©2015, Australian Computer Society. Reproduction for academic, not-for-profit purposes permitted provided the copyright text at the foot of the first page of each paper is included.

Editor:

David Parry
School of Computer and Mathematical sciences
AUT University
Private Bag 92006
Auckland 1142
New Zealand
Email: dave.parry@aut.ac.nz

Series Editors:
Vladimir Estivill-Castro, Griffith University, Queensland
Simeon J. Simoff, University of Western Sydney, NSW
Email: crpit@scem.uws.edu.au

Publisher: Australian Computer Society Inc.
PO Box Q534, QVB Post Office
Sydney 1230
New South Wales
Australia.

Conferences in Research and Practice in Information Technology, Volume 159.
ISSN 1445-1336.

Document engineering, January 2015 by CRPIT
On-line proceedings, January 2015 by the University of Western Sydney
Electronic media production, January 2015 by Laurence Park, University of Western Sydney

The Conferences in Research and Practice in Information Technology series disseminates the results of peer-reviewed research in all areas of Information Technology. Further details can be found at http://crpit.com/.
# Table of Contents

**Proceedings of the 38th Australasian Computer Science Conference (ACSC 2015), Sydney, Australia, 27 - 30 January 2015**

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Preface</td>
<td>vii</td>
</tr>
<tr>
<td>Programme Committee</td>
<td>viii</td>
</tr>
<tr>
<td>Organising Committee</td>
<td>ix</td>
</tr>
<tr>
<td>Welcome from the Organising Committee</td>
<td>x</td>
</tr>
<tr>
<td>CORE - Computing Research &amp; Education</td>
<td>xi</td>
</tr>
<tr>
<td>ACSW Conferences and the Australian Computer Science Communications</td>
<td>xiii</td>
</tr>
<tr>
<td>ACSW and ACSC 2015 Sponsors</td>
<td>xv</td>
</tr>
</tbody>
</table>

## Contributed Papers

- **Image Reconstruction based on Block-based Compressive Sensing**
  *Hanxu You and Jie Zhu*
  
  Page 3

- **An integration of software engineering methods and semantic technologies for drafting and modeling statutes and legal rules**
  *Martin Stabauer, Gerald Quirchmayr and Johann Höller*
  
  Page 9

- **Schema Mapping Using Hybrid Ripple Down Rules**
  *Sarawat Anam, Yang Sok Kim, Byeong Kang and Qing Liu*
  
  Page 17

- **A novel method for decentralised peer-to-peer software license validation using cryptocurrency blockchain technology**
  *Jeff Herbert and Alan Litchfield*
  
  Page 27

- **Visualising Moving Clusters using Cluster Flow Diagrams**
  *Jason Thompson and Torab Torabi*
  
  Page 37

- **A Machine Learning approach to Generic Entity Resolution in support of Cyber Situation Awareness**
  *Christopher Moir and Jonathan Dean*
  
  Page 47

- **A new extension of Kernel Principal Component Analysis for Finger Vein Authentication**
  *Sepehr Damavandinejadmonfared and Vijay Varadharajan*
  
  Page 59

- **Dynamic Topic Detection Model by Fusing Sentiment Polarity**
  *Xi Ding, Lanshan Zhang, Ye Tian, Xiangyang Gong and Wendong Wang*
  
  Page 65

- **A macro-level model for investigating the effect of directional bias on network coverage**
  *Graeme Smith, J W Sanders and Qin Li*
  
  Page 73

- **An Evaluation of POS tagging for Tweets Using HMM Modelling**
  *Parma Nand and Rivinda Perera*
  
  Page 83

- **Component Modeling for SCADA Network Mapping**
  *Ayodeji James Akande, Colin Fidge and Ernest Foo*
  
  Page 91

- **An Enhanced Model for Network Flow Based Botnet Detection**
  *Udaya Wijesinghe, Udaya Tupakula and Vijay Varadharajan*
  
  Page 101
Lego Mindstorms: Merely a Toy or a Powerful Pedagogical Tool for Learning Computer Programming? 11
Jacqui Chetty

Hourglass Systems .................................................. 119
Yuki Osada, Mark Reynolds and Tim French

Author Index ......................................................... 125
Preface

The Australasian Computer Science Conference (ACSC) series is an annual meeting, bringing together research sub-disciplines in Computer Science, in Australasia and around the world. The conference allows academics and other researchers to discuss research topics as well as progress in the field, and policies to stimulate its growth. This conference is unique in its ability to provide a platform for cross-disciplinary research, especially for computer scientists based in Australia and New Zealand. This volume comprises papers being presented at the Thirty-Eighth ACSC in Parramatta, Sydney, New South Wales.

ACSC 2015 is part of the Australasian Computer Science Week which runs from January 27th to 30th January, 2015. The ACSC 2015 call for papers solicited 33 submissions from Australia, New Zealand, Austria, China, Indonesia, Iran, Japan, Poland, Pakistan, South Africa, Taiwan and Thailand. The topics addressed by the submitted papers illustrate the breadth of the discipline. These included algorithms, natural language processing, machine learning, networking, service-orientated computing, education, software engineering, social media, and security, to name just a few.

The programme committee consisted of 24 highly regarded academics from Australia, New Zealand, Japan, China, Korea, Singapore and the USA. Every paper was reviewed by at least three programme committee members. Of the 33 papers submitted, 14 were selected for presentation at the conference.

The standard of papers produced by or with PhD students is particularly high, and this is a hopeful sign for our discipline. The Programme Committee determined that the “Best Student Paper Award” should go to Ayodeji James Akande, Colin Fidge and Ernest Foo for “Component Modeling for SCADA Network Mapping”; Congratulations!

We thank all authors who submitted papers and all conference participants for helping to make the conference a success. We also thank the members of the programme committee for their expertise in carefully reviewing the papers and Laurence Park, has worked tirelessly to produce the proceedings. Last, but not least, we express gratitude to our hosts at the University of Western Sydney and the ACSW general chairs Athula Ginige, and Paul Kennedy.

David Parry
Auckland University of Technology
ACSC 2015 Programme Chair
January 2015
Programme Committee

Chair
David Parry, Auckland University of Technology

Members
Yuping Shen, Institute of Logic and Cognition, Department of Philosophy, Sun Yat-sen University
Alan Litchfield, Auckland University of Technology
Parma Nand, Auckland University of Technology
Jairo Gutierrez, Auckland University of Technology
Zhiyi Huang, University of Otago
Burkhard Wünsche, The University of Auckland
Maurice Pagnucco, The University of New South Wales
Muhammed Asifnaeem, Auckland University of Technology
Alex Potanin, Victoria University of Wellington
Jun Park, Hongik University
Hua Wang, University of Southern Queensland, Australia
Curtis Dyreson, Utah State University
William Liu, Auckland University of Technology
Jixue Liu, University of South Australia
Michael E. Houle, National Institute of Informatics
Stephane Bressan, National University of Singapore
Ewan Tempero, The University of Auckland
Andrew Turpin, The University of Melbourne
Paddy Krishnan, Oracle
Jianlong Zhou, National ICT Australia
Shuji Kijima, Kyushu University
Xiaofang Zhou, University of Queensland
Wolfgang Mayer, University of South Australia
Chris McDonald, The Univ. of Western Australia
Organising Committee

Chairs
Professor Athula Ginige, University of Western Sydney, Australia
Associate Professor Paul Kennedy, University of Technology, Sydney, Australia

Local Chair
Dr Bahman Javadi, University of Western Sydney, Australia

Publicity Chair
Dr Ante Prodan, University of Western Sydney, Australia

Publication Chair
Dr Laurence Park, University of Western Sydney, Australia

Finance Chair
Michael Walsh, University of Western Sydney, Australia

Sponsorship Chair
Kerry Holling, University of Western Sydney, Australia

Doctoral Consortia Co-chairs
Professor Anthony Maeder, University of Western Sydney, Australia
Dr Siamak Tafavogh, University of Technology, Sydney, Australia

Event Coordinator
Nicolle Fowler, University of Western Sydney, Australia
Welcome from the Organising Committee

On behalf of the Organising Committee, it is our pleasure to welcome you to Sydney and to the 2015 Australasian Computer Science Week (ACSW 2015). This year the conference is hosted by the University of Western Sydney and its School of Computing, Engineering and Mathematics.

A major highlight of the ACSW 2015 will be the Industry Research Nexus day on 27th January 2015. The aim is for industry leaders and academic researchers to come together and explore research areas of mutual interest. Many University research groups and 15 industries have confirmed their participation.

ACSW 2015 consists of 9 sub conferences covering a range of topics in Computer Science and related areas. These conferences are:

- Asia-Pacific Conference on Conceptual Modelling (APCCM) (Chaired by Motoshi Saeki and Henning Köhler)
- Australasian Computer Science Conference (ACSC) (Chaired by Dave Parry)
- Australasian Computing Education Conference (ACE) (Chaired by Daryl D’Souza and Katrina Falkner)
- Australasian Information Security Conference (AISC) (Chaired by Ian Welch and Xun Yi)
- Australasian Symposium on Parallel and Distributed Computing (AusPDC) (Chaired by Bahman Javadi and Saurabh Garg)
- Australasian User Interface Conference (AUIC) (Chaired by Stefan Marks and Rachel Blagojevic)
- Australasian Web Conference (AWC) (Chaired by Joseph Davis)
- Australasian Workshop on Health Informatics and Knowledge Management (HIKM) (Chaired by Anthony Maeder and Jim Warren)
- Interactive Entertainment (IE) (Chaired by Yusuf Pisan and Keith Nesbitt)

Social events are a very important part of a conference as these provide many networking opportunities. To foster networking we have included a reception with industry on 27th January 2015, a Welcome reception on 28th January 2015 and a conference dinner on 29th January 2015.

Organising a multi-conference event such as ACSW is a challenging process even with many hands helping to distribute the workload, and actively cooperating to bring the events to fruition. This year has been no exception. We would like to share with you our gratitude towards all members of the organising committee for their combined efforts and dedication to the success of ACSW2015. We also thank all conference co-chairs and reviewers, for putting together the conference programs which are the heart of ACSW, and to the organisers of the sub conferences, workshops, poster sessions and Doctoral Consortium. Special thanks to John Grundy as chair of CoRE for his support for the innovations we have introduced this year.

This year we have secured generous support from several sponsors to help defray the costs of the event and we thank them for their welcome contributions. Last, but not least, we would like to thank all speakers, participants and attendees, and we look forward to several days of stimulating presentations, debates, friendly interactions and thoughtful discussions.

Athula Ginige
University of Western Sydney

Paul Kennedy
University of Technology Sydney

ACSW2015 General Co-Chairs
January, 2015
CORE welcomes all delegates to ACSW2015 in Sydney. CORE, the peak body representing academic computer science in Australia and New Zealand, is responsible for the annual ACSW series of meetings, which are a unique opportunity for our community to network and to discuss research and topics of mutual interest. The component conferences of ACSW have changed over time with additions and subtractions: ACSC, ACE, AISC, AUIC, AusPDC, HIKM, ACDC, APCCM, CATS and AWC. Two doctoral consortia (ACDC and ACE-DC) and an Australasian Early Career Researchers Workshop (AECRW) reflect the evolving dimensions of ACSW and build on the diversity of the Australasian computing community. A specific industry day on the 27th January to facilitate academic / industry discussion and networking is a key feature of ACSW 2015.

In 2015, we are fortunate to have Professor Omer Rana, Associate Professor Pascal Hitzler and Professor Mark Sagar providing keynote talks to the conference. I thank them for their contributions to ACSW2015.

The efforts of the conference chairs and their program committees have led to strong programs in all the conferences, thanks very much for all your efforts. Thanks are particularly due to Professor Athula Ginige, Professor Paul Kennedy and their colleagues for organising what promises to be a vibrant event. Below I outline some of CORE’s activities in 2013/14.

I welcome feedback on these including other activities you think CORE should be active in.

The major sponsor of Australian Computer Science Week:
- The venue for the annual Heads and Professors meeting
- An opportunity for Australian & NZ computing staff and postgrads to network and help develop their research and teaching
- Substantial discounts for attendees from member departments
- A doctoral consortium at which postgrads can seek external expertise for their research
- An Early Career Research forum to provide ECRs input into their development

Sponsor of several research, teaching and service awards:
- Chris Wallace award for Distinguished Research Contribution
- CORE Teaching Award
- Australasian Distinguished Doctoral Dissertation
- John Hughes Distinguished Service Award
- Various “Best Student Paper” awards at ACSW

Development, maintenance, and publication of the CORE conference and journal rankings. In 2014 this includes a heavily-used web portal with a range of holistic venue information and a community update of the CORE 2013 conference rankings.

Input into a number of community resources and issues of interest:
- Development of an agreed national curriculum defining Computer Science, Software Engineering, and Information Technology
- A central point for discussion of community issues such as research standards
- Various submissions on behalf of Computer Science Departments and Academics to relevant government and industry bodies, including recently on Australian Workplace ICT Skills development, the Schools Technology Curriculum and the Defence Trade Controls Act.

Coordination with other sector groups:
- Work with the ACS on curriculum and accreditation
- Work with groups such as ACDICT, ACPHIS and government on issues such as CS staff performance metrics and appraisal, and recruitment of students into computing
- A member of CRA (Computing Research Association) and Informatics Europe. These organisations are the North American and European equivalents of CORE.
- A member of Science & Technology Australia, which provides eligibility for Science Meets Parliament and opportunity for input into government policy, and involvement with Science Meets Policymakers

The 2014 Executive Committee has been looking at a range of activities that CORE can lead or contribute to, including more developmental activities for CORE members. This has also included a revamp of the mailing lists, web site, creation of discussion forums, identification of key issues for commentary and lobbying, and working with other groups to attract high aptitude students into ICT courses and careers.
Again, I welcome your active input into the direction of CORE in order to give our community improved visibility and impact. CORE’s existence is due to the support of the member departments in Australia and New Zealand, and I thank them for their ongoing contributions, in commitment and in financial support. Finally, I am grateful to all those who gave their time to CORE in 2014, and look forward to the continuing shaping and development of the Australasian computing community in 2015.

John Grundy
President, CORE
January, 2015
ACSW Conferences and the Australian Computer Science Communications

The Australasian Computer Science Week of conferences has been running in some form continuously since 1978. This makes it one of the longest running conferences in computer science. The proceedings of the week have been published as the Australian Computer Science Communications since 1979 (with the 1978 proceedings often referred to as Volume 0). Thus the sequence number of the Australasian Computer Science Conference is always one greater than the volume of the Communications. Below is a list of the conferences, their locations and hosts.

<table>
<thead>
<tr>
<th>Year</th>
<th>Volume</th>
<th>Host and Venue</th>
</tr>
</thead>
<tbody>
<tr>
<td>2016</td>
<td>38</td>
<td>Australian National University, Canberra, ACT.</td>
</tr>
<tr>
<td>2015</td>
<td>37</td>
<td>University of Western Sydney, NSW.</td>
</tr>
<tr>
<td>2014</td>
<td>36</td>
<td>AUT University, Auckland, New Zealand.</td>
</tr>
<tr>
<td>2013</td>
<td>35</td>
<td>University of South Australia, Adelaide, SA.</td>
</tr>
<tr>
<td>2012</td>
<td>34</td>
<td>RMIT University, Melbourne, VIC.</td>
</tr>
<tr>
<td>2011</td>
<td>33</td>
<td>Curtin University of Technology, Perth, WA.</td>
</tr>
<tr>
<td>2010</td>
<td>32</td>
<td>Queensland University of Technology, Brisbane, QLD.</td>
</tr>
<tr>
<td>2009</td>
<td>31</td>
<td>Victoria University, Wellington, New Zealand.</td>
</tr>
<tr>
<td>2008</td>
<td>30</td>
<td>University of Wollongong, NSW.</td>
</tr>
<tr>
<td>2007</td>
<td>29</td>
<td>University of Ballarat, VIC. First running of HDKM.</td>
</tr>
<tr>
<td>2006</td>
<td>28</td>
<td>University of Tasmania, TAS.</td>
</tr>
<tr>
<td>2005</td>
<td>27</td>
<td>University of Newcastle, NSW. APBC held separately from 2005.</td>
</tr>
<tr>
<td>2004</td>
<td>26</td>
<td>University of Otago, Dunedin, New Zealand. First running of APCCM.</td>
</tr>
<tr>
<td>2002</td>
<td>24</td>
<td>Monash University, Melbourne, VIC.</td>
</tr>
<tr>
<td>2001</td>
<td>23</td>
<td>Bond University and Griffith University (Gold Coast). Venue - Gold Coast, QLD.</td>
</tr>
<tr>
<td>2000</td>
<td>22</td>
<td>Australian National University and University of Canterbury. Venue - ANU, Canberra, ACT. First running of AUIC.</td>
</tr>
<tr>
<td>1999</td>
<td>21</td>
<td>University of Auckland, New Zealand.</td>
</tr>
<tr>
<td>1998</td>
<td>20</td>
<td>University of Western Australia, Murdoch University, Edith Cowan University and Curtin University. Venue - Perth, WA.</td>
</tr>
<tr>
<td>1996</td>
<td>18</td>
<td>University of Melbourne and RMIT University. Venue - Melbourne, Australia. CATS joins ACSW.</td>
</tr>
<tr>
<td>1995</td>
<td>17</td>
<td>Flinders University, University of Adelaide and University of South Australia. Venue - Glenelg, SA.</td>
</tr>
<tr>
<td>1994</td>
<td>16</td>
<td>University of Canterbury, Christchurch, New Zealand. CATS run for the first time separately in Sydney.</td>
</tr>
<tr>
<td>1993</td>
<td>15</td>
<td>Griffith University and Queensland University of Technology. Venue - Nathan, QLD.</td>
</tr>
<tr>
<td>1992</td>
<td>14</td>
<td>University of Tasmania, TAS. (ADC held separately at La Trobe University).</td>
</tr>
<tr>
<td>1991</td>
<td>13</td>
<td>University of New South Wales, NSW.</td>
</tr>
<tr>
<td>1990</td>
<td>12</td>
<td>Monash University, Melbourne, VIC. Joined by Database and Information Systems Conference which in 1992 became ADC (which stayed with ACSW) and ACIS (which now operates independently).</td>
</tr>
<tr>
<td>1989</td>
<td>11</td>
<td>University of Wollongong, NSW.</td>
</tr>
<tr>
<td>1988</td>
<td>10</td>
<td>University of Queensland, QLD.</td>
</tr>
<tr>
<td>1987</td>
<td>9</td>
<td>Deakin University, VIC.</td>
</tr>
<tr>
<td>1986</td>
<td>8</td>
<td>Australian National University, Canberra, ACT.</td>
</tr>
<tr>
<td>1985</td>
<td>7</td>
<td>University of Melbourne and Monash University. Venue - Melbourne, VIC.</td>
</tr>
<tr>
<td>1984</td>
<td>6</td>
<td>University of Adelaide, SA.</td>
</tr>
<tr>
<td>1983</td>
<td>5</td>
<td>University of Sydney, NSW.</td>
</tr>
<tr>
<td>1982</td>
<td>4</td>
<td>University of Western Australia, WA.</td>
</tr>
<tr>
<td>1981</td>
<td>3</td>
<td>University of Queensland, QLD.</td>
</tr>
<tr>
<td>1980</td>
<td>2</td>
<td>Australian National University, Canberra, ACT.</td>
</tr>
<tr>
<td>1979</td>
<td>1</td>
<td>University of Tasmania, TAS.</td>
</tr>
<tr>
<td>1978</td>
<td>0</td>
<td>University of New South Wales, NSW.</td>
</tr>
</tbody>
</table>
### Conference Acronyms

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACDC</td>
<td>Australasian Computing Doctoral Consortium</td>
</tr>
<tr>
<td>ACE</td>
<td>Australasian Computing Education Conference</td>
</tr>
<tr>
<td>ACSC</td>
<td>Australasian Computer Science Conference</td>
</tr>
<tr>
<td>ACSW</td>
<td>Australasian Computer Science Week</td>
</tr>
<tr>
<td>ADC</td>
<td>Australasian Database Conference</td>
</tr>
<tr>
<td>AISC</td>
<td>Australasian Information Security Conference</td>
</tr>
<tr>
<td>APCCM</td>
<td>Asia-Pacific Conference on Conceptual Modelling</td>
</tr>
<tr>
<td>AUIC</td>
<td>Australasian User Interface Conference</td>
</tr>
<tr>
<td>AusPDC</td>
<td>Australasian Symposium on Parallel and Distributed Computing (replaces AusGrid)</td>
</tr>
<tr>
<td>AWC</td>
<td>Australasian Web Conference</td>
</tr>
<tr>
<td>CATS</td>
<td>Computing: Australasian Theory Symposium</td>
</tr>
<tr>
<td>HIKM</td>
<td>Australasian Workshop on Health Informatics and Knowledge Management</td>
</tr>
<tr>
<td>IE</td>
<td>Australasian Conference on Interactive Entertainment</td>
</tr>
</tbody>
</table>

Note that various name changes have occurred, which have been indicated in the Conference Acronyms sections in respective CRPIT volumes.
ACSW and ACSC 2015 Sponsors

We wish to thank the following sponsors for their contribution towards this conference.

Host Sponsors
University of Western Sydney
www.uws.edu.au
Australian Computer Society
www.acs.org.au
Computing Research and Education
www.core.edu.au

Platinum Sponsor
Dimension Data
www.dimensiondata.com

Gold Sponsors
NTT Australia Pty Ltd
www.au.ntt.com
Hewlett-Packard Company
www.hp.com
Intersect
www.intersect.org.au
Cognizant Technology Solutions
www.cognizant.com

Silver Sponsors
SGI
wwwsgi.com
SMS Management and Technology
www.smsnt.com
AARNet
www.aarnet.edu.au
Macquarie Telecom
www.macquarie telecom.com

Bronze Sponsors
Australian Access Federation
aaf.edu.au
NEC Australia Pty Ltd
au.nec.com
Squiz Australia
www.squiz.net/au
Talent RISE
www.talentrise.org
Espire Infolabs Pty Ltd
www.espire.com
Contributed Papers
Abstract

The data of interest are assumed to be represented as N-dimensional real vectors, and these vectors are compressible in some linear basis B, implying that the signals can be reconstructed accurately using only a small number of basis function coefficients associated with B. A new approach based on Compressive Sensing (CS) framework which is a theory that one may achieve an exact signal reconstruction from sufficient CS measurements taken from a sparse signal is proposed in this paper. Wavelet-based contourlet transform, block-based random Gaussian image sampling matrix and projection-driven compressive sensing recovery are cooperating together in the new process framework to accomplish image reconstruction. Smoothing is achieved via a Wiener filter incorporated into iterative projected Landweber compressive sensing recovery, yielding fast reconstruction. Different kinds of images are tested in this paper, including normal pictures, infrared images, texture images and synthetic aperture radar (SAR) images. The proposed method reconstructs images with quality that matches or exceeds that produced by those popular ones. Also smoothing was imposed with the goal of improving the quality by eliminating blocking artifacts and quality of reconstruction with smoothing is better to that from pursuits-based algorithm.

Keywords: block-based compressive sensing; contourlet transform; projection driven recovery; smoothing reconstruction.

1. Introduction

The Shannon-Nyquist sampling theorem (Abdul,1977) claims that one must sample at least two times faster than the signal bandwidth while capturing it without losing information. Many signal processing applications require the identification and estimation of a few significant coefficients from a high-dimensional vector. There is an extensive body of literature on image compression, but the central concept is straightforward: we transform the image into an appropriate basis and then code only the important expansion coefficients. The crux is finding a good transform, a problem that has been studied extensively from both a theoretical and practical standpoint. The most notable product of this research is the wavelet transform; switching from sinusoid-based representations to wavelets marked a watershed in image compression and is the essential difference between the classical JPEG (Pennbaker and Mitchell,1993) and modern JPEG-2000 standards (Skodras, Christopoulos and Ebrahimi, 2001). Image compression algorithms convert high-resolution images into a relatively small bit streams (while keeping the essential features intact), in effect turning a large digital data set into a substantially smaller one. Compressive sensing (CS) (Donoho, 2006) is a way to avoid the large digital data set to begin with and that can build the data compression directly into the acquisition.

Compressive sensing (CS), proposed by Donoho (2006), Emmanuel Cand’es (2006) and Michael Elad(2007) et al, is a new developing novel theory that permits, under certain conditions, signals to be sampled at sub-Nyquist rates via linear projection onto a random basis while still enabling exact reconstruction of the original signal. As applied to 2D images, however, CS faces several challenges including a computationally expensive reconstruction process and huge memory required to store the random sampling operator. Recently, several fast algorithms, as mentioned by Figueiredo, Nowak, Wright (2007), Do, Gan, Nguyen, Tra (2008), Haupt and Nowak (2006), have been developed for CS reconstruction, while the latter challenge was addressed in the works (Gan, 2007) using a block-based sampling operation. Projection-based Landweber iterations were proposed to accomplish fast CS reconstruction while simultaneously imposing smoothing with the goal of improving the reconstructed-image quality by eliminating blocking artifacts.

In this work, we propose this same basic framework of block-based CS sampling to replace the traditional sampling and compressing process. We will address their advantages and merits in detail, and give the comparisons with the classical JPEG and modern JPEG-2000 standards. Different transforms (e.g., wavelet transform, contourlet transform and cosine transform) and different kinds of images (including normal pictures, infrared images, texture images and synthetic aperture radar (SAR) images) are tested in the experiments. Results shows that CS reconstruction which is based on directional transform (contourlet transform) outperforms equivalent reconstruction using common wavelet and cosine transforms.
We arrange the paper as follows: the background of Compressive Sensing are depicted in section 2; in section 3 we provide the contourlet transform (Do and Vetterli, 2005) and sampling strategy; the thresholding parameter associated with projection Landweber recovery algorithm is discussed in section 4. And the experiments and comparisons are given subsequently. Finally we summarize our results and prospect our future work.

2. Background of Compressive Sensing

Consider a real-valued, one-dimensional, discrete-time signal \( x(n) \), \( n=1,2,\ldots,N \), and an orthonormal basis, represented in terms of \( N \times 1 \) vectors \( \{\psi_j\}_{j=1}^N \) with the vectors \( \{\psi_j\} \) as its columns, thus \( x = \sum_{j=1}^N s_j \psi_j \) or \( x = \psi s \) can be obtained, where \( s \) is the \( N \times 1 \) vector of coefficients \( s_j = \langle x, \psi_j \rangle = \psi_j^T x \). Supposing \( K \) elements of the \( s_j \) coefficients are nonzero or largest and \( (N-K) \) are zero or negligible, \( x \) is called \( K \)-sparse or compressible, in which case we are interested. Quite often, the requisite sparsity will exist with respect to some transform \( \mathcal{V} \).

According to a common linear measurement course \( y_i = \langle x, \phi_i \rangle \), the inner products between \( x \) and \( \{\phi_i\}_{i=1}^M \) can be calculated. Thereafter a compressed signal representation is acquired directly by compressive sensing, which is indeed only \( M \) dimensions thus avoiding \( N \) samples in data acquisition system. By substituting \( \Psi \) and arranging \( \Psi^T \) as rows in an \( M \times N \) matrix \( \Phi \) and consequently \( \Theta = \Phi \Psi \) being an \( M \times N \) sensing matrix, the measurements \( y \) can be written as follow:

\[
\begin{align*}
y = \Phi x = \Phi \Psi s &= \Theta s
\end{align*}
\]

How we recover the signal? In this case, the key to CS recovery is the production of a sparse set of significant transform coefficients \( s \).

However, the CS theory tells us that when the matrix \( \Theta = \Phi \Psi \) has Restricted Isometry Property (RIP) (Baraniuk, Davenport, DeVore and Wakin, 2008), then it is indeed possible to recover the \( K \) largest coefficients \( s \). The RIP is closely related to an incoherency property between \( \Psi \) and \( \Phi \), where the rows of \( \Phi \) do not provide a sparse representation of the columns of \( \Psi \) and vice versa. The RIP and incoherency hold for many pairs of bases, including for example, delta spikes and Fourier sinusoids, or sinusoids and wavelets. When RIP holds, the idea recovery procedure searches for the \( s \) with the smallest \( \ell_0 \) norm consistent with the observed \( \hat{y} \).

\[
\hat{s} = \arg \min \|s\|_0 \text{ such that } \Theta s = \hat{y}
\]

This optimization will recover a \( K \)-sparse signal with high probability, but unfortunately solving formula (2) is a non-deterministic polynomial (NP) hard problem, and several alternative solution procedures have been proposed. Perhaps the most prominent of these is basis pursuit (BP) (Chen, Donoho and Saunders, 1998) which applies a convex relaxation to the \( \ell_0 \) norm problem resulting in an \( \ell_1 \) norm optimization.

\[
\hat{s} = \arg \min \|s\|_1 \text{ such that } \Theta s = \hat{y}
\]

Although BP can be implemented effectively with linear programming, its computational complexity is often high, leading to recent interest in reduced complexity relaxations as well as in greedy BP variants, including matching pursuits (MP), orthogonal matching pursuits (OMP) and sparsity adaptive matching pursuits (SAMP). Such algorithms significantly reduce computational complexity at the cost of lower reconstruction quality.

As an alternative to the pursuits class of CS reconstruction, techniques based on projections have been proposed already (Haupt and Nowak, 2006). Algorithms of this class form \( \hat{s} \) by successively projecting and thresholding; for example, the reconstruction in 0 starts from some initial approximation \( \hat{s}^{(0)} \) and forms the approximation at iteration \( i+1 \) as follows:

\[
\begin{align*}
\hat{s}^{(i)} &= \hat{s}^{(i)} + \frac{1}{\gamma} \Psi \Phi^T (\hat{y} - \Phi \Psi \hat{s}^{(i)}) \\
\hat{s}^{(i+1)} &= \begin{cases} 
\hat{s}^{(i)}, & \|\hat{s}^{(i)}\|_0 \geq T^{(i)} \\
0, & \text{else}
\end{cases}
\end{align*}
\]

Here, \( \gamma \) is a scaling factor uses the largest eigenvalue of \( \Phi^T \Phi \), while \( T^{(i)} \) is a threshold set appropriately at each iteration. It is straightforward to see that this procedure is a specific instance of a projected Landweber (PL) algorithm. Like the greedy algorithms of the pursuits class, PL-based CS reconstruction also provides reduced computational complexity. Additionally, and perhaps more importantly, the PL formulation offers the possibility of easily incorporating additional optimization criteria (Mun and Fowler, 2009).

3. Contourlet Transform and Sampling Strategy

3.1 Contourlet Transform

Two prominent families of such directional transforms are contourlets and complex-valued DWT. The contourlet transform preserves interesting features of the traditional DWT, namely multiresolution and local characteristics of the signal, and, at the expense of a spatial redundancy, it better represents the directional features of the image. The directional filter bank (DFB) is the key tool to capture the high frequency elements of images whereas grips low frequency elements poorly.

Fig.1. Pyramid Directional Filter Bank (PDFB)

Therefore, a multi-resolution scheme is combined to remove them before the DFB, and thus Laplace pyramid (LP) is considered to allow further sub-band decomposition to be affected on its bandpass images which can be nourished into a DFB to efficiently track down the directional information. We call it pyramidal directional filter bank (PDFB) in respect that
LP is iterated frequently on the coarse image while DFB holding the fine image allowing for a different number of directions at each scale. The illustration is showed in Fig.1. It has the redundancy of 33% owing to the LP, and a perfect reconstruction.

3.2 Sampling strategy

There are numerous matrices working well with compressive sensing framework. It is somewhat surprising that the measurement matrix can be a random, noise-like matrix, for example i.i.d Bernoulli or Gaussian random variables. And in many cases the measurement matrix is a casual, quasi-Toeplitz matrix.

In block-based compressive sensing (BCS), the two dimensional image is divided into M_b×M_b blocks and sampled with an ordinary random Gaussian matrix in our experiments. That is, suppose that x_i is a vector representing, in raster-scan fashion, block i of input data X. The corresponding y_i is then

\[ y_i = \Phi_B x_i \]  

(6)

Where \( \Phi_B \) is an \( M_B \times B^2 \) orthonormal measurement matrix with \( M_B = \lceil \frac{M}{N} B^2 \rceil \), \( \lceil \cdot \rceil \) is to round down to the nearest integer.

\[ \Phi = \begin{bmatrix} \Phi_B & 0 & \cdots & 0 \\ 0 & \Phi_B & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \Phi_B \end{bmatrix} \]  

(7)

Using BCS rather than random sampling applied to the entire raw data X has several merits. First, the measurement operator \( \Phi_B \) is conveniently stored and employed because of its compact size. Second, the encoder does not need to wait until the entire image is measured, but may send each block after its linear projection. Last, an initial approximation X with minimum mean squared error can be feasibly calculated due to the small size of \( \Phi_B \). Here we employ blocks of size \( B=32 \).

4. Thresholding Parameter

As originally described in the work (Haupt and Nowak, 2006), projection Landweber (PL) algorithm used hard thresholding in the form of (5). To set a proper \( T \) for hard thresholding, we employ the universal threshold method of soft-thresholding. Specifically, in (5),

\[ T^{(i)} = \lambda \sigma^{(i)} \sqrt{2 \log K} \]  

(8)

Where \( \lambda \) is a constant control factor to manage convergence, and \( K \) is the number of the transform coefficients. As in 0, \( \sigma^{(i)} \) is estimated using a robust median estimator,

\[ \sigma^{(i)} = \frac{\text{median}(|z_i^{(0)}|)}{0.6745} \]  

(9)

Hard thresholding inherently assumes independence between coefficients. However, bivariate shrinkage is better suited to directional transforms in that it exploits statistical dependency between transform coefficients and their respective parent coefficients, yielding performance superior to that of hard thresholding. A non-Gaussian bivariate distribution was proposed for the current coefficient and its lower-resolution parent coefficient based on an empirical joint histogram of DWT coefficients. However, it is straightforward to apply this process to any transform having a multiple-level decomposition, such as the directional transforms we consider her. Specifically, given a specific transform coefficient \( \xi \) and its parent coefficient \( \xi \) pin the next coarser scale, the \( T(\cdot) \) operator in PL is the MAP estimator of \( \xi \),

\[ T(\xi, \lambda) = \frac{(\xi^2 + \lambda^2)^{\lambda}}{\sqrt{\pi^2 + \lambda^2}} \cdot \xi \]  

(10)

Where \( (f)_+ = 0 \) for \( f < 0 \) else \( (f)_+ = f \). \( \sigma^{(i)} \) is the median estimator of (9) applied to only the finest-scale transform coefficients; and, again, \( \lambda \) is a convergence-control factor. Here, is the marginal variance of coefficient \( \xi \) estimated in a local 3×3 neighborhood surrounding \( \xi \) as in the paper (Endur and Selesnick, 2002).

5. Experimental Result

To evaluate the proposed method on image reconstruction, we deploy several experiments within the BCS framework coupled with PL recovery algorithm.

Tests are processed on Core 2 CPU 2.53 GHz 2.53 GHz, 2.00GB memory computer. The test data are including normal pictures, infrared images, texture images (Lazebnik, Schmid and Ponce, 2005) and synthetic aperture radar (SAR) images (Cumming and Wong, 2005). Size of samples is set to 512×512. Samples are showed in Fig.1.
5.2 results in different transform

The transform domain is the important component of BCS framework. Whether data are sparse or not is due to the transform domain. Here we arrange three different sparse domains to test how they affect the quality of reconstructed image. The three transform methods are DCT, DWT and contourlet transform respectively. Table 1-4 compare PSNR for several different transform at several measurement ratios, M/N.

The results indicate that the method proposed in this paper with contourlet transform achieves the best performance while comparing to DCT and DWT at most situation. When processing normal pictures (e.g., Lena, camera man), even the ratio is 0.1, the PSNR is still bigger than 27 dB. The quality of infrared images is best, though the performance of recovering texture images are not as good as others, the PSNR of texture images are still bigger than 22 dB.

Fig. 3. Lena images reconstruct by BCS at several different measurement ratios.

<table>
<thead>
<tr>
<th></th>
<th>Sampling Ratio (M/N)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.1</td>
</tr>
<tr>
<td>DCT (dB)</td>
<td>26.9312</td>
</tr>
<tr>
<td>DWT (dB)</td>
<td>26.7073</td>
</tr>
<tr>
<td>contour-let (dB)</td>
<td>27.0130</td>
</tr>
</tbody>
</table>

Table 1: PSNR of normal picture construction

<table>
<thead>
<tr>
<th></th>
<th>Sampling Ratio (M/N)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.1</td>
</tr>
<tr>
<td>DCT (dB)</td>
<td>30.5914</td>
</tr>
<tr>
<td>DWT (dB)</td>
<td>31.8296</td>
</tr>
<tr>
<td>contour-let (dB)</td>
<td>22.6698</td>
</tr>
</tbody>
</table>

Table 2: PSNR of infrared image construction

5.3 compared with other recovery algorithm

There are many compressed sensing (CS) recovery algorithms that have been proposed in recent years. Fig. 4 compares PSNR for Lena constructed by several recovery algorithms, like Compressive Sampling Matching Pursuit (CoSaMP) (Deanna and Tropp, 2001), Orthogonal Matching Pursuit (OMP) (Tropp and Gilbert, 2007), Subspace Pursuit (SP) (Wei and Milenkovic, 2009), Iteratively Reweighted Least Square (IRLS) (Rick and Yin, 2008). Experimental results indicate that the proposed BCS is better than others, though it costs more processing time. PSNR of BCS coupled with PL recovery is about 10 dB bigger that others at every sampling ratios. The fastest recovery algorithm is OMP, and the most comparable algorithm is IRLS.

<table>
<thead>
<tr>
<th></th>
<th>Sampling Ratio (M/N)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.1</td>
</tr>
<tr>
<td>DCT (dB)</td>
<td>21.5634</td>
</tr>
<tr>
<td>DWT (dB)</td>
<td>22.6660</td>
</tr>
<tr>
<td>contour-let (dB)</td>
<td>22.6698</td>
</tr>
</tbody>
</table>

Table 3: PSNR of texture image construction

<table>
<thead>
<tr>
<th></th>
<th>Sampling Ratio (M/N)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.1</td>
</tr>
<tr>
<td>DCT (dB)</td>
<td>24.1353</td>
</tr>
<tr>
<td>DWT (dB)</td>
<td>24.6089</td>
</tr>
<tr>
<td>contour-let (dB)</td>
<td>24.4337</td>
</tr>
</tbody>
</table>

Table 4: PSNR of SAR image construction
Fig. 4. PSNR of Lena reconstructed from different recovery algorithm at several different measurement ratios.

6. Conclusion

Recent theory of Compressive Sensing (CS) provides us with a novel concept that an unknown sparse signal can be exactly recovered with an overwhelming probability even with highly sub-Nyquist-rate samples. These characteristics of CS have attracted many attentions in radar applications. In this paper, we examined the use of recently proposed block-based compressive sensing (BCS) coupled with PL recovery algorithm. Through comparing the quality of different kinds of images, it shows that it works best in infrared images, and when dealing with texture images, the performance of reconstruction is reducing. Other popular CS algorithms are also tested in this paper, results show that the proposed method reconstruction is promoting not only sparsity but also smoothness. The proposed method encourages superior image quality, particularly at infrared images. However, as we can see, lots of work is waiting to be settled because the reconstruction performance has not arrived at its peak yet. As we all know the measurements are not the full information of the resource, sparing samples should always be in accordance with the overall power budget and the waveform design and sampling techniques should consider, hence some kind of reweighing and denoising steps should be introduced to the recovery stage.

7. References


An Integration of Software Engineering Methods and Semantic Technologies for Drafting and Modeling Statutes and Legal Rules

Martin Stabauer\textsuperscript{1}  
Gerald Quirchmayr\textsuperscript{2}  
Johann Höller\textsuperscript{1}

\textsuperscript{1} Department of Data Processing in Social Sciences, Economics and Business  
Johannes Kepler University Linz  
Linz, Austria  
Email: \{martin.stabauer, johann.hoeller\}@jku.at

\textsuperscript{2} Research Group Multimedia Information Systems  
University of Vienna  
Vienna, Austria  
Email: gerald.quirchmayr@univie.ac.at

Abstract

The semantic representation and modeling of legal texts has for a long time been a significant research challenge. While approaches from both, software engineering and semantic modeling, have led to impressive results, some gaps are still remaining. This paper tries to bridge the gap between generalizability and applicability by combining semantic modeling with traditional software engineering processes.

A framework for drafting legislation was implemented in OWL, SWRL, various web-based technologies and Java using the Jena framework. Links to external ontologies were made via semantic relations, following the principles of linked open data. A base model including ontologies, semantic rule sets and additional algorithms were developed and amended by a general development process for implementing diverse fields of law.

The base model and process suggested in this paper were then tested in an extensive case study, which clearly documented the approaches benefits, such as increased efficiency of the modeling process, automatic consistency checking, compatibility with established standards in legal semantics, and the reusability of base classes underlying the developed models. The case study addresses curricula based on Austrian legislation in depth, which are exemplarily covered as a whole.

Keywords: Semantics; legal semantics; legislative texts; curricula; law-making processes; linked data.

1 Motivation and Background

The issue of expressing legal information and rules is traditionally solved by the usage of legislative texts. Developments in the field of semantic modeling have led to the question whether written text really is the one and only optimal solution for the task of making the addressee understand the intentions of the legislator. Extensive research carried out over the past decades has led to very promising results based on formal logic and artificial intelligence technologies.

While early research suggested that the usage of expert systems in law would provide members of legal professions with extensive knowledge and interpretation of law and therefore would reduce dependence on experts and even replace them in certain cases (Susskind 1987), later work especially in the U.S. focused more strongly on technologies like case-based reasoning (e.g. Ashley 1991, Rissland et al. 2003).

Further pioneering work include the publications of Thorne McCarty (McCarty 1977), Marek Sergot and Robert Kowalski (Sergot et al. 1986) and Trevor Bench-Capon (Bench-Capon et al. 1987). Leading French and German research is very adequately described by Bourcieur (1995), Fiedler et al. (1984) and Fiedler & Traummüller (1986), respectively. Other examples of representative research carried out in the 1980s and 1990s are presented in Gordon (1987), Greenleaf et al. (1987) and Van Noortwijk et al. (1991).

The continuing importance of research in this field is shown in Schefbeck (2009). The author claims that structuring legal norms in XML and contextualizing them in RDF or OWL will be the future. From a jurisdictional point of view, the main task will be to describe legal norms and their relations to other norms and context elements, whereas in a global scope there will be the need to model cross-language, normative equivalences. All of that is said to happen in a mostly IT-supported way.

Some major advantages of semantic systems in comparison to the written textual communication of legal texts are listed in Höller & Ipsmiller (2009):

- Better quality of legal norms through faster unveiling of inner contradictions and incompleteness.
- Improvements in the law-making process by common understanding and terminology. Individuals with different levels of information have the chance to learn about certain topics in a very specific way.
- Improvements in general communication. Words often have several meanings, which can lead to misunderstandings and barely comprehensible texts. By contrast, semantic models claim to be unambiguous.
- In case of changes in the legal norms in question an impact analysis can be done more easily and effectively.
A better and easier implementation of IT systems that are based on the legal norms in question due to a more direct transmission of information.

Building on the available literature, this paper now tries to close one of the remaining gaps by presenting a practical approach to modeling both the content of the field of law in question and the legal text, which it is represented with. While not going so far as to try and automatically draft legislation, like described in e.g. Winkels & Den Haan (1995), the proposed framework does enable a model developer to combine the benefits of legal texts with state-of-the-art technologies in software engineering and the advantages of semantic models.

2 Related Work
The research presented in this paper is intending to close the gap between the current approaches of very generic legal ontologies and the traditional means of software engineering, which are usually aimed at very specific issues. Legal semantics were developed with the intention in mind that they should fit every thinkable branch of law and every thinkable field of application. This aspiration requires a high level of abstraction, which in turn results in most of these ontologies not being directly applicable.

One of the most widely recognized variants of these very valuable contributions is LKIF, the Legal Knowledge Interchange Format developed by the EU-funded Estrella Project ("European project for Standardized Transparent Representations in order to Extend Legal Accessibility"). Breuker et al. (2008) describes the 15 modules included, which are meant to keep the system clear and tidy. Figure 1 shows an overview.

LKIF allows legal knowledge bases to be represented in OWL, the Web Ontology Language (McGuinness & Harmelen 2004). This includes specific terminology, rules and normative statements. The rules in LKIF extend the Semantic Web Rule Language standard (SWRL) described in Horrocks et al. (2004), which itself is a combination of OWL and RuleML (Boley et al. 2001).

Apart from this, two generally applicable main categories of IT systems in the field of law have been identified in the early 1980s (e.g. McCarty & Sridharan 1982, Bing 1984): storing and consequently retrieving legal information on the one hand and legal analysis systems on the other. Building on the experience gained by a series of projects in the late 1980s the issue of logic based representations of laws was again stressed in (Kowalski 1995). Concerning text analysis, the works of Erich Schweighofer (e.g. Schweighofer 1999) belong to the most remarkable.

With ontologies being more expressive than previous modeling mechanisms and IT infrastructures offering substantially greater power, some of the ideas introduced by researchers in the 1980s and 1990s are now becoming practically relevant. According to Lauritsen (2013) this leads to a series of interesting new research challenges.

The importance of the field is again underlined by Springer planning to publish a handbook on legal reasoning and argumentation in the near future (Bongiovanni et al. 2015).

3 Semantic Model and Modeling Process
The major issue of models used today is that most of them are either very abstract or not reusable. The lack of reusability often is a result of modeling a specific field of knowledge without considering additional areas of application. How this gap (see Figure 2) is closed by our integrated model is outlined in this section of the paper. While the direct approach using either semantic modeling tools or software engineering methods in an isolated way leads to a series of impasses, the combined model yields the desired results with considerably less effort. By integrating the semantic base model with a software-engineering oriented process model the implementation of various legal regulations becomes possible, reducing the effort through employing generalized base classes and following a guided process. To illustrate the benefits of this strategy, the results of an extensive case study are presented in section 4.

The approach used in this research consists of a semantic base model that can be used to model various fields of law, and a matching development process model to help implement a specific solution for a selected application domain. A supporting framework was developed to help ensuring consistency with existing standards and external ontologies. This framework combines a semantic ruleset with elements of traditional software development processes.

3.1 Semantic Base Model
The semantic base model lays the foundation for the modeling process described in this section. It is constructed as a connection of base classes that are inte-
The underlying formal concept is inspired by the LKIF model briefly described in chapter 2. Depending on the modeled field of law extensions can be added as necessary. In most cases it is these extensions that make the model directly applicable. The effort needed to add them to the base model highly depends on the desired application. The more structured and isolated the specific legal regulation is, the easier it is to be implemented in our base model. The benefit is highest for fields of law where automatic calculations build the focus. An example could be tax law, where the ontology itself could deliver answers to questions like ‘What is the tax payable for individual X?’.

The model includes both the contents of the field of law in question and the conceptual structure. Therefore, it is not only possible to use the legal knowledge covered in the ontology for whatever application is desired, but also to automatically create and restore the semantically correct and complete human-readable legal text, even in multiple languages.

Another feature of the model is that it allows the connection to external ontologies via suitable semantic relations. These links follow the principles of linked open data as referred to in Bizer et al. (2008). An important decision was, which sublanguage of OWL (Hitzler et al. 2012) to select. We decided in favor of OWL DL, as it allows a reasonable compromise between elegant, high-level expressiveness and decidability in the ontology. As future implementations of the base model might not want to use all the expressions included in OWL 2 Full, it seems a good idea to restrict the base model to a certain amount of expressiveness in exchange for guaranteed decidability and the availability of practical reasoning algorithms.

The following figure gives an outline of the base models structure and selected connections to external ontologies.

This base model is used for checking the consistency of specific implementations and for modeling and creating the domain specific legal text, as shown in more detail in the case study in chapter 4.

### 3.2 Automated Creation of Legal Text

The automated creation of legal text is mainly achieved by a specific semantic structure that is used as input for algorithms written in Java. These algorithms then recursively construct a human-readable version of the legal ontology in the form of a classic legal text. As the basic conceptual structure of the modeled law is part of the ontology, it can be used to create the structure and hierarchy of the legal text. Therefore a set of basic classes is introduced, which can be used for a wide range of fields of law.

In essence, these basic classes represent the basic building blocks of legal regulations such as sections, paragraphs, subparagraphs etc. and their hierarchy. All these classes as well as an additional utility class called Content are subclasses of the class Textblock which provides the basic functionalities for numbering and organizing the legal text. These functionalities are modeled via included meta information and relations to other Textblocks.

The following example illustrates the creation of legal and statutory texts using the example of one sample paragraph of a Master curriculum. Figure 4 shows the structure of this paragraph in the ontology, which can be turned into legal text when combined with straightforward Java algorithms. These create the hierarchy of paragraphs, subparagraphs etc. recursively, place the correct objects and relations at the corresponding positions and produce the accurately formatted legal text.

The main algorithms for a textual output within the classes of the legal regulation itself and all sorts of Textblock then include specific formatting functions and a recursion like follows:

```java
StringBuffer sb = new StringBuffer();
Collections.sort(textblocks);
for (Textblock b : textblocks) {
    sb.append(b.toString());
}
```
In the next step a development process model is needed to ensure the satisfaction of requirements to guarantee the executability of the text-generating algorithms, semantic rules and consistency checks.

### 3.3 Development Process

The development process described in this section focuses on guiding a modeler through the development of a domain specific implementable representation of a law or statute. Drawing on experience gained in the field of software development, this process is based on the classic spiral model, described in Boehm (1988). The major advantage of the spiral model in our context is that it covers the necessity of representing continuously expanding legislation in a stepwise approach. Every time there are changes in the respective field of law, a new iteration starts by refining and/or modifying the last iteration.

![Spiral model](image)

**Figure 5: Spiral model**

Figure 5 shows the process model, which consists of the following activities:

- **Requirements analysis**
  
  As a first step, the needs and requirements of all the involved parties need to be analyzed. Often, there are several user groups with diverging technical and juristic understanding. In order to completely capture and consider all these requirements, Bourque & Fairley (2014) suggest the following structure, which is very suitable for our demands:
  
  - Requirements elicitation
  - Requirements analysis
  - Requirements specification
  - Requirements validation

  As a minimum, the requirements of the legislator and the addressee are to be analyzed. However, often there are several more stakeholders involved in the process. Furthermore, legal prerequisites have to be regarded, and there could be special needs like multilingualism, version control, accessibility, etc.

- **System design**
  
  The next step is the design of the overall system with the main objectives of reduction of complexity and minimization of the risk of maldevelopment. For example, the following questions need to be answered when implementing the model in a specific field of law:
  
  - Should the whole field of law be modeled or just parts of it?
  - What is the structure of the model, how are relations to other legal regulations implemented?
  - Are there external sources (e.g. already modeled regulations) that can be used?
  - Which semantic language, which rule language, which programming language shall be used? Which frameworks can be used?
  - Is there a need for one or more user interfaces?

- **Model implementation**
  
  The next task is to develop the semantic model, all required rulesets and user interfaces. The variation between the respective fields of law is greatest in this phase: The outcomes can differ a lot, depending on the specific legal regulation to be implemented. This is due to highly diverse legal prerequisites, different external relations and varying structures. Nonetheless, the base model ensures that certain basic elements are in place so that compatibility is guaranteed.

- **Tests**
  
  One first test is the concretization of the model itself: A specific legal regulation is implemented and the results are validated. A successful implementation can be seen as a sign for a correct base model; this is shown in the case study in chapter 4. In many cases, additional tests are useful, Bourque & Fairley (2014) specify various tests especially designed for the field of software engineering.

- **Maintenance**
  
  There are two main types of maintenance that are of high relevance in this context (IEEE Standard Glossary of Software Engineering Terminology 1990):

  - Corrective and perfective maintenance aims to improve the software, rules and/or ontology by rectifying errors, enhancing algorithms or models or implementing new functional features.
  - Adaptive maintenance tries to cope with changes in the systems environment by modifying the software, rules and/or ontology. Some legal regulations are modified regularly and the implemented model has to handle these changes. This way the models life cycle can be extended in a setting very likely to change.

The modeling process described above will be applied to a challenging and complex case study in the next section of this paper. This case study will test our approach against a hierarchy of different laws and statutes and a highly complicated regulation drafting process.
4 Case Study: Curriculum Development based on Austrian Legislation

As an example for the base model described above a concrete implementation was developed. This implementation acts as proof of concept for the base model. It shows that it is possible to model a whole coherent legal text using an integration of the aforementioned techniques.

4.1 Environment and Setting

The Austrian legislation regarding universities and other institutions of higher education has gone through enormous changes during the last years. A new university law was introduced, giving the universities greater flexibility and full legal capacity.

One of the main changes was the new duty to enact statutes, in which each university applies its own regulations including those regarding curricula. As a consequence, an IT system handling curricula has to cope with all these laws and statutes.

Another important influence of the law-making process of curricula is the so-called Bologna process, which introduced the framework of Bachelor and Master degrees as well as the European Credit Transfer System (ECTS (ETC Users’ Guide 2009)). The ECTS credits standardize study attainment and make the performance of students of higher education comparable across the EU.

4.2 Implementation of the Base Model

As a proof-of-concept of the base model described in chapter 3.1 a new set of ontologies was implemented. It is called CML (‘Curriculum Modeling Language’) and allows modeling curricula based on Austrian legislation. In order to act as an implementation of the base model all of the base classes and relations were included and enhanced in CML.

4.2.1 Ontologies

Table 1 shows the ontologies modeled in CML.

<table>
<thead>
<tr>
<th>Ontology</th>
<th>Content</th>
</tr>
</thead>
<tbody>
<tr>
<td>cml.owl</td>
<td>base structure and imports of external ontologies</td>
</tr>
<tr>
<td>cml_uni.owl</td>
<td>models universities and other institutes of higher education</td>
</tr>
<tr>
<td>cml_curr.owl</td>
<td>models a curriculum and its components</td>
</tr>
<tr>
<td>cml Templ.owl</td>
<td>templates, exemplary data and text, translations</td>
</tr>
<tr>
<td>cml_rules.owl</td>
<td>SWRL rules for automated validation</td>
</tr>
<tr>
<td>cml_ext.owl</td>
<td>mapping to external ontologies</td>
</tr>
</tbody>
</table>

Table 1: CML ontologies

4.2.2 Semantic Rules

CML makes extensive use of SWRL. Three main types of rules are used in CML to validate a modeled curriculum:

- Requirements that must be fulfilled for a curriculum to be valid. E.g. Bachelor curricula compliant with the statutes at Johannes Kepler University Linz have to include at least one course with a gender-related subject and at least 3 ECTS. This can be validated by the following SWRL rule, which categorizes a correct curriculum as CurriculumWithGenderCourse:

  \[
  \text{Curriculum(?curr)} \land \text{Course(?c)} \land \text{indirectSub(?c, ?curr)} \land \text{genderTopic(?c, true)} \land \text{ects(?c, ?ects)} \land \text{swrlb: greaterThan(?ects, 2)} \Rightarrow \text{CurriculumWithGenderCourse(?curr)}
  \]

- Rules that draw the attention of the relevant authorities to certain aspects. E.g. a Bachelor curriculum normally includes 180 ECTS, but this can be changed under certain conditions. The following SWRL rule checks if the sum of ECTS credits of a curriculum matches the standard. If the numbers differ, the rule categorizes the curriculum as CurriculumSpecialECTS:

  \[
  \text{Curriculum(?curr)} \land \text{hasType(?curr, ?type)} \land \text{ects(?curr, ?ects)} \land ?\text{ects(?type, ?standard)} \land \text{swrlb: notEqual(?ects, ?standard)} \Rightarrow \text{CurriculumSpecialECTS(?curr)}
  \]

- The third category consists of rules that cannot be modeled reasonably using semantic technologies. In most cases this is because of the open-world assumption (OWA) that is made in OWL and other languages of the semantic web. The main idea of OWA is that if a statement is not known it is not automatically false but simply unknown. This way one can distinguish false statements from missing statements (Grimm 2010).

Numerous rules in this category are sum-based, like the following example: We want to know, whether the ECTS of a curriculum match the sum of the courses subordinated. Figure 6 shows the issue: In OWA we do not know if the courses really are the only ones subordinated, there could always be another course formerly left out of consideration. A possible solution for this problem would be to add closure axioms, thereby creating an artificial closed-world. This adds a lot of axioms to the ontology and every change has to be done at various places.
Another solution would be the usage of SQWRL, which allows querying an ontology as it could be done in a closed-world environment (O’Connor & Das 2009). In this case we use SQWRL to sum up the ECTS of the courses and compare them with the curriculum.

\[
\text{Curriculum}(\text{?curr}) \land \text{Course}(\text{?c}) \\
\land \text{sub}(\text{?c}, \text{?curr}) \land \text{ects}(\text{?curr}, \text{?currents}) \\
\land \text{ects}(\text{?c}, \text{?ects}) \land \text{sqwrl: sum}(\text{?sum}, \text{?ects}) \\
\land \text{swrlb: notEqual}(\text{?sum}, \text{?currents}) \\
\Rightarrow \text{CurriculumWrongECTS}(\text{?curr})
\]

As SQWRL is not yet fully supported by some of the most widely recognized frameworks, our approach makes use of simple Java algorithms to perform the desired checks.

We now have demonstrated the implementation of different categories of semantic rules needed to validate a modeled curriculum. The next section covers an exemplary GUI developed to make life easier for the various users of the CML ontology. Understandably, such GUIs are not a necessity in all thinkable applications of the base model.

4.2.3 Graphical User Interfaces

Although it is possible to create and edit a model conform to CML by using generic tools like Protége (Protége 2014), their complexity and wide range of functions might confuse the not too advanced users. There are numerous groups of users, all of which have been determined during the requirements analysis and who have their own needs, technical and juristic understanding. This arises the necessity for specific user interfaces for each user group.

One exemplary GUI that has been developed for the study commissions in charge of creating a new curriculum from scratch is the Curriculum Designer. This web-based piece of software is written in Java and JSP and allows the modelers of curricula to create the basic building blocks of a new curriculum and output a coherent and complete model in OWL. Figure 7 shows a screenshot of the Curriculum Designer.

![Figure 7: Curriculum Designer](https://jku.at/curriculumdesigner)

4.3 Results

The results of the case study are very promising. We turned several curricula into semantic models and back into legal texts that turned out to be equal to the original material and therefore appear to be compatible with human-readable text. The basic structure including the most important information of a typical Austrian Master curriculum including around 300 to 400 course classes results in an ontology of around 4000 to 5000 axioms, which can easily be handled by the algorithms and frameworks described. Naturally, the number of axioms increases as more information regarding courses or more comprehensive rulesets are added or when additional languages are included. However, this increase is moderate and happens linearly, so that no performance issues arose at any time.

5 Conclusion

In this paper we have presented an approach for modeling legal texts based on an integration of semantic models and traditional software engineering technology. This has allowed us to bridge the gap between abstract and often too generic representations and very specific implementations that have little value for reuse. In the case study the applicability to a very complex domain was shown that comprises constitutional legislation, simple laws, administrative statutes and local university regulations. Hence, this experiment has also demonstrated the ability of our approach to cope with different hierarchies of legislative and statutory materials.

Eventually, the major contribution of the research described in this paper is the provision of a tool-supported method that significantly simplifies the representation of legislation in an executable form and also increases the reusability of the developed models.

References


for large scale applications in law: A formalisation of supplementary benefit legislation, in ‘Proceedings of the 1st international conference on Artificial intelligence and law’, ACM.

Bing, J. (1984), Handbook of legal information retrieval, Elsevier Science Inc.


Schema Mapping Using Hybrid Ripple-Down Rules

Sarawat Anam\textsuperscript{1,2}, Yang Sok Kim\textsuperscript{1}, Byeong Ho Kang\textsuperscript{1} and Qing Liu\textsuperscript{2}

\textsuperscript{1}(Sarawat.Anam, YangSok.Kim, Byeong.Kang)@utas.edu.au
School of Engineering and ICT, University of Tasmania
Sandy Bay, Hobart, Tasmania, Australia

\textsuperscript{2}Q.Liu@csiro.au
Autonomous Systems, Digital Productivity and Service Flagship,
CSIRO Computational Informatics, Hobart, Tasmania, Australia

Abstract

Schema mapping is essential to manage schema heterogeneity among different sources. Schema mapping can be conducted by using machine learning algorithms or by knowledge engineering approaches. These two approaches have advantages and disadvantages. The machine learning approaches can learn their model using the data, but they are static, so they cannot be modified to reflect the domain data changes. Inversely, the knowledge engineering approaches need domain experts, but they can be modified by reflecting the domain data changes. In order to exploit the advantages of both approaches and reduce the limitations, we propose a hybrid approach, called Hybrid-RDR, which combines a machine learning algorithm with ripple-down rules (RDR), an incremental knowledge engineering approach. A model is constructed by a decision tree algorithm and then it is extended by adding rules incrementally. This approach achieves higher performance in terms of precision, recall and F-measure compared to the machine learning algorithm. This significantly reduces the effort for classifying the related schemas one by one by manually creating rules and it is possible to modify the knowledge base by adding rules without creating model again if decision tree gives wrong classifications whenever the schema data changes over time.

Keywords: Machine learning algorithm, Knowledge engineering approach, schema mapping, incremental learning.

1 Introduction

Schema mapping is a set of logical specifications that express correspondences between semantically related schemas of different datasets through the application of a matching algorithm (Shvaiko and Euzenat, 2005). Schema mapping is used in many application domains such as data integration, data exchange, data warehousing and schema evolution (Glavic et al., 2010). Schema matching can be done in the element level and structure level. While the element level matching considers only names of the schemas, the structure level matching uses the result of element level matching for matching the full graph. In this research, we only focus on the element level matching since our major aim is to evaluate whether our proposed approach can be applied to the schema mapping problem and our approach can easily be extended by considering the structure information.

In our research, the schema mapping process is defined as:

\[ M = c(S_i, T_j, V_{ij}), \]

where \( S_i \) is a source schema, \( T_j \) is a target schema, \( V_{ij} \) is an attribute value vector (\( V_{ij} = \{v_1, v_2, ..., v_n\} \)) and \( M \) is a Boolean output (if \( S_i \) is matched with \( T_j \), return true; otherwise return false). The attribute values are derived by applying different similarity functions to \( S_i \) and \( T_j \), and to the values of \( S_i \) and \( T_j \) after text pre-processing (e.g., synonym, tokenisation, and reverse abbreviation). Therefore, the schema mapping problem is to find the classification function \( c \) that accurately predicts a real relation between two schema elements.

Many schema mapping systems (Do and Rahm, 2002a, Aumueller et al., 2005, Doan et al., 2002, Marie and Gal, 2008) have been developed by employing machine learning algorithms and/or knowledge engineering approaches. A machine learning algorithm needs training data set for building models, but usually it is very difficult to obtain fine training datasets. In addition, it is very difficult to change the model by human knowledge. A knowledge engineering approach encodes human knowledge directly, such that the knowledge base can be constructed with limited data, but it requires time-consuming knowledge acquisition.

Schemas can be created over time incessantly, and thus knowledge also changes over time. Machine learning and knowledge engineering approaches manage this phenomenon differently. Machine learning algorithms usually reconstruct their model after collecting sufficient data, while knowledge engineering approaches acquire new knowledge if necessary.

In this research, we propose a hybrid approach that constructs classification model using a machine learning algorithm and maintains new knowledge using a knowledge engineering approach, called ripple-down rules. When new data are available, the classification model may suggest wrong classification for some cases. In this case, it is necessary to add a rule, called censor rule, which stops the wrong classification and to add a
rule, called alternative rule, which correctly classifies the problem case. When a case is stopped by a censor rule but have no alternative rule that correctly classifies it, the case has no conclusion. In this case, a new rule that classifies the case correctly should be as a child rule of the root node.

In order to simulate the changes of the data, we assume that the data for schema mapping is partially available at the beginning and new data will be added onward. While the hybrid approach improves the performance of schema mapping by adding rules incrementally for correcting schema mapping errors of the current knowledge base, the decision tree improves its performance by learning a new model by including newly available data. Our experimental results show that the Hybrid-RDR approach produces slightly higher performance compared to the decision tree.

2 Related Works
Machine learning techniques have been used in the context of schema matching. Learning Source Descriptions (LSD) (Doan et al., 2001) is a schema mapping system and the extension of LSD is GLUE (Doan et al., 2002) which creates ontology mapping. Both systems use machine learning techniques like Multi-strategy learning approach as base learner, Naïve Bayes for classifying text, and Meta learner for finding matching among a set of instances. Embley et al. (2004) develop an approach based on learning rules of decision trees for discovering hidden mapping among entities. In this approach, the rules are used for matching terms in WordNet. However, the decision trees are not used for choosing the best match algorithms. Duchateau et al. (2008) present an approach for schema matching which uses a decision tree to combine the best suitable match algorithms. The approach inputs a set of schemas and a decision tree which is composed of match algorithms, and outputs a list of mappings which are validated by experts to find out whether the matching is correct or not. The feedback is used to feed into another decision tree for learning. The machine learning techniques generally require refined training dataset that should be prepared by largely in manual and the techniques cannot easily change its model without sufficient data. In the rule based approach, well defined training dataset is not necessary. In the approach, schema matching is started for a small amount of schema data by adding rules.

Some systems have already used rule based techniques for schema matching. Among them, COMA/COMA++ (Do and Rahm, 2002b, Aumueller et al., 2005) are generic schema and ontology matching systems where simple, hybrid and reuse oriented matchers are used. In the systems, schemas are internally encoded as DAGs (Directed Acyclic Graphs) and are analysed using string matching algorithms. Different aggregation functions such as average, minimum, maximum and weighted sum, and rule based techniques are used in the systems for obtaining combined match results. However, in COMA/COMA++, determining best combination of matcher is not easy. YAM (Duchateau et al., 2009) is a machine learning based schema matching factory. In the learning phase, YAM considers users’ requirement such as a preference for recall or precision, provided expert correspondences. It uses a Knowledge Base (KB) that consists of a set of classifier, a set of similarity measures, and pairs of schemas which have already been matched. In the matching phase, the KB is used to match unknown schemas. However, in the system, appropriate classifier is selected by users or to use a default classifier learned over a huge mapping knowledge base. In this research, we use the hybrid approach combining decision tree and rule based technique. In our system, the KB is empty at the beginning, and the first rule is added in the KB by classifying a dataset using decision tree learning model. Then rules are added incrementally in order to solve schema matching problems such as un-classifications and wrong classifications.

Traditional rule-based systems require time-consuming knowledge acquisition as in those systems a highly trained specialist, the knowledge engineer, and the time-poor domain expert are necessary in order to analyze domain (Richards, 2009). In order to solve the problem of time consuming knowledge acquisition, we adopt RDR (Ripple Down Rule) (Compton et al., 1991), a well-known incremental knowledge acquisition method. RDR has been successfully applied in many practical knowledge-based system developments. There are several versions of RDR methods, including Single Classification RDR (SCRDR), Multiple Classifications RDR (MCRDR), and Nested RDR. MCRDR is used in order to solve problems in some domains, e.g., pathology, text/web document classification, help desk information retrieval and medication review (Richards, 2009). Since our research aims to find matching relationship of schema (matched or not-matched), SCRDR is chosen for our research.

The success of RDR does not depend on representational differences; rather it largely depends on its distinctive operational semantics on standard production rules (SPR). SPR has the form \( p \rightarrow a \), which is interpreted as “if a case satisfies condition \( p \) then do action \( a \)”. RDR systems in general process cases sequentially and whenever the current knowledge base suggests wrong conclusions, new rules are added. Whenever a new rule is created, it is necessary to validate the rule normally by checking whether or not the future cases are given the correct classifications. If any case is wrongly classified by a rule, then RDR systems acquire exception rules for this particular rule. In this case, the expert directly refines the new rule adding conditions until all incorrect cases are removed. However, it is not easy to construct this kind of rule with resource constraints such as limited time and information. We use CPR (censored production rules) based RDR (Kim et al., 2012), to be used for acquiring exceptions when a new rule is created using censor conditions. CPR has the form \( p \rightarrow a \rightarrow c \), which is interpreted as “if a case satisfies condition \( p \) then do action \( a \) unless the case does not satisfy the censor conditions \( c \)”.

This approach also can provide multiple cornerstone cases that satisfy the main condition clause (positive cornerstone cases) as well as the censored condition clause (negative cornerstone cases). The approach is useful when we have a large number of validation cases at hand.
3 Method

In this section, we describe our proposed Hybrid-RDR approach used for schema mapping.

Hybrid-RDR

Hybrid-RDR approach combines a decision tree algorithm, J48 with CPR based RDR. The process of Hybrid-RDR is given in Fig.1. The process is described below:

Any machine learning classification algorithms such as Naïve Bayes, and decision tree, can be used for our hybrid approach. Among them, we choose the decision tree algorithm because its model can be understood by the human expert. A decision tree is a tree whose internal nodes represent the attributes, and the edges represent the conditions on the result of the attributes. All the leaf nodes represent classifications which are either true or false, indicating whether there is a match or not. We use J48 decision tree, a Java implementation of C4.5 in WEKA (Quinlan, 1993), for classifying schemas. Decision tree inputs a collection of cases (training set) where each case contains a set of features obtained from input schemas and from the application of string similarity metrics and text processing techniques to the input schemas, and also from manually providing class level (true or false). Then it builds up a classification model. The accuracy of the model is evaluated by using a test dataset.

For the knowledge engineering, we chose RDR approach since it overcomes “knowledge acquisition bottleneck” problem of the conventional knowledge engineering approaches by employing error-driven knowledge maintenance strategy, where all rules have clear relationship and they are added as either exception or alternative of the existing rules (Compton et al., 1991). RDR has been successfully applied in many application domains (Richards, 2009). The structure of Knowledge Base (KB) is designed as an n-tree. Each node of the tree is a rule and each rule consists of IF [conditions] THEN [conclusion] UNLESS [censor-condition]. The KB structure and the inference process are shown in Fig.2.

Fig.2 Hybrid-RDR Knowledge Base

In Fig.2, at the beginning when KB is empty, we define R0 (rule 0), which is always true. We denote the first level rules by R1, R2, R3 and R4 and the censored rules by C1, C2, and C3. In the approach, the first rule is added in the KB by classifying a dataset using decision tree classification model, ML. (machine Learning). Then other rules are added incrementally when schema data changes over time.

The inference process is based on searching the KB represented as a decision list with each decision possibly refined again by another decision list. Once a rule is satisfied by any case, the process evaluates whether or not the censor conditions are matched to the given case. If any censor rule is not satisfied, then the process stops with one path and one conclusion. However, if any censor rule is satisfied, other rules below the rule that was satisfied at the top level is evaluated. The process stops when none of the rules can be satisfied by the case in hand.

Knowledge acquisition is a process which transfers knowledge from human experts to knowledge based systems. Knowledge acquisition process can be divided into three parts. Firstly, a correct classification should be decided by the expert. Secondly, new rules’ locations should be specified by the system. Thirdly, new rule’s condition should be decided by the expert. If the current knowledge base suggests wrong classification, it is necessary to add a censor rule that has NULL as classification. If the current knowledge base suggests no classification for any case, a new rule should be added as an alternative rule, which is added as a child rule of the root node of the knowledge base. The cases used for creating rules are called cornerstone cases and they are used in consequent knowledge acquisition process (Compton and Jansen, 1990).

The advantage of Hybrid-RDR approach is that only one classification model is created by decision tree for a small amount of schema data and knowledge base is then built incrementally by adding rules to solve schema matching problems. The process helps to reduce time in two ways. Firstly, it does not create classification model when schema data changes over time. Secondly, it does not classify all the related schemas one by one by manually creating rules. The Hybrid-RDR approach is useful where there are large numbers of validation cases at hand.
4 Experimental Design

4.1 Datasets

Four XDR schemas of purchase order domain, such as EXCEL, CIDX, NORIS, and PARAGON, obtained from www.biztalk.org are used for this evaluation study. We denote the schema datasets EXCEL, CIDX, NORIS, and PARAGON by E, C, N, and P respectively. These schema datasets are used for some schema matcher evaluation (Do and Rahm, 2002b). These schema datasets contain different types of features such as identical words, combined words, abbreviated words and synonym words. Each schema dataset contains 35 (E), 30 (C), 46 (N), and 59 (P) schema names.

4.2 Experimental Procedure

In this research, we experiment ten matching tasks one by one. For this, we take all the combinations of six schema datasets such as E-C (first combination is to deal with two datasets, EXCEL and CIDX), E-N, E-P, C-N, C-P and N-P. Then we take the Cartesian product of the six schema datasets separately. The sizes of Cartesian product of the datasets are 1050 (E-C), 1610 (E-N), 2065 (E-P), 1380 (C-N), 1770 (C-P) and 2714 (N-P) entity pairs. We combine all the entity pairs and get total 10589 entity pairs. We randomly divide the entity pairs into ten for creating ten datasets (D1 to D10) where datasets D1 to D8 and D10 contain 1058 entity pairs and D9 dataset contains 1067 entity pairs. These ten datasets are used for ten matching tasks. In order to use the datasets for classification and to give proper knowledge to the users for creating rules, we construct attributes as follows:

Attributes Construction. In order to give proper knowledge to the users, attributes are constructed in three steps:

- The input schema names (source and target);
- Application of text processing approaches such as tokenization, abbreviations and acronyms expansion, and synonym lookup on the input schemas. In tokenization and word separation, schema names containing multiple words are split into lists of words by a customizable tokenizer using punctuation, uppercase, special symbols, whitespace and digits. For instance, “contactEmail” is split into “contact” and “Email”. Abbreviations and acronyms are expanded by using external resources such as a dictionary and/or a thesaurus. For instance, “tel” is expanded into its original form “telephone”. For this, we use the abbreviation file created for COMA (Do and Rahm, 2002b). Synonym processing is applied to use semantically identical schema names to measure similarity (e.g., ‘Invoice’ is semantically same as ‘Bill’ in purchase order domain). We use the synonym file created for COMA (Do and Rahm, 2002b);
- Application of the string similarity metrics on the features of the attributes computed from step 1 and 2, which creates another attributes. We use string similarity metrics developed by two open source projects. For Levenshtein, JaroWinkler, Jaro Measure, TFIDF and Jaccard, we use open source library SecondString\(^1\) and for Monge-Elkan, Smith-Waterman, Needleman-Wunsch, Q-gram and Cosine we use SimMetric open source library\(^2\). Similarity values are normalized, such that the value within from 0 to 1, where 0 means strong dissimilarity and 1 means strong similarity. The threshold values for deciding schema matching (true/false) are increased with 0.1 from 0 to 1. We also provide class level (true or false) manually which creates another attribute. In such a way we get 73 attributes by using schema information of two datasets (one matching task). Computed attributes represent knowledge about a relation between attributes, operator or process patterns. After preparing the attributes and the schema data under the attributes, all these are fed in to the dynamic decision tree algorithm and the Hybrid-RDR. The dynamic decision tree algorithm learns a new model by including newly available data. The evaluation approach is shown in Fig.3.

In the evaluation approach, we randomly select datasets for training and testing. For example, we select D1 for training and D10 for testing.

Dynamic ML. In the dynamic machine learning approach, we create decision tree model, \(ML_0\) for D1 and test D10. Then we incrementally add other datasets like D1+D2, D1+D2+D3 for creating decision tree models, \(ML_1, ML_2\) and test D10. In this way, we add all nine datasets for creating decision tree model, \(ML_9\) and test D10.

Hybrid-RDR. In Hybrid-RDR approach, we create decision tree model, \(ML_0\) for D1 and test D10. We also test D2 and find some un-classified cases and wrong classified cases. Then we refine the decision tree rule by adding censor rules, \(Rule_0\) and again classify the cases by adding alternative rules, \(Rule_3\). The censor rules are added as censor nodes of decision tree in the KB and alternative rules are added as parent rules in the KB. The \(ML_0+Rule_0\) is then used for testing D10 and also for testing D3. We add rules, \(Rule_4\) again for the wrong cases.\(^3\)

\(^1\)http://secondstring.sourceforge.net
\(^2\)http://sourceforge.net/projects/simmetrics
classified cases of D3, and $ML_0 + Rule_0 + Rule_1$ is used for testing D10. In such a way, we incrementally add rules for all nine datasets, $ML_0 + Rule_0 + Rule_1 + \ldots + Rule_9$ and test D10. The detail rule creation process for schema mapping is described in the following:

**Schema Mapping by Hybrid-RDR.** A simple GUI (Graphical User Interface) is created which can select any datasets from repository. The attributes that are created by the above steps of Attribute construction are represented in a “Case Browser” to provide sufficient knowledge to the users (Fig. 4). The system works in two phases: Training phase and classification phase. In the training phase, “Training by DT” of Fig. 4 is used. We use the button in order to train one dataset using decision tree, J48. The attributes which are created by the above steps of Attribute construction, are used as training sample to build a model. The purpose of building a model is to classify whether a given entity pair of schema names is matched or not based on their feature similarity measure. For all machine learning techniques, we consider 10-fold cross validation. 10-fold cross validation means that the data is split into 10 groups where nine groups are considered for training and the remaining one group is considered for testing. This process is repeated for all 10 groups. In the classification phase, “Classify” button of Fig. 4 is used. For matching entity pair of schema names using the algorithm, we provide the attributes created from another datasets. Finally, we get the matching results as true positive (if reported match by expert is true and predicted match by algorithm is true), false positive (if reported match is false and predicted match is true), true negative (if reported match is false and predicted match is false) and false negative (if reported match is true and predicted match is false) which are displayed in Fig. 4.

![Fig. 4. GUI represents 73 attributes with schema names (all the attributes are not visible)](image)

In order to solve the problem of false negative and false positive (wrong classifications), we use “Edit Classification” of Fig. 4. “Edit Classification” button helps to refine the wrong classified cases by adding new conditions until all incorrect cases are removed or creating another new rule using knowledge Acquisition GUI. Classification for the censor rule is always “NULL”. For editing classification, the Knowledge Acquisition GUI is displayed in Fig. 5.
In Fig. 5, parent condition is decision tree which gives the wrong classification for the current case. In order to edit the parent rule, it is not necessary to select the classification as classification for the censor rules is always “NULL”. First, the rule conditions are added. For each condition in the rule, the attribute, operator, and value are selected from the drop down boxes, which list all the attributes, operators and values respectively. After selecting condition, “Add Condition” adds condition. It is possible to add more than one condition and delete condition using “Delete Selected” button if users think that the added condition is not suitable. “Satisfy Condition” button helps to look at whether the rule is satisfied by the selected case or not. If rule is satisfied, the “Validate New Rule” becomes active and this helps to validate the rule on the un-classified and wrong classified cases of the dataset (Fig. 6).

In Fig. 6, Report Match shows the manual matching results and Algorithmic Match shows the results calculated from rules. The “Save Rule” button helps to save rule in the rule database (KB) and case in the case database. “Edit Classification” button helps to refine the wrong classified cases by adding new conditions until all wrong cases are removed or creating another new rule using knowledge Acquisition GUI. Classification for the censor rule is always “NULL”. The refined cases and the deleted wrong classified cases from the satisfied cases list are shown in Fig. 7.

In Fig. 7, the “Save Rule” button saves the censor rule in the rule database (KB) as censor node and the deleted cases in the case database as NULL classification. If there are more wrong classified cases, the rule can be refined by adding other censor rules. Then to classify the “NULL” classified cases, the alternative rules are created by “Add Classification” button of Fig. 4, are used. For adding classification, the Knowledge Acquisition GUI is like Fig. 5. In this case, first the classification of the rule is selected. This can be done using the drop down box at
the top, which lists TRUE or FALSE classifications for this domain. Having selected the classification, the conditions for creating rule are added. Then it is checked whether the rule is satisfied by the current case or not. If the rule is satisfied, then it is validated to determine whether the conclusion provided by the rule is matched with the reported match. The alternative rule is saved in the KB as parent rule. If any case is wrongly classified by the current rule, then the classification is edited.

4.3 Evaluation Metrics
As this task is a classification task, we use the following conventional metrics: precision = \( \frac{TP}{TP+FP} \), recall = \( \frac{TP}{TP+FN} \) and F-measure = \( \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}} \) where \( TP \) is True Positive (hit), \( FP \) is False Positive (false alarm, Type I error) and \( FN \) is False Negative (miss, Type II error). For a specific threshold value, we calculate TP, FP and FN by comparing manually defined matches (R) with the predicted matches (P) returned by the matching algorithms according to Jimenez et al. (2009).

5 Evaluation Results
Performance of the Hybrid-RDR method and dynamic decision tree depend on the features of the datasets which are created using string similarity metrics and text processing techniques. The performance of Hybrid-RDR method also depends on the efficient rule creation. We compute performance in terms of precision, recall and F-measure. Precision estimates the reliability of the match predictions and recall specifies the share of real matches. During schema mapping, manually matching schemas of two heterogeneous data sources and false identified matches by algorithms are handled by humans. The burden of deleting false identified matches is much easier than creating manual matches among thousands of schemas (Stoilos et al., 2005). As for calculating recall value, manually identified matches are necessary, so recall value is very important. Only precision or recall cannot estimate the performance of match algorithms (Cheng et al., 2005). So it is necessary to calculate the overall performance or F-measure of rule based system and machine learning techniques using both precision and recall. For this, we determine the best performing classification system based on the optimized F-measure (Marie and Gal, 2008) for almost all experimental datasets. For all experiments using decision tree, we use WEKA (Hall et al., 2009) data mining and machine learning toolbox.

5.1 Schema Mapping Results
In the experiment, we randomly select datasets for training and testing. We do three experiments to get the performance of dynamic decision tree and Hybrid-RDR method. The performances (precision, recall and F-measure) of schema mapping using dynamic decision tree and Hybrid-RDR, and the rules used by Hybrid-RDR method are described in Fig. 8.

Fig. 8. Schema mapping results using dynamic decision tree and Hybrid-RDR

In Fig. 8, for all experiments, ML means the results that are produced by dynamic decision tree and ML+RDR means the results that are got by using Hybrid-RDR. In all experiments, we randomly select one dataset for training and other dataset for testing. In dynamic decision tree method, we create decision tree model, \( ML_0 \) for one dataset and use \( ML_0 \) for testing the test dataset. Then we select another dataset and add the previous dataset for which \( ML_0 \) has been created, with the current selected dataset, and create \( ML_1 \) and use \( ML_1 \) for testing the test dataset. In this way, we create ML for all the datasets except test dataset and use ML for testing the test dataset.
In Hybrid-RDR approach, we create decision tree model, \( ML_0 \) for one dataset and use \( ML_0 \) for testing the test dataset. We also select another dataset and use \( ML_0 \) for testing and find some un-classified cases and wrong classified cases. Then we refine the decision tree rule by adding censor rule, \( Rule_0 \) and again classify the cases by adding alternative rules, \( Rule_0 \). Total \( Rule_0 \) is 12, 12, 14 for experiment1, experiment2 and experiment3 respectively. The \( ML_0 + Rule_0 \) is then used for testing the test dataset and also for testing another dataset. We add rules again for the wrong classified cases of another dataset, and total \( Rule_0 \) is 7, 10 and 4 for experiment1, experiment2 and experiment3 respectively. The \( ML_0 + Rule_0 + Rule_1 + ... + Rule_6 \) is used for testing the test dataset and also for testing another dataset. In such a way, we add rules incrementally for all nine datasets, \( ML_0 + Rule_0 + Rule_1 + ... + Rule_6 \) and use for testing the test dataset. In the table, we also see that the number of rules addition for wrong classifications decreases gradually.

The results indicate that using ML+RDR, the performance is higher than ML in almost all experiments in terms of precision, recall and F-measure. In experiment2, though the performance of ML is higher according to precision for almost all datasets except D1, D8 and D9, but recall and F-measure using ML are not higher than ML+RDR. The reason of high precision means less false positive values, and low recall means that the false negative numbers are high (Marie and Gal, 2008).

### 5.2 Prune Tree and Knowledge Base

As an example of prune tree for training one dataset and two datasets using J48 is given in Fig. 9(a) and 9(b) respectively. It is found that the prune tree for training one dataset is different from the prune tree of training two datasets.

![Fig. 9 (a). J48 Prune Tree for training one dataset](image)

![Fig. 9 (b). J48 Prune Tree for training two datasets](image)

Example of Knowledge Base (KB) of Hybrid-RDR which is created for solving un-classifications and wrong classifications is given in Table 1.

<table>
<thead>
<tr>
<th>CaseID</th>
<th>Condition</th>
<th>Conclusion</th>
<th>CaseID</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>Decision Tree</td>
<td>TRUE/FALSE</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>JaroW_ST==0.9</td>
<td>NULL</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>JaroW_ST==1.0</td>
<td>TRUE</td>
</tr>
<tr>
<td>5</td>
<td>2</td>
<td>JaroW_ST==1.0</td>
<td>TRUE</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>JaroW_ST==1.0</td>
<td>TRUE</td>
</tr>
<tr>
<td>7</td>
<td>2</td>
<td>JaroW_ST==0.9</td>
<td>NULL</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>JaroW_ST==0.9</td>
<td>NULL</td>
</tr>
<tr>
<td>9</td>
<td>6</td>
<td>JaroW_ST==0.9 &amp; JaroW_ST==1.0</td>
<td>NULL</td>
</tr>
<tr>
<td>10</td>
<td>1</td>
<td>JaroW_ST==0.9 &amp; JaroW_ST==1.0</td>
<td>FALSE</td>
</tr>
<tr>
<td>11</td>
<td>2</td>
<td>JaroW_ST==1.0</td>
<td>NULL</td>
</tr>
<tr>
<td>12</td>
<td>1</td>
<td>JaroW_ST==1.0</td>
<td>TRUE</td>
</tr>
</tbody>
</table>

**Table 1.** Knowledge Base (KB) for creating rules using Hybrid-RDR.

In Table 1, the attributes \( RID, PID, Condition, Conclusion \) and \( CaseID \) means rule id, parent rule id, condition for the rules, conclusion produced by rules and the classified case id respectively. In addition, \( Lev, S, T, AbbTokT, TokSynT, \) and \( JaroW \) means Levenshtein function, source schema, target schema, abbreviation and tokenization of target, tokenization and synonym of target, and JaroWinkler function respectively. The values 1.0, 0.9, 0.2, 0.6 are thresholds. Example of rule, \( JaroW\_ST==0.9 \) means if the value of JaroWinkler function applied on source and target equals to the threshold value 0.9, then the conclusion is TRUE.

In the table, rule 1 is always true. We use rules 2 to 12 for classifying cases of datasets. We apply rule 2 to classify one dataset and test dataset. In order to solve un-classification and wrong classification of one dataset, we create rules 3 to 8 and apply rules 1 to 8 for classifying another dataset and test dataset. Then we add rules 9 to 12 for solving wrong classification of another dataset which incrementally build the knowledge base. In Table 1, we see that the same rules, for example rule 3 and 4 are used for making NULL and TRUE conclusion. The reason is that first we create censor rule, for example rule 3 for making wrong classification as NULL classification. Then we add...
alternative rule, for example rule 4 for making the right classification. In Fig. 9(a), 9(b) and Table 1, we find that though the rules of training one dataset and two datasets are different using the dynamic decision tree, the rules of classifying one dataset and another dataset are not different using Hybrid-RDR, rather we add rules incrementally for solving wrong classifications. Therefore, the advantage of Hybrid-RDR compared to dynamic decision tree is that we do not need to create training model whenever the schema data changes over time.

6 Discussion
In order to solve the problem of a machine learning algorithm that needs training data set for building models, and a knowledge engineering approach that requires time-consuming knowledge acquisition when schema data changes over time, we propose a Hybrid-RDR. The advantage of Hybrid-RDR is that classification model is built by decision tree only for a small amount of schema data, and knowledge base is then built incrementally by adding rules to solve schema matching problems: unclassifications and wrong classifications. In this research, we only focus on element level schema matching using Hybrid-RDR in order to determine whether our proposed approach can be applied to the schema mapping problem. In future, we will perform structure level matching with our element level matching by Hybrid-RDR to improve the performance.

7 Conclusion and Future Works
In this research, we have proposed Hybrid-RDR approach by combining decision tree, J48 and CPR based RDR. We have computed attributes from the input schemas as well as from the application of text processing techniques and string similarity metrics on the schema names. In addition, we have designed a schema mapping tool and used the attributes in order to create rules using Hybrid-RDR. It can handle two problems of schema matching, unclassifications and wrong classifications using incremental knowledge acquisition techniques. We have also used the attributes to feed into a machine learning technique, dynamic decision tree and have compared the performance of Hybrid-RDR and dynamic decision tree for schema mapping. We have found that our Hybrid-RDR method shows slightly better performance than the dynamic decision tree. The main advantage of Hybrid-RDR compared to dynamic decision tree is that it is not necessary to create models whenever the schema data changes over time. The model which is created for one dataset, can be used for classifying another dataset, and rules can be added incrementally for solving wrong classifications. Later the same model and the added rules can be used for classifying another dataset. In this research, we have only considered element level matching, but accurate results of this element level matching should be a premise to work in the next step with structure level matching.

Acknowledgement
The Intelligent Sensing and Systems Laboratory and the Tasmanian node of the Australian Centre for Broadband Innovation are assisted by a grant from the Tasmanian Government which is administered by the Tasmanian Department of Economic Development, Tourism and the Arts.

References


A Novel Method for Decentralised Peer-to-Peer Software License Validation Using Cryptocurrency Blockchain Technology

Jeff Herbert
School of Computer and Mathematical Sciences
Auckland University of Technology
Auckland, New Zealand
Jeff.herbert@gmail.com

Alan Litchfield
Service and Cloud Computing Research Lab
Auckland University of Technology
Auckland, New Zealand
Alan.litchfield@aut.ac.nz

Abstract

Protecting software copyright has been an issue since the late 1970’s, and software license validation has been a primary method employed in an attempt to minimise software piracy and protect software copyright. This paper presents a novel method for decentralised peer-to-peer software license validation using cryptocurrency blockchain technology to ameliorate software piracy, and to provide a mechanism for all software developers to protect their copyrighted works.

Keywords: Cryptocurrency, blockchain, software, license, validation

1 Introduction

Methods to maintain control of copyrighted software have fallen into three main categories: software activation using a paper based key code, software license validation through an online registration (Peyravian, Roginsky, & Zunic, 2003) and hardware devices (Morgan & Ruskell, 1987). Smaller vendors most often implement software validation in the form of an activation key, whilst global vendors such as Microsoft and Adobe use proprietary centralised software license validation services using the Internet as the primary medium.

Software license validation is growing in complexity due to a combination of technological and economic developments. Commercial models for software sales and distribution have become more complex, with multiple parties existing in the supply chain including software owners, multiple levels of distributors and customers (Sachan, Emmanuel, & Kankanhalli, 2009). Similarly, software is becoming more complex as the scope of use increases (Liu & Roychoudhury, 2012).

This paper proposes the utilisation of a cryptocurrency blockchain similar to Bitcoin, to provide a method for decentralised, peer-to-peer, publicly auditable software license validation that could be used by anyone from an independent software writer to a large software vendor. We provide an overview of cryptocurrency blockchain functions and discuss the benefit of a decentralised peer-to-peer architecture. We then proceed to outline a construct of a transaction message and processes for blockchain-based software license validation, and explore future possibilities and issues.

2 Software license validation

2.1 Software piracy

The Business Software Alliance (BSA) defines software piracy as the unauthorised copying or distribution of copyright software, including downloading, sharing, selling, or installing multiple copies of licensed software. The Internet has provided a convenient medium for software piracy, enabling participants to easily download copyright software, and globalising software piracy by operating in difficult legal jurisdictions. The BSA estimates that in 2013, 43% of software on home computers around the world was not properly licensed, with a commercial value of US$62.7 billion, and even subscription based models such as cloud computing are not expected to provide a significant impact on reducing software piracy with 52% of online credentials being shared (Business Software Alliance, 2014).

Methods to protect software creators’ copyright have been in place since the early 1980’s with a variety of methods proposed and implemented. Suhler, Bagherzadeh, Malek, and Iscoe (1986) suggested that to be successful, software authorisation (validation) needed to be inexpensive, compatible with other systems, and easy to implement. Similarly, Morgan and Ruskell (1987) found various practical measures to deter or prevent unauthorised copying, however the feasibility of these measures depend on various factors such as cost of the measure versus value of the software. Three primary methods for software license authorisation are considered: copy protection, software validation using a distributed paper-based key and hardware-based keys. In these nascent stages of computing, the more effective methods of encryption and validation were limited due to the relatively high cost of hardware devices, limited computing power for encryption methods and no form of easy distribution medium for software license validation. Software copy protection was primarily restricted to
alteration of disk sectors to prevent copying, which was easily defeated with software tools and license keys that were distributed with the software media and easily duplicated. These did little to resolve the issue of software piracy.

With the advent of the Internet, new methods became possible. Peyravian et al. (2003) proposed a new client-server software license validation method using the Internet with a central database for software license validation and detection of hardware platform characteristics that the software is installed on. Using this method, vendors need to manage end user information and need an online validation process for activation of the software after installation. Larger software corporations such as Microsoft and Adobe have adopted the principles of this method. However, online validation requires a significant overhead in management of customers, maintaining security of personal information and yet the validation method is defeatable through easy means, such as redirecting DNS to fake authentication servers, code modification to remove software license validation subroutines, or could be still circumvented through duplication of keys as many license models and license keys support installation on multiple devices.

2.2 Methods to comply with software licensing

Software license validation is growing in complexity due to a combination of technological and economic developments. Commercial models for software sales and distribution have become more complex, with multiple parties existing in the supply chain including software owners, multiple levels of distributors and customers (Sachan et al., 2009). Similarly, complying with software licensing is becoming more complex as the scope of software use increases, such as feature specific enablement keys for software packages, geographical diversity of where software is employed, size of customer organisations, shifts to software as a service models and an increasing use of embedded systems that leads to Internet of Things (Liu & Roychoudhury, 2012).

The need for Software Asset Management (SAM) has developed as organisations and users attempt to comply with complex software licensing requirements. Organisations can choose to manage their software licenses through established SAM processes and standards such as ISO/IEC19770, which provides guidance for organisations to manage software, including assessment of conformity, software identification, and software entitlements (ISO/IEC, 2012). The BSA has established Verafirm to assist organisations with the management of their software licensing, that provides SAM tools and solutions for SME’s and enterprises.

3 Requirements for a software license validation method

As Suhler, Bagherzadeh, Malek, and Iscoe (1986) and Morgan and Ruskell (1987) stress, a successful software licence validation method needs to be inexpensive, compatible with other systems, easy to implement, and relevant to the value of the software. In addition, to be effective against software piracy, a successful software license validation method require several premises to be met:

1) The license mechanism needs to be hard to copy
2) Rights to software licenses need to be easily validated
3) Software licenses cannot be repeatedly generated
4) Validation needs to protect from Man-in-the-Middle attacks

Therefore we need a mechanism that can generate unique values that can’t be regenerated but can be easily verified against the source engine at any time. Cryptocurrencies such as Bitcoin already provide the essential building blocks we need for software license validation. Bitcoins are represented as cryptographically validated digital signatures and as such, unfeasible to copy, whilst the decentralised transaction feature prevents double spending of the bitcoin, ensuring a bitcoin digital signature cannot be repeatedly generated and used. Finally bitcoin transactions are cryptographically secure using public key cryptography to prevent Man-in-the-Middle type attacks. Hence, to meet the premises listed for software license validation, we propose a cryptocurrency blockchain to create a novel method for software license validation mechanism. The following section introduces the cryptocurrency blockchain and applies the blockchain concepts to software license validation.

4 Cryptocurrencies and the blockchain

Cryptocurrencies are a new form of virtual currency, first introduced with creation of Bitcoin, developed by Satoshi Nakamoto (2008). A cryptocurrency is a purely decentralised peer-to-peer electronic cash system, and is the first technology to successfully overcome the requirement for a centralised party to validate transactions. The cryptocurrency architecture provides several blended features including cryptographic validation for all transactions, decentralised money, mint and transactions, all stored on public ledgers within a quasi-anonymous framework (Brikman, 2014). Cryptocurrencies use public-key cryptography to validate transactions between all participants, and digital signatures to ensure transactional integrity and non-repudiation (Peteanu, 2014). The cryptographic mechanisms used by cryptocurrencies provide strong confidentiality, data integrity and non-repudiation services (NIST, 2001) and are in use by business, government and military organisations globally. In a cryptocurrency ecosystem, the public key can be considered as the participant’s account number whilst the private key represents the participant’s ownership credentials. All participants have digital wallets that are used to store the private keys, as well as the digital signatures that represent the cryptocurrency entitlements (coins) that the participants own. Wallets can be stored privately, or online on websites or exchanges depending on the requirements of the participant.

Cryptocurrencies as a currency and monetary system have yet to prove their robustness in both a technological and economic context, needing to be resilient to threats
and attacks as well as being a stable and liquid currency. However, the underlying feature of interest in respect to the cryptocurrency architecture is the blockchain, which is becoming the focal point of development of new cryptocurrency based applications as developers seek to use cryptocurrencies in more practical applications.

4.1 Transactions

Transactions are defined as a message between participants, and consists of 3 segments:

1) Signature: the originator’s digital signature signed with the originator’s private key so that other Bitcoin nodes can verify the message really came from the originating participant.  
2) Inputs: this is a list of the signatures of transactions already in the ledger where the originator was the recipient of bitcoins. These are the funds the originator is using in the transaction.  
3) Outputs: this is a list of how the funds in the inputs should be distributed. All the funds in the inputs must be redistributed in the outputs, so the originator will pay the recipient the required amount and return the remainder as change.

A transaction must have exactly the same number of bitcoins in the inputs and outputs. Hence if user U1 has 10 bitcoins, and wants to send 2 bitcoins to user U2, the transaction will result in U1 receiving 8 bitcoins, and U2 receiving 2 bitcoins. This can be shown as follows:

\[
\begin{align*}
&U1.\text{input}(U1, 10) \\
&U1.\text{send}(U2, 2) \\
&U1.\text{send}(U1, 8)
\end{align*}
\]

The recipient is identified through their public key, so cryptocurrency transactions can be traced throughout the blockchain, to the beginning of the creation of the cryptocurrency. This forms the mechanism for checking the ownership of cryptocurrency bitcoins. Publicly verifiable transactions by any node avoids double spending and provides a high degree of certainty to the participants of the cryptocurrency ecosystem.

4.2 The blockchain

The blockchain consists of a series of blocks where each block contains:

1) transactions or messages sent between users;  
2) a unique digest created when the new block is discovered, called “Proof-of-work”;  
3) the previous reference to the digest of the previous block.

Figure 1 illustrates how each block has a proof-of-work of the previous block, forming the blockchain. Unverified transactions are placed in an unverified transaction bucket, and will be inserted into the next block once it is created.

4.3 Cryptocurrency and blockchain economics

The concept of a cryptocurrency is to overcome the necessity of a centralised “trusted authority” (Nakamoto, 2008) and thus remove or significantly reduce transaction fees associated with transactions such as those incurred with commercial banking transactions.

A cryptocurrency, as a peer-to-peer decentralised technology, relies on a network of low cost computers running software that performs the primary functions of the cryptocurrency. The computers running this software are known as miners, who create bitcoins, validate bitcoin transactions and maintain the integrity of the blockchain public ledger. Miners are rewarded for their investment in running the bitcoin software through creation of bitcoins, and receiving a small transaction fee for their part in validating bitcoin transactions. A cryptocurrency ecosystem requires a significant number of miners to manage the integrity of the blockchain and prevent double spending of bitcoins. However, depending on the implementation of the cryptocurrency ecosystem, miners may not receive transaction fees. For example, Ripple and Gridcoin cryptocurrency participants run the transaction validation software on a voluntary non-profit basis, offering their existing computer and storage resources to run the mining software.

Most cryptocurrency ecosystems have a fixed number of bitcoins that can be created, creating a deflationary economic model due to the finite number of bitcoins as bitcoin value inherently rises due to the limited supply of bitcoins. Bitcoins can also only be created at a certain rate, determined mathematically by the cryptocurrency ecosystem to prevent an oversupply of bitcoins. However, some cryptocurrencies such as Peercoin are established on an inflationary economic model, with an unlimited supply of bitcoins.

These approaches lead to cost effective cryptocurrency and blockchain ecosystems through lower transaction fees (Hochstein, 2014) for the cryptocurrency as a financial instrument. Furthermore cryptocurrencies are found as being considerably lower cost than fiat currencies when comparing economic, environmental and socioeconomic costs (McCook, 2014).

In the next section the characteristics of the blockchain that will help provide a decentralised software validation method are described.
5  Decentralised software license validation

For the purposes of discussing decentralised software license validation, the term *bitcoin* is used generically as a descriptor for a virtual coin from an existing cryptocurrency. Bitcoins are actually digital signatures that are created and stored in user wallets, and have a full publicly verifiable transaction history through the blockchain transaction history. The characteristics of the blockchain can be utilised to provide a record of all software licenses owned by an end user. Through the decentralised peer-to-peer blockchain architecture, any software developer or vendor can allocate licenses to users easily and cost effectively. The principle of decentralised software license validation is to use bitcoins held by the owner to represent entitlement to software.

Two primary methods to utilise a blockchain for software license validation are the “Master Bitcoin Model” and the “Bespoke Model”, discussed in the following sections.

5.1  Master Bitcoin Model

The Master Bitcoin Model is a basic form of software license validation proposed by Fortin (2011) and implemented by Lebo (2014) in a proof of concept project called “dissent”, using Namecoin as the underlying blockchain. In this model, the vendor address/bitcoin combination represents license ownership, and if the user has a transaction showing the bitcoin originated from a specific vendor address, the user is considered to have ownership of the software. This concept is demonstrated in the following example.

UserAddress1 (U1): the end user address for the wallet that holds the bitcoin indicating software entitlement

1) V1 creates the M1 “MasterAddress1” on the blockchain, representing a particular Software application.
2) V1 then adds some bitcoins to M1, loading it with some bitcoins that when transferred will represent entitlement to the Software application.
3) The end user purchases the Software application through a non-cryptocurrency transaction.
4) V1 transfers a Master bitcoin from the M1 address to the U1 address. The transaction itself confers the ownership of the bitcoin, and the end user now has the bitcoin from M1, the Master bitcoin, in the user’s associated wallet. Hence, the user’s ownership of a bitcoin from M1 confers entitlement to the Software application, and is a transaction publically verifiable on the blockchain.
5) The Software application then validates that U1 has received a transaction from M1, and is the last transaction in the chain of transactions.

The sequence of transactions can be shown as follows:

V1.create(M1)
V1.send(M1, 100)  ‘V1 adds 100 bitcoins to M1
Software purchase
M1.send(U1, 1)
S1.validate(U1)

Ownership of the Master bitcoin can be transferred as shown in Figure 2, so that the software vendor can be guaranteed only a single user is using the software though checking the blockchain “chain of title” for the Master bitcoin originating address. Hence, U1 can now transfer ownership to a new party, U2. Again, the transaction itself confers the ownership, and any entity can verify the chain of transactions from U2, to U1, and back to M1 to

![Figure 2: Master Bitcoin Model transfer of ownership sequence example](image-url)
confirm that current ownership is held by U2, who will have the last transaction in the chain of transactions.

M1.send(U1, 1), U1.send(U2, 1)

Since bitcoins do not have serial numbers, once a non-Master bitcoin is combined with a Master bitcoin, the originator of each specific bitcoin cannot be identified because in reality, they are simply digital signatures that have been combined to form a new digital signature. However, in the Master Bitcoin Model, the value of the Master bitcoin is not important, only the fact that there is a transaction history from the originating Master bitcoin. Fortin (2011) proposes that if a Master bitcoin is combined with a non-Master bitcoin, the biggest recipient is the one that holds the Master Bitcoin, or whoever has the lowest address (alphabetically) has precedence for ownership of the Master bitcoin. This property establishes non-divisible ownership of the Master bitcoin allowing ownership to be transferred.

In summary, using a unique blockchain address to represent a particular software application, the Master Bitcoin Model can be used to provide non-repudiable proof of ownership of a bitcoin that originated from a specific address, thereby conferring the entitlement of the software license to the user. However, the software application will need to have the capability to read the blockchain to establish the chain of title to the user.

5.2 Bespoke Model

As mentioned earlier, most cryptocurrencies are designed with a currency in mind and so they create virtual coins represented as digital signature that are stored in users’ wallets and have a full publically verifiable transaction history that is stored on the blockchain. For the purposes of discussing the Bespoke Model, we define a Token as a digital signature that represents entitlement to a specific software application, rather than a bitcoin, because the license validation model is not using digital signatures to represent a virtual currency. A user address that holds a particular Token from a specific vendor address is entitled to the software license, and therefore is entitled to use the software. Hence, the vendor/token combination represents the entitlement for use of the software.

Blockchain specifications vary from cryptocurrency to cryptocurrency, and as such, cryptocurrencies can be architectured with unique characteristics to meet purpose specific applications. The Bespoke Model uses a custom blockchain transaction specification that includes additional fields tailored to the requirements of a flexible software license validation schema. This would provide the scope needed for the wide range of users and license models in the modern technology environment. We can also provide several useful mechanisms using the blockchain as the basis for license validation, license upgrade, transfer of ownership and even software integrity checking. A customised blockchain specification, as shown in Figure 2, could include new blockchain fields to improve software license validation and prevent software piracy through software integrity checks and protecting the software from reverse engineering and executable code modification.

These fields are all stored on the blockchain as data, encrypted using the in-built cryptocurrency public/private key mechanisms. In principle, the software vendor utilises the user’s public key to encrypt the data being placed into the fields, with the user’s private key required to decrypt the fields. The user can confirm the transaction integrity signature with the vendor’s public key.

The custom fields outlined in the proposed specification are described as follows.

The Token is used for standard license validation mechanisms where the ownership of the Token demonstrates entitlement. The Token can be used for software license validation operations such as for software upgrades, or to provide a unique attribute to the transaction, such as “first 100 purchasers” that may have collectible value in the future.

The License Key provides advantages over the Master Bitcoin Model because many software applications have specific features within the application that are activated on a per feature basis. Having the License Key securely held on the blockchain means software vendors can easily enable “feature activation”, and have flexibility with software application licensing models, where users could rent software use for a small periods of time, rather than purchasing or renting use on a month basis.

Similarly, the vendor can place a software hash of the software on the blockchain. A bootstrap loader or the software itself can read the hash and check the software version. This hash can be updated with every new patch, plus minor or major releases of the software. This could protect software from malware infection or some forms of reverse engineering.

Additional protection could be provided through a bootstrap loader, which is a portion of executable code that is used to pre-execute the software application or to be used as an integral part of application execution. The purpose of this is to further prevent reverse engineering of the software application. At some stage the unencrypted bootstrap code will be executed and stored in memory, and thus susceptible to interception by reverse engineering. This bootstrap code can change with every patch, and minor and major release, making reverse engineering a constant effort.

The signature field is a possible additional field that can be used by the vendor to sign the entire transaction contents using the private/public key pair of the software MasterAddress.

Existing software validation uses digital signatures to verify downloadable software and digital certificates to
prevent Man-in-the-Middle attacks during the download process. However the proposed custom specification provides validation of installed software on the user’s device on an ongoing basis, providing risk mitigation against malware code injection attacks.

Exploring these concepts further, we look at the Token feature. As already mentioned, the Token is used to confer ownership, however in comparison to the Master Bitcoin Model, the Bespoke Model presents significantly more opportunities to use the Token for software license validation purposes. In addition to validating that the user owns the Token, it can be used for in mechanisms to upgrade software code versions or transfer of ownership. In the first instance, license validation by reading the Token at any time interval, say every 600 seconds, user login/logoff, or software start-up. These examples read an existing transaction on the blockchain, but don’t create a transaction. For updating information on the blockchain the mechanism is to create a transaction between the user address and the software address that represents the software application. Each transaction that occurs creates a new user address with its own unique public/private key, and a new transaction with data encrypted by the user’s new public key. All addresses are unique, with their own public/private key pair. For software license validation purposes, a transaction process from a device with S1 software installed could be like:

1) S1 reads the blockchain transaction for U1
2) S1 decrypts the token from blockchain data for U1
3) S1 checks the Token originates from M1
4) S1 continues to execute on the end user’s device

Shown as:

S1.read(UserAddress1,transaction)
S1.decrypt(UserAddress1,Token)
S1.validate(Token)
S1.execute

To upgrade Software application versions such as with a patch update, the software application can periodically request an update from the vendor. The use of a new U2 address for the upgraded software application is so that entitlement to earlier software versions is maintained through U1 in case of downgrade requirements. All Ux addresses are stored in the user’s digital wallet, and as such all entitlements are associated with the user. The vendor could also transparently release new license keys with minor releases such as patch updates further reducing any risk of license keys duplication. Software upgrades could be achieved by:

1) S1 sends a request to M1 with new U2 address
2) M1 checks the token came from U1 and is valid
3) M1 creates a new transaction with update data
4) S1 reads data to check if it needs an upgrade
5) S1 auto-upgrades

Shown as:

S1.send(MasterAddress1,UserAddress2,Token)
M1.validate(UserAddress1, Token)
M1.send(UserAddress2, Token)
S1.read(UserAddress2, Token, License, Hash)
S1.upgrade

Although this is similar to existing software version checking mechanisms online such as Microsoft Update, this process allows the software vendor to re-cut a license key or hash code for the software upgrade, and have the software automatically validated. Updating the previous transaction process to include these fields as shown in Figure 4. Both the previous software version and the upgraded software version are available for use.

1) S1 send an update request to M1 with new U2 address

![Figure 4: Bespoke blockchain software upgrade sequence example](image-url)
2) M1 checks token came from U1 and is valid
3) MasterAddress2 M2 created for the new transaction
4) M2 cuts new License Key for new software version
5) M2 creates Hash for new software version
6) M2 creates new bootstrap for new software version
7) M2 encrypts the new License Key, Hash and Bootstrap using PublicKey(U2)
8) M2 signs the transaction with PrivateKey(M2)
9) M2 creates new transaction with the new data
10) S1 reads new transaction data for upgrade
11) S1 downloads software and auto-upgrades
12) S1 run itself

Shown as:

\[
\begin{align*}
S1 & .send(MasterAddress1, UserAddress2, Token) \\
M1 & .validate(UserAddress1, Token) \\
M1 & .createaddress(M2) \\
M2 & .License(License.new) \\
M2 & .hash(S1.new) \\
M2 & .bootstrap(Bootstrap.new) \\
M2 & .encrypt(M2.License) \\
M2 & .encrypt(M2.hash) \\
M2 & .encrypt(Bootstrap.new) \\
M2 & .sign(Transaction.new) \\
M2 & .send(UserAddress2, Token) \\
S1 & .read(UserAddress2, Token, License, Hash) \\
S1 & .upgrade \\
S1 & .execute
\end{align*}
\]

Hence we have shown that license validation can be easily achieved using the blockchain, and through the same mechanism, additional integrity and security protections can be added. Furthermore, blockchain scripts allow for intelligent programming of actions within a transaction. This provides a new level of dynamism as a transaction may take different actions based on the inputs, outputs, field contents and originating and destination addresses. New blockchain protocols are being developed that include full Turing completeness capability, allowing anyone to write smart contracts and decentralised applications with their own arbitrary rules for ownership, transaction formats and state transition functions (Buterin, 2014).

5.3 Issues that are overcome

We can see that the Bespoke Model overcomes the problems originally highlighted earlier in this paper, and significantly improves on the Master Bitcoin Model. However it does require a separate cryptocurrency ecosystem to be developed and maintained, whilst the Master Bitcoin Model can utilise and run in an existing cryptocurrency ecosystem. While the Master Coin Method meets the requirements outlined for a successful software license validation method and it has been demonstrated to be workable in proof-of-concept, it has limitations that may detract from its usefulness. The Bespoke Model overcomes these as follows:

1) Each software license is hard to copy because the license is represented by a transaction between vendor and user, is cryptographically verifiable, and stored in the user’s blockchain wallet. An adversary would require the password to the user’s wallet in order to access the user’s private key. Already, multifactor authentication mechanisms are available to further enhance user wallet security. In addition, the license key does not even have to be disclosed to the user, so it cannot be copied. Every transaction is cryptographically secure and cannot be modified.

2) Software licenses are easily validated through the blockchain “chain of title” and the data being held within the blockchain itself. Furthermore, having on-chain license keys allows the vendor to distribute keys for specific feature activations, and allows keys to be re-cut quickly and efficiently without any intermediate parties involved.

3) Software licenses cannot be regenerated because the software application is taking the license key directly from the blockchain, requiring the user’s private key. Even if the key generator at the vendor is compromised, there is no way to get the license key onto the blockchain without the vendor’s private key to sign the transaction.

4) There is no Man-in-the-Middle attack possible using the blockchain. An adversary cannot intercept any data in the blockchain without the user’s private key, and cannot redirect DNS or IP traffic to an adversary’s custom server to provide software validation.

There are additional benefits beyond the software license validation method, with clear scope for software vendors to provide integrity and protection for their software applications. Furthermore, the blockchain peer-to-peer architecture means that there is no single central point of failure for software license validation. Licensing validators can be run anywhere around the world, and could be run on a not-for-profit basis or on some other commercial model as appropriate. Vendors would run vendor-specific software to manage a license creation process and interaction with the blockchain but will not need to maintain their own dedicated license validation infrastructure with its associated overheads.

The proposed software license validation model provides an opportunity for small developers through to large software vendors to preserve their copyright in their software, and prevent software piracy whilst having a flexible mechanism to license their software.

5.4 Potential Issues

In order for the “Bespoke Model” to work, users will need to provide some form of authentication to their user wallet in order to access the private keys so that the software application can complete its validation function. This is a similar process to users needing to access their blockchain wallet to conduct any transaction in any cryptocurrency, so the action is commonplace. However, if we are to achieve true user mobility where a user can login to any installed application and be validated for use, the wallet will require some portability. In addition, for an automated authentication process to work, the software application will need to access the private keys for the
user addresses to complete the license validation process. Disclosing private keys is not desirable, so the wallet will need to be an application and have the ability to decrypt data on the blockchain and present that to the software application through an API interface. Alternatively, the user may be authenticated to an OpenID or OAuth authorisation service provider such as Facebook or Google, to prove their identity and allow authorised API requests to the user wallet.

Other issues are the security threat model for loss of data or the compromise of the system if a user loses control of a wallet and user credentials are exposed, or a vendor system is compromised. Multi-signature authorisation already provides potential solutions to these issues, similar to two (or more) parties being required to sign a bank cheque. This may place an additional overhead on the software license validation method, but it could also significantly reduce the risk of compromise or loss-of-ownership issues.

5.5 Further Opportunities

In this paper we deal with licensing on the basis of a single user receiving a single license for a software application and where a single user can have many addresses representing software applications in their wallet and on the blockchain. However, in a multi-user corporate environment there are additional challenges, such as licenses that are not permanently allocated to staff and licenses that have to be transferable within the organisation. For example, a staff member leaving the organisation cannot be allowed to exit with a software license in their personal blockchain wallet.

In the Bespoke Model, each license entitlement requires a unique address belonging to the organisation to be sent a Token, with multiple addresses defining the number of Tokens the organisation has available. Hence an organisation with 100 users would have 100 addresses in a wallet dedicated to the organisation. The licenses need to be allocated to users within the organisation and also be revoked. Furthermore, users need to authenticate using corporate login credentials, ideally using a single sign-on approach to access the license from the organisation’s wallet. This requires some form of authentication service internally for the blockchain software license validation with capability to integrate into a service such as LDAP or Active Directory for single sign on. An organisational level blockchain license validation application is required to implement a successful multi-user software license validation method in a multi-user environment.

Additionally, the license validation blockchain method provides an opportunity to manage licenses on non-human operated devices. As the Internet-of-Things grows and evolves, these connected devices will require mechanisms to auto-update software and validate software in a legitimate manner. For example, the customer who owns 10,000 Internet-connected devices, but only pays software maintenance on 2000 of these devices, will only have license keys to update 2000 devices. This capability is easy to achieve in a peer-to-peer decentralised software license validation ecosystem, and hard to manage using any other type of process.

Perkins (2014) states that the Identity of Things is a growing outcome of the Internet of Things. That is, devices and data have a relationship with someone or something that needs to be identified, and assets and users associated with these need to be managed. The license validation method meets the requirements as license entitlement is essentially defined as a Token/source object, providing the identities of the parties through the blockchain address, and the types of activity between parties through the transactional history.

6 Limitations

There are a number of limitations noted in this paper. The blockchain depends on other participants in the cryptocurrency ecosystem to create and validate transactions. However, we do not explore the blockchain ecosystem or business model because there exist myriad types of stakeholders who may perceive various implications or have vested interests in a blockchain ecosystem outcome. These considerations are outside the scope of this paper.

There are currently no standards for cryptocurrency or blockchain technology available, although if the software license validation mechanism was established, standards such as ISO/IEC19770 could be revised to include software validation blockchain technology.

Presently, there is limited peer reviewed work available for cryptocurrency subject matter, and readings are commonly taken from current industry sources and leaders.

7 Conclusion

Software license validation has been one of the primary mechanisms to prevent software piracy since the mid-1970’s. The methods for software license validation evolved with the Internet to include online license validation in addition to the traditional paper based license keys provided with software. Software pirates and hackers are able to reverse engineer and remove protection mechanisms whilst license keys are copied, duplicated or regenerated to provide a valid license key. We contend that software developers need a license validation method that provides a unique license key that cannot be copied or regenerated, associates the identity of the user with the license key and is cost effective.

We show that a customised cryptocurrency blockchain can be used to provide a decentralised peer-to-peer software license validation method that meets the requirements for software license validation in a cost effective manner through the use of the cryptocurrency theory. The blockchain offers many opportunities to include software integrity and protection mechanisms, providing additional value for software vendors and end users. The blockchain software license validation method can also be automated to provide license validation and identity for Internet of Things devices.

8 References


Visualising Moving Clusters using Cluster Flow Diagrams

Jason Thompson\(^1\)  Torab Torabi\(^2\)

Department of Computer Science and Computer Engineering
La Trobe University,
Melbourne, Victoria 3086
Email: j9thompson@students.latrobe.edu.au\(^1\), t.torabi@latrobe.edu.au\(^2\)

Abstract
Moving clusters represent groups of objects that move together, for instance, groups of people evacuating a building. However, because moving clusters are composed of lists of clusters, they are not directly interpretable by analysts in their raw form. Hence, this paper introduces ‘cluster flow diagrams’, a clear, concise and aggregated visualisation of a collection of these clusters. In particular, cluster flow diagrams give a snapshot of all of the formations of moving clusters, the changes within them and their interrelationships with one another. Additionally, the clusters are characterised by their member’s spatial movements through two composite visualisations. The diagrams are generated and evaluated using a synthetic evacuation scenario, a gaze tracking experiment, and a collection of storm tracks, truck trips and naval vessel trajectories.

Keywords: Moving clusters, spatio-temporal, visualisation, moving points, trajectories, groups, clusters

1 Introduction
A moving object could be a person moving about a building, a boat moving about an ocean or even a person’s point of regard as they view an image. The movements of these objects can, in the aggregate, give insights into the underlying processes that generated them. One way to analyse moving objects in the aggregate is to group them and then to analyse the formation, dispersal and evolution of these groups. But, this is challenging because the underlying data is structured as lists which are not directly interpretable by analysts in their raw form. One way to address this problem is to visually represent the groups. A straightforward solution would be to apply existing literature on visualising time varying groups to spatial objects. A time varying group is a collection of objects, e.g. collections of documents, that change membership over time. Those members may transfer from one group to another, and the groups may in turn form and disperse at any point in time.

Whilst there has been some recent research into the problem of visualising collections of time varying groups, most researchers only propose specialised visualisations for different types of underlying data e.g. topics in a corpus of documents (Cui et al. 2011), groups of people (Reda et al. 2011) and/or tweets (Xu et al. 2013) within a social network, scenes of characters in narratives (Tanahashi & Ma 2012) (Liu et al. 2013) and groups of programmers in a software engineering project (Ogawa & Ma 2010). Conversely, there has been some general research into visualising groups of vertices in dynamic graphs (Sallaberry et al. 2013). This specialisation is necessary because it allows the groups to be characterised in terms of the underlying data. If textual data is used then keywords can be rendered over the groups (Xu et al. 2013), or if a narrative is used then the groups can be annotated with the names of the characters and their location (Liu et al. 2013). This additional information produces a richer visualisation. However, there has not been much work to address the problem of visualising time varying groups of moving objects which is the focus of this paper. Specifically, the contributions of this paper are:

- a novel visualisation of groups of moving objects, ‘cluster flow diagram’;
- a ‘noise cluster’ which describes how organised the dataset is over time;
- an approach to process moving clusters to make them suitable for visualisation; and
- two composite visualisations that characterise groups in terms of their member’s movements.

2 Related Work
The related work provides the frame of reference for the proposed cluster flow diagram. Section 2.1 and 2.2 review work that addresses a similar problem whilst section 2.3 and 2.4 review the foundational work that cluster flow diagrams build upon.

2.1 View of Individuals
Views that depict the individual objects preserve the outliers in the dataset and their unique behaviours. There are two common trade-offs with this approach: the visual clutter in a diagram increases with the number of objects and clusters must be manually identified and/or characterised by an analyst. One approach is to project objects from two dimensional space to one dimensional space at regular time intervals and to form trajectories from the new points (Crnovrsanin et al. 2009) (Shrestha et al. 2013). The resulting visualisation can, in theory, be used to identify some of the movement patterns explained in (Dodge et al. 2008). Another approach is to segment...
an environment into regions of interest and then plot each trajectory as a time series with the region ID on the y-axis and time on the x-axis (Sookhanaphibarn et al. 2011)(Rasche et al. 2014). The overlapping parts of the time series can be interpreted as groups of objects. In contrast, groups of objects can be visualised by colour coding their constituent trajectories (Andrienko & Andrienko 2013). Lastly, the time varying changes in the members of a cluster can be compared to one another by stacking each object’s trajectory in a three dimensional visualisation (Tominski et al. 2012). Though this method shows how the state of a group changes with time, it does not show how different clusters relate to each other over time.

2.2 Aggregate View

An aggregated view of the objects eliminates outliers, noise, and high frequency variations to provide a summarised view of the underlying phenomena. One aggregate approach visualises time varying clusters using stacked bar graphs to represent the relative membership size of clusters over time (Bremm et al. 2011)(von Landesberger et al. 2012). However, this approach cannot scale well when the number of time samples is increased; it becomes cluttered. Alternatively, using a river-like design motif that is structurally similar to cluster flow diagrams, the movements of objects between areas of interest can be visualised(Burch et al. 2013). While the aforementioned research emphasizes the temporal characteristics of the clusters, the relationships of groups with respect to their spatial environment can be examined using flow maps (Andrienko & Andrienko 2011).

2.3 Sankey Diagrams

Cluster flow diagrams can be considered as a specialisation of Sankey diagrams, which are graphs. Characteristically, each edge is associated with a value, and its thickness is proportional to that value. These diagrams also show the splitting and merging of different quantities and have been used extensively to describe energy transformations in a system, for example, energy usage in the United States (https://flowcharts.lincoln.gov/energy.html). These types of diagrams are typically constructed manually. A list of software to construct such diagrams is available at http://www.sankey-diagrams.com/sankey-diagram-software/. Cluster flow diagrams build upon Sankey diagrams by attaching additional meaning to edges (or links), incorporating time and conserving the number of objects with a special purpose noise cluster (described in section 3.1).

2.4 Moving Clusters

A number of algorithms have been developed to identify groups of objects that match a particular spatial-temporal pattern in a given dataset, like a flock pattern (Jeung et al. 2011). However, in the case of cluster flow diagrams, only groups of closely located objects are relevant, for example, moving clusters (Kalnis et al. 2005), convoys (Jeung, Shen & Zhou 2008)(Jeung, Yu, Zhou, Jensen & Shen 2008), dynamic convoys (Aung & Tan 2010), and swarms (Li et al. 2010). Moving clusters have two properties that make them a satisfactory candidate to segment a dataset into groups: an object belongs to only one group at a time which ensures that the number of objects depicted in a diagram is the same as the number objects in the dataset; and the transfer of membership between groups is recoverable which is necessary to show the relationships between clusters. That being said, in this paper, undesirable clusters are eliminated using constraints on duration and member count that are similar to those used in convoys, dynamic convoys and swarms.

3 The Cluster Flow Diagram

In this section, we propose cluster flow diagrams as a visualisation of groups of moving objects. We describe the two visual elements that make up a diagram those being clusters (section 3.1) and links (section 3.2). In a cluster flow diagram, the x-axis represents time and runs from left to right. The y-axis has no meaning so that visual elements can be shifted up and down to produce more intelligible layouts. Cluster flow diagrams contain two types of visual elements, links and clusters, which represent aggregated transfers of membership and clusters respectfully. An example cluster flow diagram is shown in figure 1. It was generated from the VAST dataset described in section 6.2. For illustrative purposes, the clusters and some of the links are labeled $c_x$ and $l_x$, respectively, where $x$ is a numerical identifier for that cluster or link.

Over existing methods our visualisation offers some advantages. It scales in terms of the number of objects (it does not become more cluttered with additional objects but it will with additional groups) unlike other approaches that draw every object, and it scales in terms of the number of samples (it does not become more cluttered with additional time samples). It addresses the issue of characterising the groups in terms of their member’s movements. Lastly, by drawing groups as well defined bars and links with additional geometry, it provides richer and more precise semantics than other approaches that utilise an organic or river-like design motif(Burch et al. 2013)(Cui et al. 2011)(Xu et al. 2013).

3.1 Clusters

Clusters are groups of objects that are located close to one another and are designated by colour coded horizontal bars. The horizontal axis represents time, so the left and right ends of a horizontal bar correspond to the time that the cluster formed and the time that its members dispersed respectively. Furthermore, if a cluster appears to the right of another, it means that the right cluster formed later than the left. In figure 1, $c_0$ formed before $c_1$.

To show the relative memberships of the clusters, the thickness of a cluster’s bar varies in proportion to its membership. For example, in figure 1, when $c_2$ gained more members towards the end of its lifetime, its thickness increased. Also, the thicker bars in a diagram correspond to the clusters with the most members. Note that links introduce additional geometry to the clusters to which they attach, so a cluster’s membership may be smaller or larger than what is depicted at the connection point.

If at any point in time an object is not assigned to a group, it is placed in the noise cluster. The noise cluster gives insight into how organised a system is. Changes in the thickness of the noise cluster can indicate if the system is getting more or less organised as time progresses. In figure 1, for example, the noise cluster noticeably decreases in thickness as time progresses, correlating to the objects assembling at the exit points. The noise cluster is given a dark green colour to differentiate it from the other clusters which
have no assigned meaning to their colour. However, when the movements of objects are visualised, they can be colour coded by their cluster to create a semantic connection between the two visualisations.

As links represent transitions from one cluster to another, they have a particular direction. This direction is, in part, represented by the link’s colour gradient. The colour of the cluster from which the link originates fills the majority of the link while the colour of the cluster at which the link terminates has only a minor influence. The gradient is only needed to disambiguate situations where a link is nearly vertical; otherwise, the direction can be determined by the angle of the link. Because time runs along the x axis and objects cannot travel backwards through time, the objects transition from the left endpoint to the right endpoint, e.g., in figure 1 the members of $l_1$ can only transition from the noise cluster to $c_4$.

### 4 Characterising Groups

The groups as they are represented now are completely non-descriptive; they could be collections of tweets, actors, or anything. A richer representation of the groups and their membership is required. We propose two composite visualisations to characterise the groups in terms of their member’s movements: a dynamic composition and a static composition.

![Figure 1: A sample cluster flow diagram generated from the VAST synthetic evacuation dataset.](image)

#### 3.2 Links

When objects leave one group and join another, a link forms between the two corresponding horizontal bars. For example, $l_2$ shows a number of objects left $c_1$ and joined $c_2$. Links aggregate individual transitions between clusters so that the relative number of transitions between clusters is clear which would not be the case if a large number of transitions were rendered individually on top of one another. Also, aggregating the links makes the diagrams less cluttered in appearance (see figure 7). The thickness of a link is proportional to the number of transitions it represents: thicker links correspond to larger numbers of transitions. As these links represent multiple similar, but not necessarily equal, transitions, the starting location of a link is represented by three values, the minimum, maximum and average starting times of the underlying transitions that make up the link. The ending location of a link is represented similarly. In place of the minimum and maximum values, the first and third quartiles could have been used to reduce the effects of outliers. Figure 2(b) shows how the three values are depicted. The purpose of the geometry is to provide an insight into the period of time during which the objects transitioned and the skew of the starting and ending times.

When a link connects two clusters with the same y-coordinates, it is drawn in a ‘u’ shape below the clusters, e.g., $l_3$ in figure 1. If these links were to be drawn as straight lines starting inside one cluster and ending inside of another, the min, average and max times could not be shown, and they would obscure their start and end clusters and potentially other horizontal links.

![Figure 2: Two configurations that connect a link to a cluster. A link with no underlying variation (a) is positioned at the exit or entry time (i) of the objects. Conversely, a link with variation (b) depicts the minimum (i), average (ii) and maximum (iii) of the exit or entry times.](image)

The dynamic composition consists of an animated visualisation of the spatial domain and the cluster flow diagram. Moving objects are displayed on top of their trajectories in the animated visualisation. The moving objects are shown as dots coloured according to their current group. When the animation is played, the dots trace out the path of their corresponding objects and change colour as they change groups. So that the animation and the cluster flow diagram are synchronised in the temporal domain, a vertical red bar is overlayed on top of the cluster flow diagram at the animations current time. In this configuration, an analyst can playback the dataset and jump to particular points in the dataset. The analyst is able to characterise the groups by viewing them in the spatial domain as well as on the cluster flow diagram. An example is shown in figure 3.

While an animated view may be useful for exploratory analysis of data, a static figure is still more efficient – all of the information can be seen at a glance. Thus, we propose a static composition. It again includes a visualisation of each object’s trajectory and the moving objects colour coded by group; however, the elements are static in this configuration. This approach allows an analyst to annotate the cluster flow diagram with static views of the dataset. In this way, key events in the evolution of the groups can be marked and viewed in the spatial domain. An example is shown in figure 4; note that a red bar is used to indicate to which time that the static visualisation corresponds.
5 Generating Cluster Flow Diagrams

An overview of the method used to generate cluster flow diagrams is provided in figure 5. The four groups of rectangles correspond to the four main stages: identify moving clusters (section 5.1), identify links between clusters (section 5.2), layout the diagram (section 5.3) and render the diagram (section 5.4).

The quality of the end diagram depends on the quality of the clustering procedure and also the randomness of the dataset. Hence, a number of strategies have been included to mitigate these two factors. First, all of the moving clusters that are identified are filtered if they are “short-lived” and/or do not contain enough members. Second, the transitions of objects between clusters are aggregated to form links and filtered if they do not contain enough members. Finally, to improve the aesthetics of a cluster flow diagram, the number of line intersections and non-horizontal lines are minimised by shifting clusters up and down the y-axis.

5.1 Identifying Moving Clusters

Moving clusters are identified in much the same way as described in (Kalnis et al. 2005): the lifetime of a dataset is divided into a number of equal time intervals; the objects are clustered during each time interval using the DBSCAN algorithm (Ester et al. 1996); and clusters that share common members are connected to form the moving clusters. The major departure from the above method is that the noise from each time interval is always placed into a noise cluster. The noise cluster is used to conserve the number of points represented on the diagram at any point in time. Also, the noise cluster cannot be filtered.

This procedure can produce peculiar results. An object’s motion during a time interval is approximated by a point. If the time intervals are too long or an object moves too quickly, the sampled point will not be representative of the object’s motion during that time interval. This reduces the quality of the resultant moving clusters as objects can appear to be absent from a cluster when, upon visual inspection, they should not be and vice versa. Conversely, the movements of an object during a time interval could be approximated by line segments. These could be clustered instead of points, as in TRACLUS (Trajectory clustering: a partition-and-group framework 2007). Note that clustering line segments introduces a number of parameters that do not have straightforward physical interpretations. Consequently, it was decided to approximate motion using points and to reduce the length of the time intervals to produce acceptable results.

In some cases, “short-lived” and/or small (containing few objects) moving clusters can be generated. These low quality clusters result from determining cluster membership at discrete points in time and irregularities in the object’s movements. In any case, clusters that never contain more than \( \text{min count} \) objects at any point in time or that have a duration shorter than \( \text{min duration} \) are removed, and their members are reassigned to the noise cluster so that no objects are left unaccounted. This has the benefit of reducing clutter in the resultant diagram as well as reducing the number of clusters that need to be processed by the layout algorithm.

In some cases, a single object can momentarily join a moving cluster. This can happen if an object momentarily comes within range of a cluster or an object temporarily bridges two clusters. These short artifi-
cial periods of membership inflate the membership of clusters and introduce spurious transitions, so they are removed. An object must be a member of a cluster for a minimum amount of time, \textit{max member time}; otherwise, its membership is ignored and it is placed in the noise cluster. Once these momentary memberships have been removed, the members of each cluster are recounted. If a cluster no longer has any members then it is removed.

Additionally, objects can appear to exit and rejoin a cluster, resulting in a lapse of membership. This happens for a number of reasons: an object on the edge of cluster temporarily moves out of range; a bridging object(s) temporarily moves out of range splitting a cluster; or an object genuinely leaves a cluster, for some reason, and re-joins it later. The first two cases are artifacts of the clustering process while the second is an interesting characteristic of the dataset. In this method, cases like the first two are identified and eliminated by way of a constraint on the duration of lapses in membership: if an object spends less than \textit{max absentee time} outside of a cluster then that object is considered to have never have left the cluster.

Figure 6 shows the cluster membership of eight objects. Each row represents the lifetime of an object, and time runs from left to right. The colour of the row indicates which cluster that object belongs to at that point in time. Figure 6 shows the lifetimes of the objects before removing momentary memberships (a), after removing momentary membership (b) and after removing the artificial lapses in memberships (c). Although the intention is to remove artificial transitions, an inappropriately selected threshold can eliminate useful information. By using diagrams like the ones in figure 6, it should be possible to select an appropriate value.

![Figure 6: The trajectories before processing (a), momentary memberships are surrounded by red rectangles; after removing the momentary memberships (b), artificial lapses in memberships are surrounded by red rectangles; and after removing artificial lapses in memberships (c).](image)

5.2 Identifying Links between Clusters

A transition denotes the change in cluster membership of an object. It consists of a source cluster, a destination cluster, a start time which is the last time that the object is a member of the source cluster, and an end time which is the first time that the object is a member of the destination cluster. There are two cases when a transition is outputted. The first case is when an object transitions from a moving cluster, to the noise cluster and then to another moving cluster. The resultant transition joins the two moving clusters ignoring the time spent in the noise cluster. The rationale for ignoring the transition to and from the noise cluster is that when an object transitions from one cluster to another there may be a period of time where it appears as noise, but it is actually purposefully moving between clusters. Ignoring the time that the object spent as noise produces a more meaningful transition. Of course there is the risk of ignoring time that an object genuinely spent as noise before joining another cluster. This case was not considered in the implementation, but it might be identified by looking at the amount of time that the object spent as noise or maybe the tortuosity of its path. Alternatively, a domain specific strategy may yield more meaningful results. The second case, the obvious one, produces a transition when an object transitions from any moving cluster to another. This case applies only when the first case does not.

At this stage, a lot of transitions can be generated. Not all of the transitions are unique and some are very similar. Consequently, the transitions can be easily aggregated into groups called \textit{links}. Notably, operating on links offers some benefits such as it speeds up the layout stage and reduces the appearance of clutter in the final visualisation in addition to the reasons stated in Section 3.2. Links are formed by grouping all of the transitions that are similar to one another where two transitions are similar iff the following holds true:

- the start times differ by no more than \textit{max time}
- the end times differ by no more than \textit{max time}
- they start at the same cluster
- they end at the same cluster

Note that \textit{max time} is a user set parameter. Larger values of \textit{max time} correspond to less discriminate clusters. A link is the largest collection of transitions that are all similar to one another either directly or indirectly through another transition. Additionally, none of the transitions inside of a link can be similar to the transitions inside of another link.

As an alternative definition, a graph can be formed where vertices correspond to transitions and two vertices are connected if their corresponding transitions are similar. Clusters are the connected components of this graph. The concept of connected components is described in (Skiena 2008). This type of clustering was selected because it has one straightforward parameter, \textit{max time}, and does not require the number of clusters to be known beforehand. Moreover, it produces satisfactory results.
Lastly, links can be filtered if they do not represent many transitions. For example, compare the unfiltered diagram in figure 7 (b) with the filtered version in figure 1. The filtered version does not show links that represent two or fewer transitions.

5.3 Layout Problem

This section will describe the layout problem. The layout algorithm is not the contribution of this paper, so it is not described in detail. In reality any layout algorithm would suffice, e.g., the evolutionary algorithm approach used to optimise storyline visualisations (Tanahashi & Ma 2012).

A simplified form of the cluster flow diagram is used in the layout stage to make the layout algorithm tractable. Time runs along the x axis, and the y axis has no meaning assigned to it. Clusters are represented by horizontal lines with no thickness. The line’s leftmost coordinate corresponds to the time that the moving cluster formed and the rightmost coordinate corresponds to the time that the cluster dispersed. Each cluster is assigned a slot which defines its y-coordinate. Slots are spaced equally along the y axis. The minimum number of slots is the maximum number of concurrently defined moving clusters. As an example, the maximum number of concurrently defined moving clusters in figure 8 is three which is also the number of slots.

Links between clusters are represented as lines, again, without any thickness. The first x-coordinate of a line corresponds to the average starting time of the link, and the first y-coordinate corresponds to the slot of the starting cluster. The second x and y-coordinates are defined similarly. As an example, figure 8 has five links, one of which is horizontal. Horizontal links are always placed halfway towards the next slot and below the clusters they connect. While it is possible to test multiple locations for horizontal links, placing the links below the clusters produced reasonable results. A particular layout of some clusters is an ordered list of slots. The first element corresponds to the first cluster and so forth. For example, the layout of figure 8 is \((s_0, s_2, s_1, s_2)\).

A function is defined to describe the quality of any given layout. The function used was the weighted sum of the number of overlapping lines and the number of non-horizontal lines. The layout algorithm finds the layout with smallest value, i.e., the highest quality layout. In the prototype, we used a directed search algorithm; however, any appropriate optimisation algorithm would suffice.

5.4 Rendering the Diagram

The moving clusters are straightforward to render. Their colour is selected from a predefined list. Special care is given to ensure that no link completely covers another link which is tantamount to deleting a link. This is accomplished by putting thicker links below thinner links. In addition, horizontal links are arranged in the y direction to keep multiple horizontal links from overlapping one another, e.g., see the horizontal links in figure 10.

6 Results

The prototype was developed as a web application using KinectJS and KnockoutJS. No processing times are given because they varied significantly between executions most likely due to the browser environment. This section presents representative cluster flow diagrams generated from four real world datasets and one synthetic dataset. The parameters for each dataset are given in table 1.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>VAST</th>
<th>Truck</th>
<th>Eye</th>
<th>Imis</th>
<th>Storm</th>
</tr>
</thead>
<tbody>
<tr>
<td>time slices</td>
<td>100</td>
<td>500</td>
<td>200</td>
<td>200</td>
<td>100</td>
</tr>
<tr>
<td>eps</td>
<td>5</td>
<td>1000</td>
<td>60</td>
<td>45</td>
<td>320</td>
</tr>
<tr>
<td>min points</td>
<td>5</td>
<td>10</td>
<td>15</td>
<td>10</td>
<td>15</td>
</tr>
<tr>
<td>min count (a)</td>
<td>10</td>
<td>50</td>
<td>10</td>
<td>10</td>
<td>15</td>
</tr>
<tr>
<td>min duration</td>
<td>7</td>
<td>25</td>
<td>15</td>
<td>20</td>
<td>20</td>
</tr>
<tr>
<td>max member time</td>
<td>10</td>
<td>10</td>
<td>0</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>max absentee time</td>
<td>10</td>
<td>20</td>
<td>10</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>max time (b)</td>
<td>15</td>
<td>15</td>
<td>15</td>
<td>15</td>
<td>15</td>
</tr>
<tr>
<td>min count (c)</td>
<td>12</td>
<td>5</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>min count (d)</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>2</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 1: (a) the minimum number of members in a cluster, (b) the max time between transitions for them to grouped in a link, (c) the min number of transitions in a link for it to be used in the layout stage, (d) the min number of transitions in a link for it to be drawn

6.1 Truck Dataset

The revised trucks dataset (retrieved from http://www.chorochronos.org/?q=node/10) contains the trajectories of 50 trucks as they deliver concrete around the Athens metropolitan area in Greece. Initially there was no apparent grouping in the trucks; however, when the trajectories were shifted to start at the same time, the trucks that took similar routes appeared to move in groups. The truck dataset cluster flow diagram (figure 9) shows that initially the dataset is split up into three clusters. It is interesting to note that the noise cluster is the smallest of the three which indicates that the system is initially well organised. All of the clusters decrease in volume over time because the trucks have unequal lifetimes.

The diagram shows that the three main clusters are defined throughout the lifetime of the dataset. The three bottom clusters can be seen to form one cluster that breaks up and reforms over time.

6.2 Evacuation Dataset

The VAST dataset (IEEE VAST 2008 Challenge 2008) (figure 1) is a synthetic dataset depicting of-
office workers evacuating a building in response to a bomb that has exploded. Besides changing the file format, this dataset was not modified. The leftmost part of the diagram shows four clusters and a large noise cluster. This indicates that initially a noteworthy number of workers are scattered throughout the building. In the middle of the diagram, the clusters converge on the yellow cluster and the noise cluster which is the response to the bomb. Next, the yellow cluster breaks up into smaller clusters which correspond to the evacuation zones. Hence, the workers have reorganised themselves in response to the explosion. The decrease in thickness of the noise cluster indicates that the system is more organised at the end of the dataset than at the beginning.

6.3 Gaze Tracking Dataset

Besides changing the file format, this dataset was not modified. The gaze tracking dataset (KASPROWSKI 2004) (retrieved from http://www.kaggle.com/c/emvic/data) shows the point of regard of a group of human test subjects in response to an animated visual stimulus. Note that only the right eye’s point of regard from the test dataset was used. The resulting diagram (figure 10) has some distinctive features. The noise cluster consistently increases in volume indicating that the behaviour of the test subjects is becoming more random as the experiment progresses. There are no links from the noise cluster to any other cluster; this is another indication that the system is tending to disorder. Lastly, the system alternates between well organised and completely random as evidenced by the sequence of clusters. The decreasing width of the clusters indicates that the subjects are becoming desensitised to the organising stimulus as the experiment progresses.

6.4 IMIS Dataset

The Imis3days dataset (retrieved from http://www.chorochronos.org/?q=node/8) was collected by IMIS Hellas S.A. It contains the movements of naval vessels. In order to make parameter tuning and loading the file faster, the sampling frequency of this dataset was reduced from once every ~26 seconds (on average) to once every minute at most. This reduced the data to ~14% of the original number of samples and the file size from ~200Mb to ~20Mb. The average distance between samples beforehand was ~64m, and afterwards ~425m. If desired, once the parameters have been determined, a cluster flow diagram could be computed from the full dataset. Lastly, the latitude/longitude coordinates were converted to Cartesian coordinates using an equirectangular projection. The resultant cluster flow diagram (figure 11) is perhaps the most stable of all of the datasets. All of the clusters have a reasonably constant width and there are three clusters that are defined for the whole lifespan of the dataset. Lastly, this diagram is shown with (smaller diagram) and without (larger diagram) the links to the noise cluster. Note that the diagram with links to the noise cluster appears more cluttered and less intelligible.

6.5 Storm Dataset

IBTrACS-WMO v03r04 (The International Best Track Archive for Climate Stewardship-World Meteorological Organisation) dataset (retrieved from http://www.ncdc.noaa.gov/ibtracs/index.php?name=wmo-data) contains the movements of hurricanes from the mid 1800’s to modern times. The storms were shifted to start at the same time for the same reason as the trucks dataset. Also, the latitude/longitude coordinates were converted to Cartesian coordinates using an equirectangular projection. The dataset was truncated at ~13 days because only a small number of storms had a lifespan that long, ~7% of the hurricanes; otherwise, it would produce a very wide diagram. It resulted in a loss of only ~3% of the samples. The cluster flow diagram shown in figure 12 does not offer much insight into this dataset because it is well organised and remains well organised. The storms appear to be naturally grouped. It may be more insightful to examine each cluster in isolation. Again the diagram is shown with (smaller diagram) and without (larger diagram) links to the noise cluster.

7 Summary, Discussion and Future Work

This paper introduces cluster flow diagrams as a snapshot of all of transfers of membership between groups and the formations and dispersals of groups themselves within a given a dataset. It also presented a four stage methodology to generate these diagrams. Additionally, the methodology has considerations to...
remove noise and unwanted artifacts of the clustering process. Additionally, we proposed two composite visualisations to further characterise the groups in terms of their members' movements and environment. This approach allows domain experts and analysts to identify at a glance trends in the membership of groups and systematic transfers of membership between groups. Additionally, the noise cluster allows an analyst to see trends in how organised a dataset is over time.

The result section showed a number of representative cluster flow diagrams. Parameters must be set according to the research questions of the analyst; they cannot be set universally for a particular dataset. Some research questions may be more tolerant to filtering out data, whilst others may require all of the data to be represented. In the future, we plan on developing case studies to show the insight generating capabilities of the visualisation. In the process of doing this, we will develop visualisations to aid the setting of the parameters.

Some of the possible areas for future work include: encoding spatial attributes of the groups on the cluster flow diagram; introducing interactivity to allow users to isolate subsets of the data in which they are interested; and allowing users to interactively construct and modify parameters of cluster flow diagrams.

References


**URL:** [www.cs.umd.edu/hcil/VASTchallenge08](http://www.cs.umd.edu/hcil/VASTchallenge08)


A Machine Learning approach to Generic Entity Resolution in support of Cyber Situation Awareness

Christopher Moir and Jonathan Dean
Defence Science and Technology Organisation
PO Box 1500, Edinburgh 5111, South Australia

Abstract

This paper introduces the Generic Entity Resolution (GER) framework; a framework that classifies pairs of entities as matching or non-matching based on the entities’ features and their semantic relationships with other entities. The GER framework has been developed as part of an AI-based system for the development of Cyber situational awareness and provides a data fusion role by resolving entities discovered across multiple disparate data sources. The approach utilizes supervised machine learning to identify the set of features and semantic relationships that result in the optimum classification accuracy. We evaluated the GER framework using several well-known data sets and compare the framework’s accuracy to existing state-of-the-art resolution algorithms. We found that the GER framework’s accuracy compares favourably to existing state-of-the-art resolution algorithms for the data sets used in this evaluation.

Keywords: Entity Resolution, Machine Learning, Situation Awareness, Genetic Algorithm

1 Introduction

A much-enhanced cyber situation awareness capability is a priority for Defence in support to cyber warfare (Department of Defence 2009). The widely held definition of situation awareness, as put forward by Endsley (1988), is:

'\textit{the perception of the elements in the environment within the volume of time and space, the comprehension of their meaning and the projection of their status in the near future}'

While the definition holds for the cyber domain, generating and maintaining cyber situation awareness is an increasingly challenging task as adversaries become more capable and malware increases in both volume and technical sophistication (Onwubiko and Owens 2012, McAfee Labs 2013, Symantec Corporation 2013). Further, the scope and the complexity of the cyber domain are significantly higher than other domains (McMillan and Tyworth 2012).

Situation awareness in the cyber context has been traditionally generated by a series of techniques such as vulnerability assessment, intrusion detection or digital forensics which are applied at a low-level of data and abstraction (Barford, Dacier et al. 2010). This requires the human operator to develop and maintain the required higher level situation awareness. Commonly this ‘picture’ is manifested through manual, time-consuming tasks defined by standard operating procedures and kept as a mental model in the analyst’s head, aided by tools such as Security Information and Event Management Systems (SIEM). This approach is neither scalable nor sustainable due to the complexity of the cyber environment. Cyber situation awareness must be improved for decision makers and considered for automated systems (Blumenthal, Haines et al. 2012).

One of the requirements for the construction of cyber situation awareness (for man or machine) is data fusion from multiple disparate sources (Onwubiko and Owens 2012). This includes entity resolution, which is defined as the process of “identifying entities (objects, data instances) referring to the same real-world entity” (Köpcke and Rahm 2010). The process of entity resolution combines multiple observations of an object into a unified representation. Further, the use of a heterogeneous selection of sources, including both typical sources (IDS alerts, network capture and audit logs) with atypical sources (corporate directories, travel documents, business forms) allows for a holistic approach providing a richer representation of entities and a broader context for situational awareness (Grove, Murray et al. 2013).

Performing entity resolution across disparate sources is not trivial. Observations from sensors are not always complete, may not uniquely nor explicitly identify entities present and may come in a variety of formats. Further, rule-based methods require hand-tuning to perform well and are not robust over time (Grove, Murray et al. 2013).

We believe that the development of comprehensive cyber situation awareness will leverage a number of AI and fusion techniques to deal with the volume of data, along with the uncertain, incomplete, erroneous and conflicting nature of information and information sources. In this paper we present one technique known as generic pair-wise entity resolution for consideration as part of a broader integrated solution.
Generic pair-wise entity resolution is designed to resolve entities from different sensors with varying attributes and levels of completeness in their representation. Further, our approach differs from other pair-wise techniques as it: firstly, learns which entity features are best for resolving entities; secondly, learns which metrics for a feature result in optimal resolution accuracy; thirdly, utilizes the semantic associations, or relationships, between entities to enhance the accuracy of resolution; and, finally, uses existing machine learning techniques to avoid hand-tuning or utilising domain-specific expert rules, thereby providing a high level of automation to the resolution process.

The underlying assumption of the proposed generic pair-wise entity resolution algorithm is that two instances with similar values for one or more features, or attributes, are more likely to represent the same real-world entity than two records that do not have any values that are the same. If some of these features are the same then the two records may represent matching entities. Conversely, if none of these features are the same then the two records are more likely to represent non-matching entities.

The contributions of this paper are twofold. Firstly, this paper defines the individual processing steps that form our proposed generic entity resolution algorithm. Secondly, this paper evaluates the generic resolution algorithm against existing resolution algorithms using several publically available data sets.

This remainder of this paper is structured as follows: Section 2 discusses existing entity resolution techniques and frameworks; Section 3 describes in detail how the proposed entity resolution framework works; Sections 4 and 5 report on the evaluation of the proposed entity resolution framework using existing data sets; Section 6 provides a discussion on the efficacy and limitations of the proposed entity resolution framework; Sections 7 and 8 contain the conclusions and future work, respectively, for this research.

2 Related Work

The problem of entity resolution is well-studied, and has led to the development of many different algorithms for resolving entities. This section reviews some of these techniques for performing entity resolution, including those that utilize machine learning algorithms to resolve pairs of instances.

The Stanford Entity Resolution Framework (SERF) projects (Benjelloun, Garcia-Molina et al. 2009) have developed a generic resolution framework that focuses primarily on improving the efficiency of entity matching. They were less concerned with how to match instances, instead choosing to focus their attention on developing efficient matching algorithms that minimize the number of comparisons between database records. They defined 3 algorithms for entity resolution: G-Swoosh, R-Swoosh and F-Swoosh. These algorithms are optimized for matching and merging database records. They each make certain assumptions about the record matching and merging operations. The G-Swoosh algorithm is the most general, but least efficient, of all 3 algorithms. The F-Swoosh algorithm can be significantly more efficient than the R-Swoosh algorithm by avoiding repeated feature comparisons (Benjelloun, Garcia-Molina et al. 2009).

Zhao and Ram (2005) propose a multiple classifier system approach that utilizes a variety of supervised machine learning algorithms to resolve records in a database. These algorithms include: neural networks; k-nearest neighbours; decision trees; Naïve Bayes; and linear and logistic regression. Individual record fields are compared using exact matching, sub-string matching, Soundex or Levenshtein’s string edit distance (Levenshtein 1966). Zhao and Ram’s multiple classifier system combines the outputs from multiple individual classifiers to derive an overall resolution. The multiple classifier system uses either bagging (Breiman 1996), boosting (Schapire and Freund 2012) or cross-validated committees (Parmanto, Munro et al. 1995) to combine the outputs from homogeneous base classifiers; for heterogeneous base classifiers it uses cascade generalization (Gama and Brazdil 2000) or stacked generalization (Wolpert 1992).

Bilenko and Mooney’s (2003) Multiply Adaptive Record Linkage with Induction (MARLIN) combines multiple string similarity matchers using a Support Vector Machine (SVM), which is a maximal-margin, kernel-based classifier (Cortes and Vapnik 1995). MARLIN utilizes a two level learning approach: at the first level, string similarity measures are trained to estimate the similarity between values in the same database field; at the second level a SVM is trained to identify when two records match using the similarity measures learnt at the first level. MARLIN supports two methods for selecting training data in a semi-automatic manner: static-active and weakly-labelled negative selection. In static-active selection, near duplicate pairs of instances are identified by comparing instances to a string similarity measure and selecting only those pairs that are classified as similar according to this measure. In weakly-labelled negative training selection, MARLIN randomly selects entity pairs that have few shared values as these pairs are least likely to be duplicates. A human operator then verifies each pair is correctly labelled in the set produced by either method.

Christen’s (2008) Freely Extensible Biomedical Record Linkage (FEBRL) application allows users to match biomedical records. To match record pairs, the user must first manually select which attributes of the biomedical records to match. FEBRL contains 26 different similarity measures for matching attribute values; most being variations of well-known approximate string comparison algorithms. FEBRL also contains other special functions for comparing numerical values, or fields that contain date, age or time values. The comparison functions all return a similarity value in the range [0, 1], where a score of 0 signifies total dissimilarity between feature values and 1 signifies an exact match. The user-selected attributes form a
vector of similarity scores for each record pair. FEBRL classifies the record pair as either matching or not matching based on the similarity scores.

The Self-Tuning Entity Matching (STEM) (Köpcke and Rahm 2008) framework automatically constructs strategies for matching entities based on their attributes, or features. STEM consists of three principal steps: firstly, the generation of training data; secondly, the computation of attribute similarity for the entity pairs in the training data; and, thirdly, learning the overall entity resolution strategy. STEM combines the output from the individual similarity measures using SVMs, decision trees, logistic regression, or some combination of these 3, to form its entity resolution strategy. An entity pair’s overall resolution is determined by the output from STEM’s entity resolution strategy.

3 The Generic Entity Resolution Framework

The Generic Entity Resolution (GER) framework is a software architecture for determining whether pairs of entities in data represent the same real-world entity. The GER framework consists of separate software modules that classify an entity pair as matching or not matching by comparing both entities’ values for a single feature, or attribute, only. Each software module is called a generic feature resolver, or simply a generic resolver. The ‘first name’ generic resolver in Figure 1 classifies two people as matching if it deems their first names the same. If the ‘first name’ resolver deems the first names different then it will classify the two people as non-matching. The resolvers are termed generic as they are not specific to an individual feature or data type; the generic resolver’s operation is the same for every feature and data type.

![Composite Resolver](image)

Figure 1: Conceptual overview of GER framework illustrating how individual generic resolvers for each comparison feature are combined in a Naïve Bayes network to obtain an overall resolution for pairs of instances

Each generic resolver is comprised of four main components: a set of metrics, which are software functions that map two feature values to a real-valued number in the range [0, 1]; a SVM; a genetic algorithm (Sivanandam and Deepa 2008); and a Reinforcement Learning algorithm known as the Q-learning algorithm (Mitchell 1997). Each function calculates a real-valued number representing a normalized similarity score for the two feature values. The SVM uses these similarity scores to determine the optimum decision boundaries for classifying instances as matching. The genetic algorithm identifies the set of metrics that result in the greatest overall resolution accuracy. Some features can also have more than one value. For example, a person may have multiple variations and spellings of their first name. Each generic resolver uses the Q-learning reinforcement learning algorithm to learn how many values must match before it classifies two instances as matching.

The generic resolvers are combined together to form a naïve version of a Bayesian network (Pearl 1988). The GER framework in Figure 1 resolves two people by comparing their first name, family last name and date of birth. Each comparison feature has its own instance of a generic resolver. Each generic resolver operates independently of the others; the output from one generic resolver does not influence the output from the others.

The root of the Bayesian network, as shown in Figure 1, is a Composite Resolver that classifies two entities as matching or non-matching based on the output from one or more generic resolvers. The Composite Resolver calculates the probabilities that two people match and do not match given the classifications from the first name, last name and date of birth resolver. The Composite Resolver classifies two people as ‘matching’ if the probability they match exceeds the probability they do not, otherwise it classifies both people as ‘not matching’.

The novelty of the GER framework arises from the way it utilizes existing research to resolve entities. The SVM, Q-Learning algorithm, genetic algorithm and Bayesian Network are all used as ‘off-the-shelf’ components in the GER framework. The GER framework’s novelty arises from the way it uses these ‘off-the-shelf’ components to learn: firstly, how to classify each feature differently from the others; and, secondly, which feature classifications it should use to resolve a pair of instances.

There are two distinct phases in the algorithm for constructing the Bayesian network of generic resolvers. In the first phase, individual generic resolvers are created to classify instances as matching or non-matching based on the values of a single feature. In the second phase, a composite resolver is created to combine the output from one or more generic resolvers to determine an overall classification for pairs of instances. The output from the first phase of the algorithm is a set of generic resolvers that have been optimized for resolving pairs of instances using a single assigned feature only. The output of the second phase of the algorithm is a Bayesian network consisting of one or more generic resolvers that can resolve pairs of instances based on their feature values.

The procedure for creating a set of generic feature resolvers is described in Figure 2. Each feature resolver
utilizes a genetic algorithm to identify the set of metrics that results in the greatest resolution accuracy for the instances of that feature in the validation data. During the optimisation process, the training data is transformed to coordinates using a subset of metrics determined by the genetic algorithm. The SVM is then trained with these coordinates. The SVM’s empirical risk functional, which is an estimate of the SVM’s expected classification error (Vapnik 1995), is calculated using a distinct validation set. The SVM with the lowest empirical risk functional is retained. This process is repeated until the genetic algorithm terminates. Finally, if an instance has more than one value for a feature, such as multiple email addresses or phone number, the Q-learning algorithm is utilized to determine the optimum number of values from the training data that must match to classify pairs of instances as the same.

Constructing a set of generic feature resolvers

Let \( E \) be the entity type to resolve, let \( F \) be set of features for matching entities, let \( M_f \) be the set of metrics for comparing the feature, \( f \):

1. For each \( f \in F \):
   a. Create generic resolver, \( g_f \), for \( f \).
   b. Set initial metrics set for \( g_f \) to \( M_f \).
   c. Run genetic optimization algorithm for \( g_f \) to find optimal metrics set for resolving instances based on \( f \). At each iteration, \( j \), in genetic algorithm:
      i. Convert labelled training data for \( f \) to set of labelled feature space coordinates using current evolved metrics set, \( M_f' \subseteq M_f \).
      ii. Convert labelled independent validation data for \( f \) to set of labelled feature space coordinates using metrics set in i).
      iii. Train SVM using coordinates from i).
      iv. Classify coordinates from i) using SVM.
      vi. If empirical risk for SVM is lower than a previous SVM, \( 0 < k < j \), then set optimal SVM and optimal metrics set for \( g_f \) to SVM and \( M_f' \) respectively; otherwise discard SVM and \( M_f' \).
      vii. Repeat steps i) to vi) until genetic algorithm terminates.
   d. If 1 or more instances of \( E \) has more than 1 value for \( f \) then use Q-learning algorithm to determine the optimum number of matches for classify instances of \( E \) as matching.

Figure 2: the algorithm for constructing a set of generic feature resolvers

In the second phase of the algorithm, the composite resolver identifies the set of features that results in the greatest overall resolution accuracy. The steps for creating the composite resolver are described in Figure 3. The composite resolver learns which features are optimal for resolving pairs of entities from the set of features present in the training data. Instead of arbitrarily choosing the features to resolve pairs of entities, the generic resolver learns the optimal set of features from the training data using a genetic algorithm. The composite resolver combines the output of the generic resolvers for each of the features in the optimal feature set to obtain a single overall classification for a pair of instances.

The GER framework has a library of different metrics, or algorithms, for comparing string values, including: Dice’s coefficient; Soundex; Metaphone; Caverphone; regular expression comparison; exact match; Levenshtein; Nysiis; and diffliibratio, which is the sequence similarity ratio calculated by the SequenceMatcher class in the difflib Python library. These comparison metrics all differ significantly in how they compare strings. For example, the Dice’s coefficient algorithm calculates the proportion of bigrams, which are sequences of two adjacent elements in a string, which match in both strings. The Soundex, Metaphone, Nysiis and Caverphone algorithms match words based on their phonetics. These algorithms encode the phonetic sounds in the English language into their own unique intermediate form so that they can match words that sound the same. The regular expression algorithm classifies two strings as matching if both strings fit a specified regular expression pattern. The Levenshtein algorithm calculates the number of insertions, deletions and mutations necessary to convert one word into another. The Levenshtein algorithm deems two words as more alike if fewer transformations are required to convert one string into the other.

Constructing the Bayesian network of generic feature resolvers

Let \( E \) be the entity type to resolve, let \( F \) be set of features for matching entities, let \( g_f \) be the generic resolver for feature, \( f \):

1. Create Composite Resolver, \( C_E \), for \( E \).
2. Set initial feature set for \( C_E \) to \( F \).
3. For each \( f \in F \):
   a. Add \( g_f \) to \( C_E \).
4. Run genetic algorithm for \( C_E \) to find optimal Bayes Network for resolving instances. At each iteration, \( j \), in genetic algorithm:
   a. Create Bayes Network, \( BN_f \), containing only \( g_f \) for features in current evolved feature set, \( F_j \subseteq F \).
   b. Convert labelled training data for \( E \) to set of labelled feature space coordinates for each \( f \in F \).
   c. Classify coordinates in 4b) using \( BN_f \).
   d. Calculate \( BN_f \) empirical risk using classifications in 4c).
   e. If empirical risk for \( BN_f \) is lower than previous \( BN_f \), \( 0 < k < j \), then set optimal Bayes network for \( C_E \) to \( BN_f \), otherwise discard \( BN_f \).
   f. Repeat steps 4a) to 4e) until genetic algorithm terminates.

Figure 3: the algorithm for creating the composite generic resolver

The GER framework also has metrics defined for comparing dates, time and computer network address. The date and time metrics permit an exact match using the day, month, year, hours, minutes and seconds. The date and time metrics also define a total ordering by virtue of the ‘greater than’ operator.

Multiple instances of the same class of metric were included in the comparison metric library if there were differences in a metric’s operation or the metrics were not functionally identical. For example, the Metaphone and Soundex algorithms both belong to the class of phonetic algorithms. Metaphone and Soundex use different sound encodings for phonetic representation. Metaphone represents ‘ck’ in a word using the letter ‘k’ while Soundex represents each ‘c’ and ‘k’ in a word
using the number 2. Both algorithms were therefore included in the generic resolver’s set of candidate metrics due to their markedly different operation. Three different versions of the Soundex algorithm were also included as each version represented their sound encodings slightly differently. Similarly, ‘strike a match’ and Dice’s coefficient were also included due to slight differences in implementation approaches between our algorithms. Since it is not known a priori which implementations of a given algorithm will provide better results, it was decided to include them all in the set of comparison metrics and have the genetic algorithm determine which implementation is better for the training data supplied.

The GER framework can also utilize the associations, or semantic relationships, between different entities to help resolve pairs of instances. For example, there is a ‘many-to-many’ association between movies and actors: many actors appear in a single movie, and a single actor can appear in many different movies. A movie also has its own attributes, including title, viewer advisory rating, synopsis and release date. These attributes may not permit the GER framework to resolve movies accurately, since different movies may have the same viewer advisory rating or release date. The movie title and synopsis may also not prove useful for resolving movies as a movie is sometimes released under another title in other countries. Further, remakes are considered different movies, but often share the same title and synopsis. However, in the first instance a movie will share the same actors, while for the later the movies are unlikely to share the same actors. The inclusion of the associations between a movie and its actors may therefore improve the GER framework’s overall resolution accuracy.

We hypothesized it is possible to resolve entities by learning: first, an ‘optimal’ set of features for resolving the entities; second, an ‘optimal’ set of similarity metrics for each feature; and, thirdly, the similarities that demarcate two entities that are the same from two entities that are different. If this research hypothesis is true then the GER framework will correctly resolve entity pairs after undergoing an initial training process to learn the set of optimal features, optimal metrics for each feature and the similarities that designate two entities as the same. This prediction presumes that the optimal features, the optimal metrics for each feature, and the similarities that designate entities as the same, are learnable from the training data. To examine this research hypothesis, a series of evaluations was performed, as detailed in Section 4.

4 Methodology

The GER framework was evaluated using data from the Fodor and Zagat restaurant guides, the Canadian Opinion Research Archive (CORA), and the Abt-Buy e-Commerce data set. These data sets were chosen to evaluate the proposed GER framework because: firstly, they capture the imperfections and nuances typical of real-world data; secondly, these two data sets have previously been used to evaluate other entity resolution algorithms and techniques; and, thirdly, Defence owned cyber-related data sources were not releasable, or hampered open publishing of results. The Abt-Buy data set was selected because this data set is challenging to resolve (Köpcke, Thor et al. 2010). The CORA and Abt-Buy data sets both contain instances with missing information; using these data sets therefore permits an evaluation of the GER framework with incomplete data. Evaluating the proposed GER framework using the Restaurant, CORA and Abt-Buy data sets also provides a direct comparison between the GER framework and existing entity resolution algorithms.

The GER framework was also evaluated using data from the IMDB and themoviedb.org motion picture databases. Two different configurations for the GER framework were evaluated. In the first configuration, the GER framework resolved pairs of movie records using only the basic attributes of the movie, such as the title or synopsis. In the second configuration, the GER framework used the basic attributes of the movie and the association between a movie and its actors to determine if two movies matched. The Mann-Whitney U test (Mann and Whitney 1947) was used to determine whether the association between a movie and its actors significantly increased the framework’s resolution accuracy.

Separate training, validation and test sets were used to evaluate the GER framework. The SVM kernel function and set of metrics for comparing feature values were learnt from the training data. The validation set was used as a pseudo-test set: during the genetic optimization phase, each SVM was repeatedly evaluated against the validation set to identify which type of SVM kernel and set of metrics produced the greatest resolution accuracy.

A total of 30 evaluations were performed for each of the CORA, Restaurant, Abt-Buy and IMDB-themoviedb.org data sets. New training, validation and test sets were generated for each evaluation. The resolver’s F-measure, true positive rate and false positive rate were calculated for each evaluation run; resulting in a sample size of 30 for all three measures. Each evaluation was performed on a Fedora 19 64-bit virtual machine running on an IBM HX5 blade, with 2 Intel Xeon E7-2830 2.13 GHz CPUs, a 100 GB Hard Disk Drive and 110 GB of RAM.

4.1 Data

The Fodor and Zagat restaurant data set consists of 864 records. Restaurants are distinguished by the following four features: name, address, city, and restaurant type. Restaurant telephone numbers were not included in the data set since they are known to artificially ‘boost’ the resolution accuracy. 112 pairs of records are related to the same restaurants. Figure 4 shows two matching records from the guide. The records for the same restaurant do not match precisely, suggesting that naïve comparison techniques, such as exact string
comparison, are unlikely to accurately resolve pairs of restaurants.

\[\text{la cote basque}, \, 60 \, w. \, 55th \, st. \, between \, 5th \, and \, 6th \, ave., \, \text{new york}, \, \text{french} \]
\[\text{la cote basque}, \, 60 \, w. \, 55th \, st., \, \text{new york city}, \, \text{french (classic)}\]

Figure 4: matching restaurants from the Restaurant data set

The CORA data set consists of 1295 academic publication citations to 122 computer science research papers. The CORA data set records the following 12 features of a publication: author, volume, title, institution, venue, address, publisher, year, pages, editor, note, and month. Figure 5 shows two matching instances from the CORA data set. The records do not match precisely, suggesting that naïve comparison techniques, such as matching the titles exactly, are unlikely to accurately resolve pairs of publications from the CORA data set.

\[\text{kearns, m.}, \, \text{a bound on the error of cross validation using the approximation and estimation ratios, with consequences for training-test split}, \, \text{neural information processing 8,} \, \text{morgan kauffmann,} \, \text{\textcopyright (1996)}, \, \text{pp. 183-189, \, \textquotesingle\textquotesingle ed: d.s. touretzky, m.c. mozer and m.e. hasselso.}\]

\[\text{m. kearns, a bound on the error of cross validation, with consequences for the training-test split, in advances in neural information processing systems 8, \textquotesingle\textquotesingle the mit press,} \, \text{\textcopyright 1996, \, \textquotesingle\textquotesingle to appear.}\]

Figure 5: matching publications from the CORA data set

The Abt-Buy data set consists of over 1000 items for sale at both the Abt.com and Buy.com e-Commerce stores. The Abt.com e-Commerce store records an item’s name, description and price; Buy.com records an item’s name, description, price and also the manufacturer. Figure 6 shows two matching instances from the Abt-Buy data set. The first instance in Figure 6 has values for the name and manufacturer features, but no values for the price or description features. The second instance in Figure 6 represents the same model of TV. It has values for the name and description features but not for manufacturer or price. These two instances are therefore only comparable by their name.

\[\text{Samsung LN32A450 32'720p LCD HDTV, \textquotesingle\textquotesingle Samsung}\]
\[\text{\textquotesingle\textquotesingle Samsung 32' Black Flat Panel Series 4 LCD HDTV - LN32A450, \textquotesingle\textquotesingle Samsung 32' Black Flat Panel Series 4 LCD HDTV - LN32A450/ 10,000:1 Dynamic Contrast Ratio/ 1366 x 768 True 720p Resolution/ 6ms Response Time/ Cold Cathode Fluorescent Lamp (CCFL)/ Hidden Bottom Speakers/ SRS TruSurround XT/ Built-In ATSC/Clear QAM Tuner/ V-Chip System/ Swivel Stand/ Black Finish}\]

Figure 6: matching items from the Abt-Buy data set

The IMDB and themoviedb.org databases contain detailed information about movies, the cast of actors, the directors and crew. In total, data about 11,992 movies and 56,670 actors were retrieved from the IMDB and themoviedb.org databases. The IMDB and themoviedb.org websites have publically-accessible Application Programming Interfaces (APIs) that allows individuals to retrieve data about movies, the cast of actors, the directors and crew. Both websites format information about a movie using JavaScript Object Notation (JSON).

There are several differences between the data obtained from the IMDB and themoviedb.org databases. The data from themoviedb.org is generally more detailed and contains additional fields than the IMDB data. Figure 7 illustrates this for the movie: ‘The Dark Knight’. Each actor in themoviedb.org output has an id, cast id, order and character name compared to just the actor’s name from IMDB. The output from themoviedb.org also includes other information not in the IMDB output, such as: the production companies; the movie’s budget; and the total movie revenue. The IMDB output includes the languages spoken in the movie which, in the case of ‘The Dark Knight’, does not exactly match the languages reported by themoviedb.org. The IMDB output also includes fields that are not present in the output from themoviedb.org. For instance, the movie’s rating and year of release appear in the IMDB output but not in themoviedb.org output.

\{"id\: 272870, \"title\: \"The Dark Knight\", \"imdb_id\: tt0468569, \"ratings\: [\{\"vote\: 7.6, \"vote_count\: 4452, \"casts\: \[\{\"id\: 1810, \"name\: \"Heath Ledger\", \"order\: 1, \"cast_id\: 3, \"profile_path\: /yzpNhwXxVjMuqInBlBtjQ.jpg\}, \{\"id\: 525, \"name\: \"Christian Bale\", \"character\: \"Batman\", \"order\: 0, \"cast_id\: 35, \"profile_path\: /vM6jzR8hI8AM6KSo/Y.png\}\}\], \"runtime\: \"152 min\", \"type\: \"M\", \"also_known_as\: \"[\"Batman - El caballero de la noche\"]\}\}

Figure 7: Sample JSON output from the IMDB website (top) and themoviedb.org website (bottom)
Not all the available features from the IMDB-themoviedb.org data were used to evaluate the GER framework. The plot description, runtime, release date and actors were the only features used to compare movies. Features such as movie title and IMDB id were not used as these features were deemed too discriminatory; in other words, these features would likely render the resolution too easy. For example, it is possible to perfectly resolve movie instances using just their IMDB id as its value unique to each movie. Excluding movie remakes, most movie titles are unique, so resolving movie instances based on their title is likely to enhance the GER framework’s efficacy. Actor instances were compared using only their name as this is the only feature for an actor common to both the IMDB and themoviedb.org data sets.

4.2 Training, validation and test data selection

Distinct training, validation and test sets were constructed from the pre-labelled data in the Restaurant, CORA and Abt-Buy data sets. Records that referred to the same real-world entity (restaurants in the Restaurant data set, publications in the CORA data set and sales items in the Abt-Buy data sets) were constructed by pairing together records with the same label. Records that referred to different real-world entities were constructed by randomly selecting records whose labels did not match. Values for the training, validation and test sets were then randomly selected from the sets of matching and non-matching entity pairs. The training, validation and test sets were not permitted to have duplicate entity pairs. The training, validation and test sets for the Restaurant data contained 112 instance pairs; half labelled as matching and the other half labelled as non-matching. The training and validation sets for the CORA and Abt-Buy data set contained 1000 instance pairs; half were labelled as matching and the other half labelled as non-matching. The test sets for the CORA data set consisted of 200 entity pairs, while the test set for the Abt-Buy data set consisted of 100 entity pairs. The training and validation sets for the IMDB-themoviedb.org evaluation also contained 1000 entity pairs; half matching and the other half non-matching. The IMDB-themoviedb.org test sets also consisted of 200 pairs of instances.

Distinct training, validation and test sets for the IMDB-themoviedb.org evaluation were constructed using a stratified sampling approach, which is the process of dividing members of the population into relatively homogeneous groups and then sampling from these groups individually. Stratified sampling was used to ensure that movie sequels and movies sharing at least one actor were included in the training, validation and test sets. Movies that share at least one actor or are sequels are potentially more difficult for the GER framework to resolve as they are different movies, but have some of the same actors. A sequel often has some, or all, of the same cast, while movies that share at least one actor have at least one association with an actor that is the same. If all the non-matching movie instances did not have any common actors, the GER framework could achieve near-perfect resolution accuracy simply by verifying that the movies had no actors in common. Such a simplistic rule for distinguishing between matching and non-matching movies could potentially result in an overly optimistic assessment of the GER framework’s efficacy. A stratified sampling approach helped ensure that some non-matching movies in the training, validation and test sets had the same associations as the matching pairs of movies; thereby providing a less biased evaluation of the genetic resolution framework's efficacy.

5 Results

The GER framework’s median F-measure, median true positive rate and median false positive rate for the Restaurant, CORA and Abt-Buy data sets are shown in Table 1. The resolver’s median F-measure exceeded 0.96 for all three data sets. A median false positive rate of no more than 0.02 across all three data sets is strong evidence that the GER framework correctly resolved all the non-matching entities in the majority of evaluations. The GER framework’s median F-measures for all three data sets suggest that: firstly, nearly all of the resolver’s classifications of entities as ‘matching’ are correct; and, secondly, the resolver correctly resolved nearly all of the matching entities in the majority of evaluations.

<table>
<thead>
<tr>
<th>Data Set</th>
<th>Median F-measure</th>
<th>Median true positive rate</th>
<th>Median false positive rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Restaurant</td>
<td>0.963</td>
<td>0.946</td>
<td>0</td>
</tr>
<tr>
<td>CORA</td>
<td>0.964</td>
<td>0.940</td>
<td>0.01</td>
</tr>
<tr>
<td>Abt-Buy</td>
<td>0.969</td>
<td>0.960</td>
<td>0.02</td>
</tr>
</tbody>
</table>

Table 1: the GER framework’s median F-measure, true positive rate and false positive rate for the Restaurant, CORA and Abt-Buy data sets.

The distribution of F-measures, true positive rates and false positive rates in Figure 8 suggest that the GER framework is able to accurately resolve entities across multiple test sets. The greater variation in true positive rate for the Restaurant data set highlights that the resolver misclassified more matching entities than non-matching entities in several of the evaluations. A possible explanation for this is some of the matching entity pairs mapped to similar coordinates in feature space as the non-matching entity pairs.

The GER framework favours features that have high discriminatory power. The relative frequency of occurrence for the name and title features for the Restaurant, CORA and Abt-Buy data sets confirm this: the GER framework included these features in every one of the evaluations for these data sets (see Table 2). In 17 out of 30 evaluations the GER framework resolved restaurant pairs using only their names. On the other hand, the GER framework never utilized the address to resolve restaurant instances; nor did it use the editor feature and the manufacturer feature to resolve publications and e-Commerce items, respectively. The editor feature was a poor discriminator of matching and
non-matching publications because different publications may have the same editor. Manufacturer was a poor discriminator of matching and non-matching items because a manufacturer can produce many different items. The restaurant address was a poor discriminator because matching pairs of records in the Restaurant dataset do not specify the restaurant’s address identically. For example, the results in Figure 4 a) specify the same address very differently. The GER framework was unable to learn how to accurately distinguish matching and non-matching restaurants from their addresses. As a result, the resolver did not utilize the address feature to resolve pairs of records in the Restaurant dataset.

The variance in F-measure, true positive rate and false positive rate is also lower with the inclusion of the movie-actor association (see Figure 9); suggesting that the resolution accuracy is consistently greater when the relationship between a movie and its actors is included in the resolution. Given that actors frequently have unique names to disambiguate themselves from others, the inclusion of GER framework that resolves actors based on their name significantly increased the overall resolution accuracy. These results provide empirical evidence that associations between entities (movies and actors in this context) can permit the GER framework to match instances from different data sources more accurately.

<table>
<thead>
<tr>
<th>Data set</th>
<th>Feature</th>
<th>Relative frequency of feature occurrence in optimal composite resolver</th>
</tr>
</thead>
<tbody>
<tr>
<td>Restaurant</td>
<td>name</td>
<td>1.0</td>
</tr>
<tr>
<td></td>
<td>address</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>city</td>
<td>0.4</td>
</tr>
<tr>
<td></td>
<td>restaurant type</td>
<td>0.23</td>
</tr>
<tr>
<td></td>
<td>author</td>
<td>0.73</td>
</tr>
<tr>
<td></td>
<td>volume</td>
<td>0.1</td>
</tr>
<tr>
<td></td>
<td>title</td>
<td>1.0</td>
</tr>
<tr>
<td></td>
<td>institution</td>
<td>0.03</td>
</tr>
<tr>
<td></td>
<td>venue</td>
<td>0.6</td>
</tr>
<tr>
<td></td>
<td>address</td>
<td>0.27</td>
</tr>
<tr>
<td></td>
<td>publisher</td>
<td>0.27</td>
</tr>
<tr>
<td></td>
<td>year</td>
<td>0.73</td>
</tr>
<tr>
<td></td>
<td>pages</td>
<td>0.3</td>
</tr>
<tr>
<td></td>
<td>editor</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>note</td>
<td>0.37</td>
</tr>
<tr>
<td></td>
<td>month</td>
<td>0.1</td>
</tr>
<tr>
<td></td>
<td>name</td>
<td>1.0</td>
</tr>
<tr>
<td></td>
<td>manufacturer</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>description</td>
<td>0.6</td>
</tr>
<tr>
<td></td>
<td>price</td>
<td>0.76</td>
</tr>
</tbody>
</table>

Table 2: the proportion of occurrences for each feature in the optimal composite resolver, across all 30 evaluations for each data set.

The GER framework’s accuracy compares favourably to other entity resolvers’ accuracy for the Restaurant and CORA data sets. Bilenko and Mooney (2003) reported maximum F-measures of 0.922 for the Restaurant data set and 0.867 for the CORA data set using their MARLIN system. The GER framework’s F-measure is greater than MARLIN’s maximum F-measure for both the CORA and Restaurant data sets in over 90% of evaluations. Köpcke and Rahm (2008) attained a maximum F-measure of 0.97 for the Restaurant data set with their STEM algorithm. The GER framework attained a F-measure of at least 0.97 in 13 out of 30 test runs. Chaudhuri, Chen et al. (2007) reported a F-measure of 0.985 for the Restaurant and
CORA data sets using their record matching operator tree algorithm. The authors do not specify whether they repeated their evaluation using different training sets. As a result, it is difficult to assess whether the F-measure of 0.985 is typical for their algorithm. Cohen and Richman (2002) attained maximum F-measure values of 0.964 and 1 for the CORA and Restaurant data sets, respectively. However, their evaluation consisted of only 2 test runs using separate training and test data sets. The GER framework attained a F-measure of at least 0.964 in 16 out of 30 evaluations for the CORA data set. In 20 out of 30 evaluations for the Restaurant data set, the GER framework’s F-measure is greater than Cohen and Richman’s lowest F-measure. Together, these results suggest the GER framework’s accuracy for the Restaurant and CORA data sets is similar to existing state-of-the-art resolution algorithms.

The GER framework’s accuracy also compares favourably to other entity resolvers’ accuracy for the Abt-Buy data set. The GER framework’s lowest F-measure score for the Abt-Buy data set is greater than the maximum F-measures reported in Kopcke, Thor et al. (2010) for the same data set. The results reported in Kopcke, Thor et al. also suggest that the GER framework outperforms FEBRL (Christen 2008) and MARLIN (Bilenko and Mooney 2003) for the Abt-Buy data set. Kopcke, Thor et al. (2010) found that FEBRL’s and MARLIN’s F-measures for the Abt-Buy data set never exceeded 0.8, even for larger training set sizes. The GER framework attained a median F-measure of 0.969 for the same data set. It should be noted that FEBRL and MARLIN only used at most two attributes to resolve instances in the Abt-Buy data set, while the GER framework utilized up to 3 attributes.

<table>
<thead>
<tr>
<th>Data set</th>
<th>Median F-measure</th>
<th>Median true pos rate</th>
<th>Median false pos rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Without movie-actor relationship</td>
<td>0.67</td>
<td>0.53</td>
<td>0.03</td>
</tr>
<tr>
<td>With movie-actor relationship</td>
<td>0.97</td>
<td>0.95</td>
<td>0.00</td>
</tr>
</tbody>
</table>

Table 3: the GER framework’s median F-measure, true positive rate and false positive rate for the IMDB-themoviedb.org data set

A plausible alternative explanation for the GER framework’s apparent efficacy is inadvertent interdependencies between the training, validation and test sets. Test data that map to the same coordinates in feature space as the training or validation data would most likely introduce optimistic bias in the evaluation. In effect, the GER framework is being evaluated using the same data it was trained on. The GER framework was therefore re-evaluated using training, validation and test data sets that do not map to the same coordinates in feature space.

Re-evaluating the GER framework using CORA test sets that do not map to the same feature space coordinates reduced the framework’s median F-measure by 0.03. The F-measures exhibited greater variation for the independent test set, suggesting the framework is unable to resolve pairs of records that represent the same publication as consistently in the independent test set. The Mann-Whitney U test results confirm the framework’s efficacy is lower for the independent CORA test sets. The p-value for the F-measures using partially dependent and independent test sets was 0.00018, which is statistically significant at α = 0.05. Since the median F-measure is lower for the independent test data, it is reasonable to conclude there is an overall reduction in the framework’s efficacy when it was evaluated using independent test data. This result is evidence of some optimistic bias in the framework’s initial evaluation using the CORA data set. Even accounting for this optimistic bias, the GER framework’s precision and recall for the CORA data set still exceeds 90%. It is therefore reasonable to conclude the GER framework’s efficacy for resolving the CORA data set is not merely due to test data mapping to the same feature space coordinates as the training or validation data.

Further evidence that the GER framework’s efficacy is not solely attributable to optimistic bias was obtained from the results for the Restaurant data set. In one evaluation less than 4% of the test data mapped to the same coordinates as either the training or validation data. The F-measure obtained in this evaluation was 0.943. The F-measure declined by nearly 2% to 0.926 when the duplicate data was removed from the test set. In a second evaluation, less than 8% of the test data mapped to the same coordinates as either the training or validation data. The F-measure fell from 0.982 to 0.935 when the duplicate test data was removed from the test.
set; a decrease in F-measure of nearly 5%. Even with the duplicate data removed from both test sets, the GER framework’s precision and recall is still high; supporting the claim that the GER framework’s efficacy is not solely attributable to any optimistic bias arising from partially dependent test data.

6 Discussion

The experimental results support the research hypothesis that it is possible to resolve instances by learning: the set of features for resolving the entities; the set of similarity metrics for each feature; and the similarities that demarcate matching and non-matching entities. The GER framework’s accuracy for the Restaurant, CORA, Apt-Buy and IMDB-themoviedb.org data sets confirms the viability of our proposed machine learning approach to entity resolution. The re-evaluation of the GER framework using independent test data provided further support for the research hypothesis. Even though a slight decline in the resolver’s median F-measure was noted with independent test data, the GER framework could still correctly resolve matching and non-matching entity pairs. Together, the experimental results provide strong evidence in support of the research hypothesis.

Examination of the GER framework’s output revealed that it failed to correctly resolve CORA citations representing matching publication that humans might also find difficult to resolve. For example, the GER framework failed to correctly resolve the citations in Figure 10 a) and b). The citations in Figure 10 a) represent the same publication even though the publication title and year do not match. The citations in Figure 10 b) also represent the same publication even though the titles differ significantly. It can reasonably be argued that the resolver should classify these citations as different publications since: firstly, different values for the title and year is strong evidence that the publications are different; and, secondly, the same value for the author provides less evidence that the publications are the same since researchers typically publish many papers during their career. The citations in Figure 10 a) provide a clue that they may represent the same publication: the venue is the same for both. This clue alone does not provide overwhelming evidence that the two citations in Figure 10 a) represent the same publication because an author may submit multiple publications to the same academic journal or conference. In sum, some people would classify both pairs of citations in Figure 10 as representing different publications; so it is of little surprise then that the GER framework would also classify both pairs of citations as representing different publications.

The composition of the GER framework is strongly influenced by the feature dependencies entailed in the training set. Figure 11 shows a pair of records from the Restaurant data set that the GER framework incorrectly classified as different restaurants. The only feature that differed in these two records was the restaurant name; all other features were identical in value. It can be argued that both records obviously represent the same restaurant so the GER framework should have resolved them accordingly. Yet following the training phase containing only 112 training instances, the GER framework identified that only the restaurant name and type were needed to resolve pairs of records from the Restaurant data set; with restaurant name given significantly more evidential weight than restaurant type. Stated another way, the GER framework obtained its optimum resolution accuracy for the training data when it matched pairs of records using just the restaurant name and type. Since the two restaurant names in Figure 11 do not match; the GER framework incorrectly classified the two restaurants as non-matching. With more training instances, the GER framework may have learnt that other features, such as the restaurant address and city, are also useful for resolving restaurant instances.

An advantage of the proposed GER framework compared to other approaches is its reduced reliance on a priori knowledge of the data set to determine the optimal model parameters for resolving entities. Köpcke and Rahm (2008) argue there are three key decisions that determine the success of entity resolution: firstly, the selection of features; secondly, the choice of similarity measures; and, thirdly, the selection of similarity threshold values for comparing similarity scores, where the similarity threshold values correspond to the decision boundaries for the SVMs for each feature in the GER framework. If a resolver uses features or metrics that poorly discriminated between matching and non-matching instances its accuracy is likely to decline as a result. If the similarity threshold values are set too high the resolver may classify matching instances as non-matching; if they are set too low the resolver may instead classify non-matching instances as matching. Given the importance of these three key decisions on the resolution accuracy, it is preferable to use machine learning techniques to identify the features, metrics and similarity threshold values that result in optimal resolution accuracy rather than select them using a priori knowledge. Utilizing the
machine-learning approach advocated in this paper enables the GER framework to set the features, metrics and similarity threshold values to optimize its resolution accuracy.

Figure 11: records from the 'Restaurant' data set that the GER framework failed to correctly classify using only the restaurant name and type

The use of Naïve Bayes networks restricts the GER framework’s ability to learn the optimal model for resolving instances. Any model that contains conditional dependencies between individual features is not representable by the GER framework. Stated another way, the use of Naïve Bayes networks to combine the output from individual generic resolvers imposes a restriction on the types of models that the GER framework can evaluate. This restriction is a form of representational bias that defines the states in the GER framework’s search space (Gordon and Desjardins 1995, Mitchell 1997). It follows that if a particular model is not contained in the GER framework’s search space then the framework cannot fit that model to the training data. For example, the GER framework is unable to represent the following probabilistic dependency between publication page numbers and venue: the same publication is unlikely to appear on identical page numbers in 2 different journals or conference proceedings; and publications appearing on different pages in a journal or conference proceedings are also highly unlikely to be the same. It is therefore reasonable to conclude that the GER framework cannot learn the optimal model for training data that has strong conditional dependencies between 2 or more features.

7 Conclusion

The results described in this paper support the claim that the generic pair-wise entity resolution approach can resolve entities from heterogeneous data sources. The GER framework was able to accurately resolve entities from the Restaurant, CORA, Abt-Buy and IMDB-themoviedb.org data sets. These results support the research hypothesis that the generic pair-wise entity resolution approach can enhance cyber situation awareness by learning: firstly, the ‘optimal’ set of features for resolving instances; secondly, the ‘optimal’ set of similarity metrics for comparing feature values; and, thirdly, the similarities that constitute matching entity pairs.

8 Future Work

We plan to evaluate the GER framework using other algorithms for combining the output from individual feature resolvers. It was argued in Section 6 that one can view the use of Naïve Bayes to combine individual feature resolvers as a form of representational bias that restricts the hypotheses that the GER framework can form. To avoid this representational bias, and also to assess the impact of this bias on the GER framework’s efficacy, future work will investigate alternative algorithms for combining individual feature resolvers. We also intend to integrate more comparison metrics into the GER framework. We conjecture that the accuracy of the GER framework will improve with the inclusion of additional comparison metrics. To test this claim we will re-evaluate the GER framework using the Restaurant, CORA, Abt-Buy and IMDB-themoviedb.org data sets. Finally, the GER framework is capable of supporting a hierarchical naïve Bayes network containing multiple Composite resolvers. This allows relationships between entities to be exploited to support entity resolution. We did not utilise this feature of the GER framework in this study, so future work will seek to evaluate the effectiveness of this.

9 References


Mann, H. B. and D. R. Whitney (1947). "On a test of whether one of two random variables is stochastically larger than the other." The annals of mathematical statistics 18(1): 50-60.

McAfee Labs (2013) "McAfee Threats Report: First Quarter ".


A New Extension of Kernel Principal Component Analysis for Finger Vein Authentication

Sepehr Damavandinejadmonfared and Vijay Varadharajan

Advanced Cyber Security Research Centre
Dept. of Computing, Macquarie University
Sydney, Australia
Sepehr.damavandinejadmonfared@mq.edu.au - Vijay.varadharajan@mq.edu.au

Abstract
In this paper, we introduce a new method of data transformation for finger vein recognition system. Our proposed method uses kernel mapping functions to map the data before performing Principal Component Analysis. Kernel Principal Component Analysis (KPCA) is a well-known extension of PCA which is suitable for finding nonlinear patterns as it maps the data nonlinearly. In this work we develop an extension of KPCA which is both faster and more appropriate than PCA for finger vein recognition system. The proposed method is called Feature Dependent Kernel Principal Component Analysis (FDKPCA). In FDKPCA the data is mapped differently from KPCA resulting in lower-dimension feature space where more important and valuable features are selected and extracted. Furthermore, extensive experiments reveal the significance of the proposed method for finger vein recognition systems.

Keywords: Finger vein Recognition, Kernel PCA, and Spectral data transformation.

1 Introduction
Data transformation has been a wide area for researchers as several challenges can be addressed by transferring data into another space where finding genuine patterns and features is desired. There is an extremely large amount of literature on data transformation algorithms and methods. Principal Component Analysis (PCA)(Abdi, Hervé 2010) is one of the well-known methods for dimensionality reduction and feature extraction. PCA(Beng & Rosdi 2011) is a fast method having several usages in multiple areas and application especially in pattern recognition. However, PCA is a linear method which may be inefficient when dealing with nonlinear patters and data. To address the mentioned drawback of PCA, kernel PCA(Kim et al. 2002) was developed which is known as a very well-known and influential extension on PCA. In KPCA, PCA is performed in a kernel feature space which is nonlinearly related to the input data. More specifically, the whole input data space is mapped into another space (kernel space) having higher dimension than the input data dimension. It is enabled using a positive semidefinite (psd) kernel function computing the inner products within the new space (kernel feature space). Therefore, constructing the so-called kernel matrix or the inner product matrix is vital. Then, using the top eigenvalues and their corresponding eigenvectors will lead to kernel PCA data transformation method. Kernel PCA has widespread use in many different areas namely, in machine learning algorithms, data classification, and data de-noising. Such methods have been used in biometrics systems such as face and finger vein recognition. In 2010, R. Jenssen proposed Kernel Entropy Component Analysis(Jenssen 2010) KECA as a new extension on kernel PCA. In 2012(Shekar et al. 2011), KECA was proposed in face recognition system. It is believed that kernel PCA and kernel ECA(Hu & Yang 2010) are more superior methods than PCA as the previous research shows these methods reach more accuracy rate and reliability in terms of data classification and image processing. Considering the way PCA, and KPCA are implemented on images for the purpose of classification and identification, where there are some samples available from each individual to train the system and the remaining samples to test, we propose FDKPCA to improve the performance of the mentioned algorithms. The mentioned methods have been proposed in both face recognition and finger vein recognition systems (Damavandinejadmonfared 2012).

In this paper, we develop a new spectral data transformation method, which can be more stable and faster than KPCA as in FDKPCA the dimension of the feature space is dependent on the dimension of the input data, not the number of input data. It means no matter how many data to analyse, the dimension of kernel matrix (kernel feature space) is fixed. One promising biometric(Delac et al. 2005) is finger vein authentication which has been given considerable attention recently. We have conducted experiments on finger vein(Wu & Liu 2011) database to be able to compare the outcome of the proposed method with KPCA. Experimental results show that not only the proposed method outperforms KPCA in finger vein system, but also it is more time efficient.

The reminder of this paper is organized as follows: Section 2 illustrates some examples of spectral data transformation methods of importance. In section 3, Feature Dependent Kernel Principal Component Analysis (FDKPCA) is introduced. In section 4, Image acquisition and ROI extraction algorithms are explained. A finger vein recognition algorithm is proposed in section 5. Experimental results are presented in section 6. Finally, section 7 concludes the paper.
2 Spectral Data Transformation

In this section, we explain the fundamentals of PCA, and KPCA with examples to comprehend spectral basic data transformation methods.

2.1 Principal Component Analysis (PCA)

A well-known spectral data transformation method is PCA. Let \( X = [x_1, ..., x_N] \) where \( x_i \in \mathbb{R}^d \) and \( t = [1, ..., N] \). As PCA is a linear method, the following transformation is sought assuming \( A \in [d \times d] \) such that

\[
y_t = A'x_t \text{ and } y_t = [y_1, ..., y_N].
\]

\( Y_{pca} = AX \) where \( Y_{pca} = [Y_1, ..., Y_N] \). Therefore, the sample correlation matrix of \( Y_{pca} \) equals to:

\[
\frac{1}{N} Y_{pca} Y_{pca}^T = \frac{1}{N} AX (AX)^T = A \frac{1}{N} XX^T A^T
\]

(1)

The sample correlation matrix of \( X \) is \( \frac{1}{N} XX^T \).

Determining \( A \) such that \( \frac{1}{N} Y_{pca} Y_{pca}^T = I \) is the goal. Considering eigen-decomposition, we will have \( \frac{1}{N} XX^T = V \Delta V^T \), where \( \Delta \) is a diagonal matrix of the eigenvalues \( \delta_1, ..., \delta_d \) in descending order having the corresponding eigenvectors \( v_1, ..., v_d \) as the columns of \( V \).

Substituting into (1), it can be clearly observed that \( A = \Delta^{-1} V^T \) leads to the goal such that \( Y_{pca} = \Delta^{-1} V^T X \).

Performing a dimensionality reduction from \( d \) to \( l \leq d \) is often achieved by the projection of data onto a subspace spanned by the eigenvectors (principal axes) corresponding to the largest top \( l \) eigenvalues. Hence, it is also well-known that \( l \)-dimensional \( Y_{pca} \) preserves the maximum amount of second order statistics in the dimensionality reduced data in comparison with the original \( d \)-dimensional data.

2.2 Kernel Principal Component Analysis (KPCA)

Kernel PCA is a non-linear version of PCA operating in a new feature space called kernel feature space. This space is non-linearly related to the input space. The nonlinear mapping function (kernel function) is given \( \Phi : \mathbb{R}^d \rightarrow F \) such that \( x_i = \Phi(x_i), t = 1, ..., N \) and \( \Phi = [\phi(x_1), ..., \phi(x_N)] \). After performing such mapping in input data, PCA if implemented in \( F \), we need an expression for the projection of \( P_u \Phi \) of \( \Phi \) onto a subspace of feature space principal axes, for example, top \( l \) principals. It can be given by a positive semi-definite kernel function or Mercer kernel, \( k_\sigma : \mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R} \) computes an inner product in the Hilbert space \( F \):

\[
k_\sigma(x_i, x_j) = \langle \phi(x_i), \phi(x_j) \rangle.
\]

(2)

The \((N \times N)\)kernel matrix \( K \) is defined such that element \((i, i')\) of the kernel equals to \( k_\sigma(x_i, x_{i'}) \). Therefore, \( K = \Phi^T \Phi \) is the inner product matrix (Gram matrix) in \( F \). Then, Eigen-decomposing the kernel matrix we have \( K = E \Lambda E^T \) where \( E \) is the eigenvectors \( e_1, ..., e_N \) column wise and their corresponding eigenvalues are in \( D - \lambda_1, ..., \lambda_N \).

Williams in (C.K.I. Williams 2002) discussed that the equivalence between PCA and KPCA holds in KPCA as well (kernel feature space). Hence, we have:

\[
\Phi_{pca} = P_{u\Phi} = D_l^T E_l
\]

(3)

where \( D_l \) is the top large \( l \) eigenvalues of \( K \) and \( E_l \) is their corresponding eigenvectors stored in columns. It means that projecting \( \Phi \) onto spanned feature space (principal axes) is given by \( P_{u\Phi} = \sqrt{\lambda_i} e_i \).

Considering the analogy in (3), \( \Phi_{pca} = D_l^T E_l \) is the solution to the following optimization problem:

\[
\Phi_{pca} = D_l^T E_l^T : \min_{\lambda_1, e_1, ..., \lambda_l, e_l} \| (K - K_{pca}) \|^2 F \cdot 1 \cdot 1.
\]

(4)

where \( K_{pca} = \Phi_{pca}^T \Phi_{pca} = E_l D_l E_l^T \). Therefore, this procedure minimizes the norm of \( K - K_{pca} \).

3 Feature Dependent Kernel Principal Component Analysis (FDKPCA)

Generally, in spectral data transformation methods, finding the most valuable principal axes (appropriate directions in the feature space) is of most importance. In PCA, for example, it is extracted linearly from the principal feature space. In KPCA, however, these axes are extracted from kernel feature space as discussed in previous subsection. We define Feature Dependent Kernel PCA as a \( k \)-dimensional data transformation method obtained by projecting input data onto a subspace spanned by principal kernel axes contributing to the feature dependent kernel space. Feature dependent kernel space is defined as follows:

Let \( X = [x_1, ..., x_N] \), where \( x_i \in \mathbb{R}^d \) and \( t = [1, ..., N] \). The nonlinear mapping function is given \( \Phi : \mathbb{R}^d \rightarrow \mathbb{R}^d \) such that \( x'_i = \Phi(x'_i), t = 1, ..., d \) where \( x'_i \) is an \( N \) dimensional vector including all of the \( t_{th} \) features from \( N \) input data. Explaining this, we have \( \Phi = [\phi(x'_1), ..., \phi(x'_N)] \). The use of a positive semi-definite
kernel function or Mercer kernel computes an inner product in the new space $F^d$:

$$k_d(x_i', x_j') = \langle \phi(x_i'), \phi(x_j') \rangle.$$  \hspace{1cm} (5)

The $(N \times N)$kernel matrix- we define that as $K_{FDKPCA}$ is now defined such that element $(i, j)$ of the kernel matrix is $k_d(x_i', x_j')$. Therefore, $K_{FDKPCA}$ is the Gram matrix or the inner product matrix in $F^d$. The next stage in FDKPCA is to perform PCA on $K_{FDKPCA}$. Note that the kernel matrix given in FDKPCA feature space ($K_{FDKPCA}$) is totally different from that of KPCA. Firstly, we explain KPCA feature space for the sake of clarity and then, FDKPCA feature space is introduced.

Figure 1 illustrates a brief flow diagram of reaching kernel feature space from the scratch. As it is shown in Figure 1, $N$ input data are first mapped into kernel space by $\phi$ and then the Gram matrix (kernel matrix) is calculated using inner product. Note that the dimension of kernel matrix is equal to the number of input data- $N$. Eigen-decomposition is the next step where all eigenvalues and their corresponding eigenvectors are extracted and reordered in a descending manner from the greatest to the smallest value. After finding the kernel axes in this space, the kernel matrix, which represents the input data, is projected onto the kernel feature vectors (eigenvectors). The drawback to PCA is that the dimension of feature space and kernel matrix could become too high and as a result data transformation could be computationally expensive. In addition, finding the most optimized sub-space in kernel feature space could be challenging and sometimes inefficient.

In FDKPCA feature space, the input data is projected onto a subspace spanned by principal kernel axes contributing to the feature dependent kernel space. In FDKPCA all features having the same dimension from all input data are firstly considered in separate vectors, and then mapped into kernel space which is called FDKPCA feature space. Finally, the kernel matrix (Gram matrix) using inner products which is a $d$-dimensional space is computed. Note that the input data has the dimension of $d$ which means there is no growth of dimension while computing FDKPCA feature space. Having $d$-dimensional FDKPCA feature space, the eigenvectors and their corresponding eigenvalues are decomposed in this step. The original input data is projected onto a subspace of FDKPCA feature vectors for the purpose of transformation and dimensionality reduction. In the FDKPCA feature space, the non-linear relations between data inputs are extracted in a feature wise manner which results in having both more efficiency and higher speed.

### 4 Image Acquisition and Region of Interest (ROI) Extraction Algorithm

Based on the proven scientific fact that the light rays can be absorbed by deoxygenated hemoglobin in the vein, absorption coefficient (AC) of the vein is higher than other parts of finger. In order to provide the finger vein images, four low cost prototype devices are needed such as an infrared LED and its control circuit with wavelength 830nm, a camera to capture the images, a micro-computer unit (MCU) to control the LED array, and a computer to process the images. The web-cam has an IR blocking filter; hence, it is not sensitive to the infrared (IR) rays. To solve this problem an IR blocking filter is used to prevent the infrared rays from being blocked.

Three major steps are used to crop images optimally: first one is detecting the edge. In order to perform the cropping part, two horizontal lines are determined by finding the horizontal edges in original images. Two conditions should be satisfied to find the appropriate lines by edge detection algorithm: (1) the pairs of detected points should be located between 35% and 65% of the height of the captured image, and (2) among the detected pairs, the pair of the edge that are the widest will be chosen. Last step is to crop the images from 5% from right border and 5% percent from left border vertically. An example is shown in Figure 2 a) and b).
5 Proposed Finger Vein Recognition Algorithm

The flow diagram of the proposed finger vein recognition algorithm is shown in Figure 3. First step is to extract the region of interest from the samples which was explained in section 4. After extraction the region of interest, the proposed FDKPCA is conducted on the data to find the optimal axes (eigenvectors) to project the data onto. Actually, based on the dimension of the input data, the number of experiments is assigned. It is because of the nature of PCA based algorithm as there are as many different dimensions as the dimension of the input data. for instance, if the dimension of the input images is 100, there are 100 different implementations on the same data using 100 different feature vectors to reduce the dimension and extract the features. And finally, Nearest Neighbor classifier is taken into account to classify the extracted features and make the final decision. Figure 3. Indicated the flow diagram of the clustering algorithm.

6 Experimental Results

In this section, the experiments are conducted to corroborate the performance of Feature Dependent Kernel Principal Component Analysis (FDKPCA) over Kernel Principal Component Analysis (KPCA). We used Gaussian kernel function as the mapping function in both KPCA and FDKPCA. Finger vein database used in the experiments consists of 500 images from 50 individuals; 10 samples from each subject were taken. In this experiment 4, 5, and 6 randomly selected samples are used to train and the remaining 6, 5, and 4 samples are used to test respectively. In each experiment, the accuracy is calculated using the first 200 components of the extracted features meaning that each experiment is repeated 200 times using the first 200 features to project the data onto, and also the dimension is reduced from 100% to 0% in different experiments. The results are shown in Figure 4, 5, and 6.

As it was expected, the more the number of training samples gets, the higher accuracy rate goes. It is observed that no matter how many samples to train and test and no matter how high the dimension of the feature vector is, using FDKPCA results in a higher accuracy rate in all experiments and all different feature vectors. The discrepancy between the obtained accuracies is very dramatic in the first half of the graphs.

It can be explained by the nature of the FDKPCA which is able to find more valuable feature vectors in the
lower dimension than KPCA. For example, FDKPCA reaches the accuracy of around 98% to 100% using less than 50 vectors to transfer the data while KPCA gets its highest accuracy which is around 90% to 93% using the feature vectors with the dimension of more than 120.

Experimental results reveal that not only is the proposed system (FDKPCA) more superior than KPCA for finger vein recognition, but also it can be considered much faster than KPCA as it reaches its peak in much lower dimension than KPCA.

7 Conclusion

In this research, we proposed a new method of dimensionality reduction and feature extraction (FDKPCA) which is a combination of the well-known Principal Component Analysis (PCA) and Kernel Principal Component Analysis (KPCA), which was proven to be faster and more accurate than PCA and KPCA in terms of finger vein recognition. We also proposed a new finger vein recognition algorithm using the FDKPCA method to extract the most valuable features from the samples and reduce the dimension of the data. In the proposed system, the images are automatically cropped first and then FDKPCA is performed for the purpose of feature extraction. Finally, Nearest Neighbour classifier is conducted to classify the extracted feature and make the final decision. Extensive experiments on our finger vein data reveal the significance of the proposed method in comparison with the traditionally used methods.

8 References


Dynamic Topic Detection Model by Fusing Sentiment Polarity

Xi Ding\textsuperscript{1}, Lanshan Zhang\textsuperscript{2}, Ye Tian\textsuperscript{1}, Xiangyang Gong\textsuperscript{1} and Wendong Wang\textsuperscript{1}
\textsuperscript{1}State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing, 100876, China
\textsuperscript{2}School of Digital Media and Design Arts, Beijing University of Posts and Telecommunications, Beijing, 100876, China

Email: dingxi515@163.com, zls326@sina.com, \{yetian, xygong, wdwang\}@bupt.edu.cn

Abstract

Traditional static topic models mainly focus on the statistical correlation between words, but ignore the sentiment tendency and the temporal properties which may have great effects on topic detection results. This paper proposed an LDA-based dynamic sentiment-topic (DST) model, which could not only detect and track topics but could also analyse the shift of general's sentiment tendency towards certain topic. This model combines the data with the sentiment and dynamic properties of time by maximum likelihood estimation and the sliding window. We use Gibbs sampling method to estimate and update model parameters, and use random EM algorithm for model reasoning. Experiments on real dataset demonstrate that DST model outperforms the existing algorithms.

Keywords: dynamic sentiment topic model, sentiment analysis, Gibbs sampling.

1 Introduction

With the rapid development of Web 2.0 technology, social media, represented by Facebook, Twitter, Blog and Weibo, has really taken off in the last few years. More and more people choose social media as main information exchange platform to publish and access real-time information. In addition to daily communication functions, the huge amount of user generated content (UGC) often also carries people’s attitudes towards certain public events or products/services. Existing researches show that compared with traditional media channels, social media data which conveys public sentiment has more important social and economic value for community management agency, enterprise and the general public. Affected by this, the sentiment analysis and topic detection of unstructured social media data has emerged as a hot research hotspot among the various tasks of social networking analysis area.

Researchers who are concentrating on the area of sentiment analysis mainly focus on studying the sentiment polarity (positive, neutral, negative) classification methods of the social media data. Although a host of research achievements have been disclosed [Pang, B. and Lee. L][Aue, A. and Gamon, M][Turney, P. D], they are mostly based on supervised learning methods, and there still existed two limitations. First, a lot of manually annotated samples are needed for parameters adjustment process. Second, sentiment classifier trained from certain topic area often does not apply to other topic areas since the sentiment distribution and topic content are closely related. Besides, an unavoidable problem is that the sentiment word dictionary that was trained from traditional corpus could not be applied to social data that are flexibly expressed with much more emotion icons and disjunctive questions used, considering the huge difference of language style between social media and traditional media. Topic detection derives from TDT (Topic Detection and Tracking) technology, which mainly focuses on detecting and organizing unknown topics from traditional formal expressed text stream. Topic detection task consists of two branches: historical topic detection and online topic detection. For the former, its objective is to dig out the hidden topics from a given corpus with unsupervised clustering means. Each cluster corresponds to a certain topic. While online topic detection determines whether newly arrived text stream belongs to an existing topic or a new topic according to historic information. Compared with newspaper, periodicals and academic report, social media are most non-standard expressed unstructured short texts in real-time formalism. These features make the task of topic detection which takes social media as study object more challenging.

Sentiment analysis and topic detection are taken as two independent research tasks in current social networking analysis fields. However, “sentiment” and “topic” are two highly associated concepts. On one hand, the generation and spread process of sentiment must rely on a certain body, i.e., a specified topic. On the other hand, the change of sentiment would react on its carrier, i.e., certain specified topic, and consequently affect the evolution of topic. Take the event “the loss of communication of flight M370” as example. At the beginning stage, anxiety and trepidation are circulating widely among the people who express concern about this event. As time goes on, the mood gradually evolves into sadness and discontent. Under the influence of this mood, M370 event evolves into a new stage of “relatives’ doubt about Malaysian Airline’s emergency treatment” from “the lost of aircraft” and “the search and rescue”. Thus it could be seen, the probability distribution of topic is affected by sentiment, and these two concepts are highly relevant.

By fusing the polarity of sentiment, a dynamic sentiment-topic (DST) model is proposed based on Latent Dirichlet Allocation (LDA) in this paper. LDA model assumes that, a document is composed of various topics with different probabilistic combination, within which each topic itself is also a probability distribution of a series of words. For a given corpus, LDA detects all hidden
topics by computing the posterior probability of each word that it belongs to a certain topic. LDA model gains tremendous success in topic analysis area. It is widely applied to social networking analysis areas [Lacoste, S, Sha, F and Jordan, M,] [Ramage, D., Hall, D., Nallapati, R., and Manning, C.][Wallach, H., Mimno, D., McCallum, A.]. LDA model divides the document generation process into two stages: “document-topic” generative process and “topic-word” generative process. However, in “document-topic-word” three layer architecture, “topic” is affected by the polarity of “sentiment”. The influence is mainly embodied in three aspects. Firstly, “document-topic” probability distribution would change as time passes. The probability distribution of topics within a document at time slot $t$ is different with that at time slot $t-1$, however the probability distribution at time slot $t$ depends on the probability distribution at time slot $t-1$. Secondly, the probability distribution of the sentiment polarity would change as time passes. The probability distribution of the sentiment polarity at time slot $t$ depends on the probability distribution at time slot $t-1$. Thirdly, “topic-word” probability distribution would change as time passes. The probability distribution of word within a topic at time slot $t$ is different with that at time slot $t-1$. However the probability distribution at time slot $t$ depends on the probability distribution at time slot $t-1$. Based on the above considerations, the DST model proposed in this paper takes the influence that sentiment polarity exerts on the probability distribution of topic and the probability distribution of word into account. Also, DST model adjusts the probability distributions dynamically according to the dependencies in temporal dimension. Thus, for the large amount of social media data, this model could not only detect the hidden topics, but can also classify the topics according to their sentiment polarity. Furthermore, the evolution process of topic and its corresponding sentiment would be deduced.

The main contribution of this paper embodies in the following aspects. Firstly, by fusing sentiment polarity into the topic modelling process, DST model could properly reflect the impact that sentiment exerts on topic detection. Secondly, DST model enhances LDA model by introducing the dynamic characteristics, such that it is more appropriate for those social media data with strong real-time characteristics. Thirdly, experiments on a large real dataset demonstrate that DST model is superior to existing algorithms.

The rest part of this paper is organized as follow. Section 2 gives a brief introduction of the related work. In Section 3, we describe the proposed dynamic sentiment-topic model in detail. In the next section, we conduct an empirical analysis of DST model on a large real dataset. Section 5 concludes the whole paper and discusses the future work.

2 Related Work

Vast majority of research work has been conducted on topic analysis fields. Traditional topic detection method, such as TDT, aims to segment text stream into different news articles. TDT deals principally with formal text stream. Since the huge gap in language style, expressing way between formal text and social media text, TDT cannot work well. Statistical learning theory is the basis of most traditional topic detection algorithms. Latent Semantic Indexing (LSI) [Hofmann, T] is a typical statistical learning based topic model, within which Single Value Decomposition (SVD) method is utilized to decompose the document-word matrix, and the subspace with the largest degree of distinction in TF-IDF (Term Frequency-Inverse Document Frequency) feature space then corresponds to the hidden topic. PLSA (Probabilistic Latent Semantic Analysis) which is also called Aspect Model derives from LSI. PLSA enhanced LSI from linear algebra level to probability space. Though PLSA is significant for topic modelling, the disadvantages are also obvious. The number of parameters in PLSA model grows linearly with the size of corpus, such that it may cause over-fitting problem. LDA (Latent Dirichlet Allocation) expands PLSA by introducing Dirichlet as prior distribution. LDA greatly reduces the number of required parameters, meanwhile the over-fitting problem get well solved. However, multinomial distribution based topic model (LDA) still face the over-fitting problem when it comes to short texts since the sparse vocabulary.

In recent years, plenty of improved topic models are proposed based on LDA. DTM (Dynamic Topic Model) [David, M., John, D.] models the natural parameter space with state-space model. It constructs a LDA-like topic model for corpus within each time slot, and these topic models locate in a common markov chain. That means dependency exists between the adjacent topic models, and the topic model within time slot $t$ evolves from that within time slot $t-1$. Unlike DTM, TOT [Xuera, W., McCallum, A.] (Topic over Time) model could depict the long term evolution of a certain topic, and it can also predict the topic distribution within a given time slot. However, this model only takes the relationship between word and time into account, but ignores the influence that sentiment polarity imposes on topic modelling. A newly proposed JST (Joint Sentiment-Topic) model [He, Y. and Lin, C.] is weakly supervised. This model integrates sentiment factor into topic modelling process, and it assumes all words in a document embody sentiment tendency, so each topic also has its own sentiment attributes. Strong sentiment would influence people’s concern about the topic, and consequently change the topic distribution and word distribution. Based on this assumption, JST model could not only detect the topic hidden in text streams, but could also classify the topics according to its corresponding sentiment polarity. However, the evolution of sentiment polarity of a specified topic is not taken into account in JST model. This limitation leads to the inadaptability when deal with social media data which are timeliness and expressed with rich affections.

Seldom researches consider dynamic characteristics of topics and the influence of sentiment polarity simultaneously in topic modelling process. The dJST (dynamic joint sentiment-topic) model [He, Y., Lin, C., Gao, W., and Wong, K.-F] is most close to our work. It derives from LDA model, but combines both dynamic temporal characteristic of topic and author’s sentiment tendency in their model. dJST model estimates the hyper parameters of topic distribution and word distribution. The drawback is that dJST overlooks the fact that the sentiment polarity probability distribution in a document is not
constant, but always changing according to the evolution process of topic.

3 DST Model

3.1 Overview

In this section we would detail the DST Model. It is a generative model, and belongs to a form of Bayesian probability model which is a predictive model that learns by the joint probability density distribution of the data, and then finds the conditional probability distribution. The model uses reasonable mathematical methods, such as maximum likelihood estimation and sliding window that makes the sentimental with the dynamic characteristics and time properties reasonably integrate into the existing traditional static text analysis model—LDA model, so we can get the public’s sentimental tendencies of certain topic which is occurred in a period of time and the changes of the sentimental tendencies of certain topic in the trend. Because of the actual situation, the text file generated by social network often goes with the author's sentiment and sentimental tendencies vary among people which go with dynamic changes over time. The framework of our model has five layers where the document layer contains many types of sentimental tendencies, and topic layer is associated with sentiment and sentimental tendencies vary among people which go with dynamic changes over time. These topics all obey the topic distribution. Sentimental tendencies layer is the connecting bridge and plays an important role in the five layers. A graphical model of DST model is represented in Figure 1.

![Figure 1: DST MODEL](image)

Model assumes that at each epoch t, each of text documents set contains sentimental tendencies, and all the sentimental tendencies is divided into L Classes which express as \( L = \{ l_1, l_2, ..., l_L \} \). The object processed by the model is text document collections, which are received with their order of publication timestamps preserved and are represented as \( D_t = \{ d_{1,t}, d_{2,t}, ..., d_{n,t} \} \) and the text document collections’ size is \( D_t \). Appearing in the text document collections at epoch t, all words constitute a vector of word token whose size is \( V_t \), and the word token can be expressed as \( V_t = \{ W_{t_1}, W_{t_2}, ..., W_{t_n} \} \). Our notations are summarized in Table 1.

Model defines that for each epoch \( t \), a document \( d_i \) contains \( L \) classes of sentimental tendencies, and the sentimental tendency obeys the sentiment distribution \( \pi'_i \). Each sentimental tendencies \( l \) can contain \( Z \) topics, and \( \pi'_i \) obeys the gamma distribution, \( \pi'_i \sim \text{gamma}(\alpha'_i, \beta'_i) \); at current epoch \( t \), the topic distribution corresponding to the document \( d \) and the sentimental tendencies \( l \) is affected by the topic in the past (the parameters \( \alpha'_{l,z} \) with the corresponding Dirichlet distribution \( \theta'_{l,z} \) obeys gamma distribution, \( \alpha'_{l,z} \sim \text{gamma}(\mu_{l,z}^{-1}, \mu_{l,z}) \); same time the current sentiment-topic specific word distributions \( \phi'_{l,z} \) at epoch \( t \) are generated according to the word distributions at previous epochs (the word distributions at previous epochs is reflected in two parameters introduced \( \delta'_{i,z,s} \), \( \sigma'_{i,z,s} \). These

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>( D_t )</td>
<td>number of documents in epoch ( t )</td>
</tr>
<tr>
<td>( L )</td>
<td>number of sentiment labels</td>
</tr>
<tr>
<td>( Z )</td>
<td>number of topics</td>
</tr>
<tr>
<td>( V_t )</td>
<td>number of unique words in the current epoch ( t )</td>
</tr>
<tr>
<td>( S )</td>
<td>number of time slices</td>
</tr>
<tr>
<td>( \gamma' )</td>
<td>matrix of ( D_t \times L ) dimension, row ( d ) represents the priors of the mixing proportion of sentiment in document ( d_i )</td>
</tr>
<tr>
<td>( \alpha'_i )</td>
<td>matrix of ( L \times Z ) dimension, row ( l ) represents the priors of the mixing proportion of topics in sentiment label ( l )</td>
</tr>
<tr>
<td>( \beta'_i )</td>
<td>matrix of ( L \times Z \times V_t ) dimension, priors for the word distribution conditioned on sentiment labels and topics</td>
</tr>
<tr>
<td>( \lambda'_i )</td>
<td>matrix of ( L \times V_t ) dimension which encodes the word prior sentiment polarity information</td>
</tr>
<tr>
<td>( \sigma'_{i,z,s} )</td>
<td>parameter notation for the sentiment label mixture proportion for document ( d_i )</td>
</tr>
<tr>
<td>( \phi'_{l,z} )</td>
<td>multinomial distribution over words for the ( l )th sentiment label for document ( d_i )</td>
</tr>
<tr>
<td>( \delta'_{i,z,s} )</td>
<td>multinomial word distribution of sentiment label ( l ) and topic ( z ) with time slice ( s ) at epoch ( t ), ( \delta'<em>{i,z,s} = { \delta'</em>{i,z,s,w} }_{w=1}^{V_t} )</td>
</tr>
<tr>
<td>( \sigma'_{i,z,s} )</td>
<td>weight vector, ( \sigma'<em>{i,z,s} = { \sigma'</em>{i,z,s,w} }<em>{w=1}^{V_t} ), each of which determines the contribution of time slice ( s ) in computing the priors of ( \phi'</em>{l,z} )</td>
</tr>
</tbody>
</table>

Table 1: Notations used in the paper
two parameters take different values at different time scales. So our stipulate
\[ \beta_{t} = \Sigma k_{t} \sigma_{t} \]  

(1)

Document generation process is mainly determined by the topic and emotional tendencies mark. Formal definition of document process in the model shown in Figure 1 is as follows:

(1) For each sentiment label \( l = 1, \ldots, L \)
For each topic \( z = 1, \ldots, Z \)
At the scale \( S \), Compute \( \beta_{t} = \Sigma k_{t} \sigma_{t} \)
Select the matrix of \( L \times Z \) word distribution \( \phi_{t} \sim \text{Dir}(\beta_{t}) \);
(2) For each document \( d = 1, \ldots, D_{t} \)
According to \( \gamma' \sim \text{gamma}(\zeta_{t}^{-1}, \zeta) \), compute \( \gamma' \), Choose a sentiment distribution \( \pi_{l,t} \sim \text{Dir}(\gamma'_{t}) \);
For each sentiment label \( l \) under document \( d \), according to \( \alpha_{l} \sim \text{gamma}(\mu_{l}, \nu_{l}) \), compute \( \alpha_{l} \), choose a topic distribution \( \theta_{l,t} \sim \text{Dir}(\alpha_{l}) \);
For each word \( W = 1, \ldots, V \) in document \( d \)
Choose a sentiment label \( l_{w} \sim \text{Mult}(\pi) \);
Choose a topic \( z_{w} \sim \text{Mult}(\theta_{l,t}) \);
Choose a word \( w_{l} \sim \text{Mult}(\phi_{l,t}) \);
(3) Repeat the above process can get the whole document

**Figure 2**: Document generation process

### 3.2 Inference Learning

We present a stochastic EM algorithm to sequentially update model parameters at each epoch using the newly obtained document set and the derived evolutionary parameters. At each iteration step, we use the collapsed Gibbs sampling to estimate potential distribution of sentiment label \( l \) and topics \( z \), and use the maximum likelihood estimation method to estimate the hyper parameters.

The total probability of the model for the document set \( V_{t} \) at epoch \( t \) given the evolutionary parameters \( \delta_{t} \), \( \sigma_{t} \), and the previous model parameter is:
\[ p(w' | \gamma', \gamma, \phi, \theta, \pi) = p(l' | \gamma') p(w' | l', \gamma', \phi, \theta, \pi) p(l' | \gamma') p(w' | l', \gamma', \phi, \theta, \pi) \]  

(2)

For the first term on the right-hand side of Equation (2), by integrating out \( \pi \), we obtain:
\[ p(l' | \gamma') = \frac{\prod_{k} \Gamma(\sum_{j} k_{t,j,k}) \prod_{k} \Gamma(N_{d,k,j} + \gamma'_{k})}{\prod_{k} \Gamma(N_{d,k,j} + \gamma'_{k})} \]  

(3)

That is equal to \( p(l' | \gamma') = \frac{\prod_{k} \Delta N_{d,k,j}}{\prod_{k} \Delta N_{d,k,j}} \) . Where \( D_{t} \) is the total number of documents in epoch \( t \), \( N_{d,j} \) is the number of times sentiment label \( l \) being assigned to some word tokens in document \( d \) at epoch \( t \), \( N_{d,j} = \sum_{i} N_{d,i,j} \) . It is a constant which can be derived from statistical data input and \( \Gamma \) is the gamma function.

For the second term by integrating out \( \theta \), we obtain:
\[ p(z' | \alpha_{l}, \gamma_{t}') = \prod_{d} \prod_{k} \Gamma(\sum_{j} k_{t,j,k}) \prod_{k} \Gamma(N_{d,k,j} + \alpha'_{l}) \]  

(4)

That is equal to \( p(z' | \alpha_{l}, \gamma_{t}') = \frac{\prod_{d} \Delta N_{d,k,j}}{\prod_{d} \Delta N_{d,k,j}} \) . Where \( N_{d,k,j} \) is the number of times a word from document \( d \) being associated with topic \( z \) and sentiment label \( l \) at epoch \( t \). It is a constant which can be derived from statistical data input and \( N_{d,k,j} = \sum_{i} N_{d,k,i,j} \) .

For the last term, by integrating out \( \phi \), we obtain:
\[ p(w' | l', \gamma_{t}') = \prod_{d} \prod_{j} \Gamma(\sum_{k} k_{t,j,k}) \prod_{k} \Gamma(N_{d,k,j} + \alpha'_{l}) \]  

(5)

That is equal to \( p(w' | l', \gamma_{t}') = \frac{\prod_{d} \Delta N_{d,k,j}}{\prod_{d} \Delta N_{d,k,j}} \) . Where \( N_{d,k,j} \) is the number of times word \( w \) appeared in topic \( z \) and with sentiment label \( l \) at epoch \( t \). It is a constant which can be derived from statistical data input and \( N_{d,k,j} = \sum_{i} N_{d,k,i,j} \) .

Gibbs sampling is the use of Markov Chain Monte Carlo method. According to the prior distribution of model parameters and statistical data which is given on the pre-S slots, it sequentially samples each variable of interest, sentiment label \( l \), and topic \( z \) through the case that it must meet the detailed balance condition. To meet this condition, let the index \( x = (d,n,t) \) and the subscript \( x \) denote a quantity that excludes counts in word position \( n \) of document \( d \) in epoch \( t \), the conditional posterior for \( z_{t} \) and \( l_{t} \) by marginalizing out the random variables \( \phi, \theta, \pi \), and \( x \) is:
\[ p(x_{t} = j, l_{t} = k | \theta_{t}^{T}, \gamma_{t}', \phi_{t}^{T}, \alpha_{t}^{T}, \sigma_{t}^{T}) \propto \sum_{z_{t}} N_{d,j,z} + \alpha'_{l} \sum_{z_{t}} \alpha'_{l} \sum_{j} \sum_{k} \alpha'_{l} \]  

(6)

There are two types of parameters to be estimation which are evolutionary parameters and hyper parameters.

#### A. The evolutionary parameters estimation

We use the fixed-point iteration method to estimate the weight vector \( \delta_{t} \), directly from data by maximizing the joint distribution in Equation (2). The update formula is:
\[ (\delta_{t,j}^{d})_{m+1} = \frac{\delta_{t,j}^{d} \sum_{i} N_{d,k,i,j} A}{B} \]  

(7)

Where,
\[ A = \phi(N_{d,k,j} + \sum_{i} \delta_{i,j,k}^{d} \sigma_{i,j,k}) \]  

(8)

\[ B = \phi(N_{d,k,j} + \sum_{i} \delta_{i,j,k}^{d}) - \phi(\sum_{i} \delta_{i,j,k}^{d}) \]  

(9)
And $\phi(.)$ is the digamma function defined by

$$\phi(x) = \frac{d \log \Gamma(x)}{dx}. $$

The evolutionary parameter $\sigma'_{d,k,j,u}$ accounts for the historical word distributions at different time slices. So we estimate $[\sigma'_{d,k,j,u}]$ in topic $j$ and sentiment label $k$ at time slice $s$, which can be calculated as follows:

$$\sigma'_{d,k,j,u} = \frac{C_{d,k,j,u}}{\sum_s C_{d,k,j,u}} \tag{10}$$

Where $C_{d,k,j,u}$ is the expected number of times word $w$ is assigned to sentiment label $k$ and topic $j$ at time slice $s$. At different time scales, its value is different. Thus $C_{d,k,j,u}$ can be obtained directly from the count $N_{d,k,j,u}$, that is $C_{d,k,j,u} = \frac{N_{d,k,j,u}}{\sum_s N_{d,k,j,u}}$, the expected number of times word $w$ is associated with sentiment label $l$ and topic $z$ at epoch $t$, which can be calculated by:

$$\hat{N}_{d,k,j,u} = N_{d,k,j,u} + \sum_z \delta_{l,z} \sigma_{d,k,j,u}$$

Where $N_{d,k,j,u}$ is the observed count for the number of times word $w$ is associated with sentiment label $l$ and topic $z$ at epoch $t$.

**B. Hyper parameters ($\alpha'$, $\gamma'$) estimation**

We estimate $\alpha'$ and $\gamma'$ from data using maximum-likelihood as part of the online stochastic EM algorithm.

A common practice for the implementations of topic models is to use symmetric Dirichlet hyper parameters. However, it has been found that an asymmetric Dirichlet prior over the per-document topic proportions has substantial advantages over a symmetric prior. So when first entering a new epoch, we initialize the asymmetric $\alpha' = 0.01$, $\gamma' = 0.01$. Afterwards according that $\gamma'$ obeys gamma distribution $\gamma' \sim \text{gamma}(\xi'_{\gamma}, \xi)$ which lead to the probability function of $\gamma'$ is

$$p(\gamma' | \xi_{\gamma}') = \frac{\gamma'^{\xi_{\gamma}-1} \exp(-\xi_{\gamma}' \gamma')}{\Gamma(\xi_{\gamma}')}$$

and $\alpha'$ obeys gamma distribution $\alpha' \sim \text{gamma}(\mu_{\alpha'}, \mu)$ which lead to the probability function of $\gamma'$ is

$$p(\alpha' | \xi_{\mu}) = \frac{\alpha'^{\mu_{\alpha'}-1} \exp(-\mu \alpha')}{\Gamma(\mu_{\alpha'})}$$

For every 40 Gibbs sampling iterations, $\alpha'$ and $\gamma'$ are learned directly from data using maximum-likelihood estimation.

$$\left(\alpha'_{\gamma}ight)_{\text{new}} \leftarrow \frac{\mu \alpha'_{\gamma} + 1 + \gamma' \sum_j (\phi(N'_{d,j} + \alpha'_{\gamma} ) - \phi(\sum_k \alpha'_{\gamma}))}{\mu + \sum_j (\phi(N'_{d,j} + \sum_k \alpha'_{\gamma} ) - \phi(\sum_k \alpha'_{\gamma}))}$$

The detail description of the online stochastic EM algorithm for the DST model is given in Algorithm 1.

**Algorithm:** Gibbs sampling procedure for DST Model

**Input:** Number of topics $Z$, number of sentiment labels $L$, number of time slices $S$, word prior polarity transformation matrix $\lambda$, epoch $t \in \{1, 2, \ldots, \text{maxEpochs}\}$, a stream of documents $D_t = \{d_{t,1}, d_{t,2}, \ldots, d_{t,S}\}$

**Output:** DST model;

Sort documents according to their time stamps;

FOR $t = 1$ to maxEpochs do

IF $t == 1$ THEN

Set $\beta'_{l,z} = \lambda \times 0.01$;

ELSE $\sigma'_{l,z}$, $\alpha'_{l,z}$

Set $\delta_{l,z} = \sigma'_{l,z}$;

Set $\sigma'_{l,z} = 1/S$;

Set $\beta'_{l,z} = \Sigma \delta_{l,z} \sigma'_{l,z}$;

END

Set $\gamma' = (0.05 \times \text{Average document length})/L$;

Set $\alpha' = (0.05 \times \text{Average document length})(L \times Z)$;

Initialize $\pi'$, $\theta'$, $\phi'$, and all count variables;

Initialize sentiment label and topic assignment randomly for all word tokens in $D_t$;

FOR $i = 1$ to max Gibbs Sampling iterations DO

Update $\pi'$ using Equation (12);

Update $\gamma'$ using Equation (13);

Update $\delta_{l,z}$ using Equation (7);

Set $\beta'_{l,z} = \Sigma \delta_{l,z} \sigma'_{l,z}$;

END

FOR every 200 Gibbs sampling iterations DO

Update $\pi'$, $\theta'$, $\phi'$ with the new sampling results;

END

END

Update $\beta'_{l,z}$, using Equation (10);

END

**Figure 3:** Online Stochastic EM Algorithm

$$\left(\gamma'_{\text{new}}\right) \leftarrow \frac{\xi_{\gamma}' - 1 + \gamma' \sum_j (\phi(N'_{d,j} + \gamma'_{\text{new}}) - \phi(\gamma'_{\text{new}}))}{\xi' + \sum_j (\phi(N'_{d,j} + \gamma'_{\text{new}}) - \phi(\gamma'_{\text{new}}))}$$

The detail description of the online stochastic EM algorithm for the DST model is given in Algorithm 1.

**4 Experiment**

To evaluate the proposed solution, we conducted experiments using a real dataset collected from Sina Weibo.

**4.1 Dataset Preparation**

We implement a Sina Weibo crawler using the APIs provided by Sina Corp. (http://open.weibo.com/) to collect the experiment dataset.

From January 23, 2013 to February 23, 2013, the data crawling process continued for 31 days. Table 1 illustrates the statistics of the dataset. Considered that many public
events happened in the period of December 19, 2011 to December 26, 2011, we select the Weibo data generated in that time period as our test dataset.

### Table 2: Statistics of Collected Dataset

<table>
<thead>
<tr>
<th>Statistical Measure</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Users</td>
<td>531,935</td>
</tr>
<tr>
<td>Number of Posts</td>
<td>25,838,961</td>
</tr>
<tr>
<td>Number of Comments</td>
<td>14,498,416</td>
</tr>
</tbody>
</table>

4.2 DST Model V.S Existing Models

In order to evaluate the effectiveness of our proposed model in topic detection, we compared the performance of DST with LDA and DST with metric of perplexity.

Perplexity is a metric which was mostly used in language modelling. It could measure a language model's prediction ability when applied on an uncharted dataset. Perplexity is defined as the reciprocal geometric mean of the likelihood of a test corpus given a trained model's Markov Chain state $\mathcal{M}$. In this paper, we adopted the definition of perplexity as below [He, Y., Lin, C., Gao, W., and Wong, K.-F.]:

$$\text{Perplexity } P(D_t | M) = \exp \left( \frac{\sum_{\ell=1}^{N_t} \log p(v_{\ell,t} | M)}{\sum_{\ell=1}^{N_t} w_{\ell,t}} \right)$$

Here, this metric means the per-word predictive perplexity of the unseen test set $D_t$ at each epoch $t$ based on the previously trained model $M$.

In the experiments we studied the influence of the topic number settings on the DST model performance. With the number of time slices fixed at $S = 4$, we vary the topic number $T \in \{1, 5, 10, 15, 20\}$.

Figure 4 shows the average per-word perplexity over epochs with different number of topics. DST has lower perplexities than all the other models with the increased number of topics.

![Figure 4: Perplexity VS. number of topics](image)

The average perplexity for each epoch with the number of time slices set to 4 and the number of topics set to 15 for the DST models is shown in Figure 5. In addition, we also plot the perplexity results of LDA and DTM. LDA only uses the data in the previous epoch for training and hence it does not model dynamics while DTM uses all past data for model learning. We set the number of topics to 15 for both DTM and DST. For LDA, the number of topics was set to 3 corresponding to positive, negative, and neutral sentiment labels.

![Figure 5: Perplexity vs. number of epochs](image)

Figure 5 shows that LDA has the highest perplexity values followed by DTM and DST. The perplexity gap between DTM and the DST models increases with the increasing number of epochs. The variants of DST models have quite similar perplexities.

5 Conclusion

In this paper, we proposed a dynamic Sentiment Topic (DST) model, which could models the topics and its corresponding sentiment polarities from huge amount of user generated content data. By taking the sentiment effects into account, DST could properly reflect the impact that sentiment exerts on topic detection. Also, it enhances LDA model by introducing the dynamic characteristics, such that it is more appropriate for those social media data with strong real-time characteristics. Experiments results on a large real dataset show that compared with existing models, DST performances better in terms of perplexity. In future work, we may work on reducing the temporal and spatial overhead of this model.

Acknowledgements

This research was partially supported by National Natural Science Foundation of China under Grant No. 61402045, Specialized Research Fund for the Doctoral Program of Higher Education under Grant No. 20130005110011, and Fundamental Research Funds for the Central Universities under Grant No. 2014RC1301.

References


A Macro-Level Model for Investigating the Effect of Directional Bias on Network Coverage

Graeme Smith1 J.W. Sanders2,3 Qin Li1,*

1School of Information Technology and Electrical Engineering
The University of Queensland, Australia
Email: smith@itee.uq.edu.au, qli@sei.ecnu.edu.cn
2African Institute for Mathematical Science, South Africa
3Department of Mathematical Sciences, Stellenbosch University, South Africa
Email: jsanders@aims.ac.za

Abstract

Random walks have been proposed as a simple method of efficiently searching, or disseminating information throughout, communication and sensor networks. In nature, animals (such as ants) tend to follow correlated random walks, i.e., random walks that are biased towards their current heading. In this paper, we investigate whether or not complementing random walks with directional bias can decrease the expected discovery and coverage times in networks. To do so, we develop a macro-level model of a directionally biased random walk based on Markov chains. By focussing on regular, connected networks, the model allows us to efficiently calculate expected coverage times for different network sizes and biases. Our analysis shows that directional bias can significantly reduce coverage time, but only when the bias is below a certain value which is dependent on the network size.

Keywords: Random walks, Markov chains, network coverage

1 Introduction

The concept of a random walk was introduced over a century ago by Pearson (Pearson 1905) and has been studied extensively since then (Dvoretzky & Erdős 1951, Brummelhuis & Hilhorst 1991, Caser & Hilhorst 1996). Recently, random walks have been proposed for searching, or disseminating information throughout, communications and sensor networks where the network’s structure is dynamic, or for other reasons unknown (Bar-Yossef et al. 2006, Dolev et al. 2006, Avin & Brito 2004, Sadagopan et al. 2005). They are ideal for this purpose as they require no support information like routing tables at nodes (Avin & Krishnamachari 2006) — the concept of a random walk being for the agent performing the walk to move randomly to any connected node.

The efficiency of random-walk-based algorithms can be measured in terms of the average number of steps the agent requires to cover every node in the network (and hence be guaranteed to find the target node in the case of search algorithms). This is referred to as the coverage time under the assumption that the agent takes one step per time unit. Obviously, improving the coverage time for algorithms is an important goal.

One straightforward approach to this is to have multiple agents (Alon et al. 2011). For some algorithms, such an approach is made even more effective when stigmergy is employed (Parunak 1997, Bonabeau et al. 1998, Ghosh et al. 2008), i.e., agents leave information for other agents directing them to their goals. Such an approach, inspired by the way ants leave trails of pheromones directing other ants to food (Camazine et al. 2001), is only useful when target nodes need to be visited by more than one agent. This is not always the case. More importantly, stigmergy is effective in directing agents only once a target node has been found. The time for the first agent to find a target node is not reduced. This can only be done by considering the agent’s ‘movement model’.

For this reason, it has been suggested that random walks should be constrained, e.g., to prevent an agent returning to its last visited node, or to direct an agent to parts of the network where relatively few nodes have been visited (Lima & Barros 2007, Randall et al. 2007). We take a similar approach in this paper. We base our movement model on that observed in nature. Many models used by biologists to describe the movement of ants and other animals are based on correlated random walks, i.e., random walks which are biased to the animal’s current direction (Kareiva & Shigesada 1983, Bovet & Benhamou 1988, McCulloch & Cain 1989). Based on our own observations of ants, we also investigate including a small probability of a non-biased step at any time to model occasional random direction changes.

To the best of our knowledge, the efficiency of directionally biased walks in networks have been investigated by only one other group of researchers. Fink et al. (Fink et al. 2012) look at the application of directional bias in a cyber-security system in which suspect malicious nodes must be visited by multiple agents. They compare coverage times for directional bias, with those for pure random walks, and random walks with stigmergy. Their conclusion is that directionally biased walks are more efficient even than random walks with stigmergy. This conclusion, however, is based on micro-level simulation, i.e., direct simulation of agents taking steps, for a single network size and bias. It cannot be generalised to arbitrary
network size or bias.

The micro-level simulation approach of Fink et al. requires coverage times to be calculated as the average of multiple runs. They performed 500 simulation runs for each movement model. Such an approach is impractical for a deeper investigation of the effect of directional bias which considers various network sizes and biases. For that reason, in this paper we develop a more abstract, macro-level model of a directionally biased walk. It builds on the work of Mian et al. (Mian et al. 2010) for random walks, describes the directionally biased walk in terms of a Markov chain (Norris 1998) and allows us to calculate the coverage time for a given network size and bias directly. Although certain special cases have analytic solutions, we have found this model to be helpful for a general approach to calculating coverage time.

This paper extends our previous work (Smith et al. 2014) by including a mathematical description of the macro-level model. We begin in Section 2 by describing the concept of a random walk, and our notion of directional bias in more detail. In Section 3 we present the Markov-chain model of a directionally biased walk on a network and show how it can be used to calculate the expected coverage time. In Section 4 we present and discuss the results of applying our model to the calculation of coverage times on a range of network sizes and biases. We conclude in Section 5.

2 Directionally biased walks

Random walks have been studied in 1-dimensional, 2-dimensional and multi-dimensional spaces. Many of the results from 2-dimensional walks are applicable to communications and sensor networks which are commonly modelled as connected graphs. In particular, it is known that with probability 1 a random walk will cover every node of a connected graph (Aldous 1989) and a number of approaches for calculating the coverage time have been proposed (Aldous 1991, Dembo et al. 2004, Lima & Barros 2007, Mian et al. 2010). The investigation in this paper focusses on regular, connected graphs where each node has exactly 8 neighbours (see Fig. 1(a) and, for the probability of the next step in a random walk over such a graph, Fig. 1(b)). Furthermore, to allow our graphs and hence networks to be finite, we wrap the north and south edges and the east and west edges to form a torus. Our aim is to provide a deeper analysis of directional bias than in the recent literature which also investigates the notion on regular toroidal graphs (Fink et al. 2012). While the results do not apply directly to arbitrary irregular networks, they do apply to random geometric graphs which are often used to model ad hoc sensor networks. Approaches using regular toroidal graphs to determine coverage time on random geometric graphs include that of Lima and Barros (Lima & Barros 2007) and Mian, Beraldi and Baldoni (Mian et al. 2010).

For modelling directional bias in nature, biologists typically use the von Mises distribution (Crist & MacMahon 1991). The von Mises distribution is a continuous angular function with a parameter $\kappa$ which affects heading bias (see Figure 2).

We do not adopt the von Mises distribution in our approach for two reasons. Firstly, we have only a discrete number of directions and so do not require a continuous distribution. Secondly, as in random walks, we would like the computations the agent needs to perform to be simple. Our notion of directional bias limits our agent to choose either its current direction with a probability $p$ (referred to as the bias), or any neighbouring direction, i.e., $\pi/4$ radians (45°) clockwise or anti-clockwise from the current direction, with equal probability of $1-p/2$. When the bias $p$ is high (as illustrated in Fig. 3(a)), the movement model approximates (discretely) that of the von Mises distribution for a high value of $\kappa$ (such as $\kappa = 4$ in Fig. 2). This is not the case, however, for low values of $p$ (as illustrated in Fig. 3(b)).

Let $\text{Direction} = \{0, \pi/4, \pi/2, 3\pi/4, \pi, 5\pi/4, 3\pi/2, 7\pi/4\}$ be the possible directions in radians. Our notion of directional bias is then defined formally as follows.

Definition 1 (Directional bias) Given the current direction $d \in \text{Direction}$ and bias $p \in [0, 1]$, the probability of moving in direction $d' \in \text{Direction}$ at the next step, $P(d')$, is defined as follows.

$$d' = d \Rightarrow P(d') = p$$
$$d' \in \{(2\pi + (d - \pi/4)) \mod 2\pi, (d + \pi/4) \mod 2\pi\} \Rightarrow P(d') = (1-p)/2$$
$$d' \notin \{d, (2\pi + (d - \pi/4)) \mod 2\pi, (d + \pi/4) \mod 2\pi\} \Rightarrow P(d') = 0$$

We also investigate adding occasional random steps to our directionally biased walks. The idea is that with probability $r$ the agent will make a random, rather than directionally biased, step. This better matches our own observations of the movements of ants.

Definition 2 (Directional bias with random steps) Given the current direction $d \in \text{Direction}$, bias $p \in [0, 1]$ and probability $r \in [0, 1]$ of a random step, the probability of moving in direction $d' \in \text{Direction}$ at the next step, $P(d')$, is defined as follows.
with bias, $p = 3/5$
\begin{align*}
2/5 & \quad 2/5 \\
1/5 & \\
\end{align*}
with bias, $p = 1/5$
\begin{align*}
(b) \text{ Probability of next step} & \\
3/5 & \quad 1/5 \\
1/5 & \\
\end{align*}
\begin{align*}
(2/5 & \quad 2/5) \mod 2\pi, (d + \pi/4) \mod 2\pi} \\
\Rightarrow P(d') & = r * 1/8 + (1-r) / 2} \\
\Rightarrow P(d') & = r * 1/8}
\end{align*}
To analyse coverage time under our models of directional bias, we adapt a Markov-chain model (Norris 1998) developed for random walks by Mian et al. (Mian et al. 2010). As explained in Section 3, this allows us to calculate the coverage time directly, and hence compare coverage times for different network sizes and biases.

3 A macro-level model

The previous work on directional bias by Fink et al. (Fink et al. 2012) shows that directionally biased walks are more efficient than random walks on a regular, connected toroidal graph; but only for the one specific network size and bias considered in their paper. In this paper, we produce more general results by investigating the effect on coverage time of varying the network size and directional bias. The micro-level model and simulation approach used by Fink et al. is not suited to this goal, requiring numerous simulations runs to calculate the coverage time for each network size and bias. We therefore use a more abstract, macro-level model which allows us to calculate the coverage time for a given network size and bias directly.

Our model is based on the work of Mian et al. (Mian et al. 2010) who provide a Markov-chain approach (Norris 1998) to model and calculate coverage time for random walks on a regular, connected toroidal graph. Given a network of $N$ nodes, let the vector $v$ of length $N$ denote the state probability distribution with elements $v_i$ for $0 \leq i < N$, and the matrix $M$ of size $N \times N$ denote the transition probability matrix with elements $M_{i,j}$ for $0 \leq i, j < N$.

Definition 3 (Markov-chain model) Let $v^{(0)}$ denote the initial state distribution and $v^{(k)}$ denote the state distribution after the $k$th step.

- The elements of the state probability distribution sum to 1.
\[
\forall k \geq 0 \bullet \sum_{i=0}^{N-1} v_i^{(k)} = 1
\]
- The rows of the transition probability matrix sum to 1.
\[
\forall i < N \bullet \sum_{j=0}^{N-1} M_{i,j} = 1
\]
- The state distribution at step $k$ is calculated by multiplying the initial distribution by the transition probability matrix $k$ times.
\[
\forall k > 0 \bullet v^{(k)} = v^{(0)} M^k
\]

Often the system described by such a model would begin in a particular node with probability 1.
\[
\exists i < N \bullet v^{(0)}_i = 1 \land (\forall j < N \bullet j \neq i \Rightarrow v^{(0)}_j = 0)
\]
For a random walk, we call this node the starting node. A random walk is specified by letting the transition from a node to any of its neighbours occur with a probability of $1/n$ where $n$ is the number of neighbours. For example, for a 1-dimensional network of 5 nodes such that
\[
v^{(0)} = (0 \quad 0 \quad 1 \quad 0 \quad 0)
\]
the transition probability matrix for a random walk would be
\[
M = \begin{pmatrix}
0 & 1/2 & 0 & 1/2 & 0 \\
1/2 & 0 & 1/2 & 0 & 0 \\
0 & 1/2 & 0 & 1/2 & 0 \\
0 & 1/2 & 0 & 1/2 & 0 \\
1/2 & 0 & 0 & 1/2 & 0
\end{pmatrix}
\]
where, for example, row 0 (the topmost row of $M$) indicates that an agent at node 0 (the first node of $v$) has a probability of $1/2$ of moving to node 1 and a probability of $1/2$ of moving to node 4 (since we wrap the east and west edges). Multiplying $v^{(0)}$ by $M$ results in
\[
v^{(1)} = (0 \quad 1/2 \quad 0 \quad 1/2 \quad 0)
\]
and so on. For a 2-dimensional network of $n \times m$ nodes, the vector $v$ would have $n \times m$ elements, with those from $i \times m \text{ to } (i+1) \times m - 1$ for $0 \leq i < n$ denoting the nodes in the $i$th row of the network. So, for example, row 0 of the matrix for a random walk on a network of 5 $\times$ 5 nodes is shown in Figure 4.

In order to calculate coverage time, we modify the standard Markov-chain model for a random walk so that the starting node is an absorbing node, i.e., a node from which the probability of a transition to any neighbour is 0 (and the probability of a transition to itself is 1). We then model the system as starting from the state distribution after the initial distribution, i.e., that in which all neighbours of the starting node have probability $1/n$ where $n$ is the number of neighbours per node.

Definition 4 (Markov-chain model with absorbing node) Given a Markov-chain model as defined in Definition 3, let $s < N$ be the position of the starting node. A Markov-chain model with absorbing starting node is defined in terms of a transition probability matrix $M'$ and initial state probability distribution $v^{(0)}$ as follows.
The initial state probability distribution is that reached after 1 step from a distribution in which the agent is in the starting node with probability 1.

\[ v^{(0)} = v M \]

where \( v_s = 1 \land (\forall j < N \bullet j \neq s \Rightarrow v_j = 0) \)

- The starting node transitions to itself with probability 1 in \( M' \).

\[ M'_{s,s} = 1 \land (\forall j < N \bullet j \neq s \Rightarrow M'_{s,j} = 0) \]

- All other transitions in \( M' \) are as in \( M \).

\[ \forall i,j < N \bullet i \neq s \Rightarrow M'_{i,j} = M_{i,j} \]

Due to the starting node being an absorbing node, the probability of being in the starting node never decreases as the number of steps increase. For example, for the 1-dimensional network above, the transition probability matrix is

\[
M' = \begin{pmatrix}
0 & \frac{1}{8} & 0 & 0 & \frac{1}{8} \\
\frac{1}{8} & 0 & \frac{1}{8} & 0 & \frac{1}{8} \\
0 & \frac{1}{8} & 0 & 0 & \frac{1}{8} \\
0 & 0 & \frac{1}{2} & 0 & \frac{1}{2} \\
\frac{1}{2} & 0 & 0 & \frac{1}{2} & 0
\end{pmatrix}
\]

and the state probability distribution is

\[ v^{(0)} = (0 \frac{1}{8} 0 \frac{1}{8} 0) \]
\[ v^{(1)} = (\frac{1}{2} \frac{1}{8} \frac{1}{8} \frac{1}{8} \frac{1}{8}) \]
\[ v^{(2)} = (\frac{5}{8} \frac{1}{8} \frac{1}{8} \frac{1}{8} \frac{1}{8}) \]
\[ v^{(3)} = (\frac{1}{16} \frac{1}{16} \frac{1}{16} \frac{1}{16} \frac{3}{16}) \]
\[ v^{(4)} = (\frac{3}{32} \frac{1}{16} \frac{1}{16} \frac{1}{16} \frac{3}{32}) \]

and so on.

The probability of the starting node at step \( k \) in this model is the probability that the system has returned to the starting node within \( k \) steps. It can be used to calculate the coverage time as follows.

Let \( \gamma_i \) denote the expected number of new nodes covered in the \( i \)th step. The total expected number of nodes covered at the \( k \)th step, \( C_k \), is then

\[
C_k = \sum_{i=0}^{k} \gamma_i.
\]

The initial node is covered at step 0, so we have \( \gamma_0 = 1 \). For all \( k > 0 \), \( \gamma_k \) is equal to the probability that the node, \( n_k \), reached at the \( k \)th step has not been visited before, i.e., \( \gamma_k = P(n_k \notin \{ n_i \mid i < k \}) \) or

\[ \gamma_k = P(n_k \neq n_0 \land \ldots \land n_k \neq n_{k-1}) \]

Due to the regularity of the network and the fact that an agent behaves the same at each node, the probability of returning to the starting node after, say, 10 steps is equal to the probability of returning to the second node reached after 12 steps. More generally, we have \( P(n_k = n_0) = P(n_k = n_i) \). From which it follows that \( P(n_k = n_0) = P(n_k = n_i) \). Hence, from above

\[
\gamma_k = P(n_0 \neq n_1 \land \ldots \land n_0 \neq n_k)
\]

which is equal to the probability that the system has not returned to the starting node, \( n_0 \), within \( k \) steps. In other words, given the modified Markov-chain model of Definition 4

\[
C_k = \sum_{i=0}^{k} (1 - v^{(i)}_s).
\]

Coverage time can then be defined as follows.

Definition 5 (Coverage time) Given a Markov-chain model with absorbing node as defined in Definition 4, the time to cover \( x \% \) of the nodes is the smallest \( k \geq 0 \) such that

\[
C_k = \sum_{i=0}^{k} (1 - v^{(i)}_s) = (x/100) \times N.
\]

With directional bias, we add an additional dimension to our representation of a network: the current direction of movement. For a network with \( N \) nodes, the number of entries in the state probability distribution \( v \) is hence no longer \( N \) but \( n \times N \) where \( n \) is the number of neighbours per node (and hence the number of directions of movement). Each entry represents the probability of being in a node having entered from a specific direction. We organise these entries so that those from \( d \times N \ldots d \times N + N - 1 \) for \( 0 \leq d < n \) denote the probabilities of being in a node of the network having entered from direction \( d \). The corresponding transition probability matrix \( M' \) for the Markov-chain model is of size \( n \times N \times n \times N \), and since there are \( n \) positions corresponding to the starting node (one for each direction from which the starting node was entered) there will be \( n \) absorbing positions in the matrix.

Definition 6 (Markov-chain model for directional bias) Let \( M \) be a transition probability matrix of size \( n \times N \times n \times N \) (such that all rows sum to 1). A Markov-chain model for directional bias is defined in terms of a transition probability matrix \( M' \) and initial state probability distribution \( v^{(0)} \) as follows.

- The initial state probability distribution is that reached after 1 step from a distribution in which the agent is in the starting node (with a particular current direction) with probability 1.

\[ v^{(0)} = v M \]

where

\[
\exists d < n \bullet v_{d \times N + s} = 1 \land (\forall j < n \times N \bullet j \neq d \times N + s \Rightarrow v_j = 0)
\]
• The starting node transitions to itself (without changing the current direction) with probability 1 in $M'$. 
\[
\forall d < n \bullet M'_{d,N+s,d,N+s} = 1 \wedge (\forall j < n \bullet N \wedge j \neq d \Rightarrow M'_{d,N+s,j} = 0)
\]

• All other transitions in $M'$ are as in $M$.
\[
\forall i,j < n \bullet N \Rightarrow M'_{i,j} = M_{i,j}
\]

For example, consider adding directional bias to the 1-dimensional network of 5 nodes. Let the probability of continuing in the current direction be $p = 3/4$, and that of changing direction to be $(1 - p) = 1/4$. Let 0 denote direction east (or right) and 1 denote direction west (or left), and let the starting node be node 2 (the centre node). The transition probability matrix is then

\[
M' = \begin{pmatrix}
0 & 3/4 & 0 & 0 & 0 & 0 & 0 & 0 & 1/4 \\
0 & 0 & 1/2 & 0 & 0 & 1/2 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1/2 & 0 & 1/2 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 1/2 & 0 & 1/2 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\end{pmatrix}
\]

where, for example, row 0 indicates that an agent at node 0 whose current direction is east will move to node 1 (and not change the direction) with probability 3/4 and to node 4 (changing the current direction to west) with probability 1/4. Also, rows 2 and 7 indicate that an agent in node 2 will remain in node 2 and not change the current direction. If the starting node’s initial direction is set to east then the state probability distribution is

\[
v = \begin{pmatrix}
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0
\end{pmatrix}
\]

and hence the initial state probability distribution for the Markov-chain is

\[
v^{(0)} = \begin{pmatrix}
0 & 0 & 0 & 3/4 & 0 & 0 & 1/2 & 0 & 0 & 0
\end{pmatrix}
\]

4 Investigating directional bias

To perform our investigation into directional bias, we implemented our Markov-chain model for directional bias (Definition 6) and coverage time (Definition 7) in Java using the JAMA library for matrices and matrix operations (math.nist.gov/javanumerics/jama) and the JFreeChart library for plotting graphs (www.jfree.org/freechart).

Initially, we plotted graphs of coverage time versus bias (for bias values from 0 to 0.95 in steps of 0.05) for graph sizes $5 \times 5$ (25 nodes) to $15 \times 15$ (225 nodes). We calculated the time for coverage of 99% of the network nodes. This was to avoid problems arising with 100% coverage when the coverage converged to a point just below the network size due to inaccuracies in the floating-point arithmetic.

The movement model was that of Definition 1. The graphs for the $5 \times 5$ and $15 \times 15$ networks are shown in Fig. 5 and Fig. 6, respectively. The horizontal line represents the coverage time for a random walk, and the curved line that for a directionally biased walk under the range of biases. The general shape of the latter and its position in relation to the horizontal line for random bias was consistent for all network sizes in the range considered.

A number of interesting results follow from this analysis.

1. The best coverage time is achieved for a bias of 0. This corresponds to an agent which always changes direction by $\pi/4$ radians on every step.

2. While for low directional biases (0 up to around 0.7 for the $5 \times 5$ case) coverage time is less than that for a random walk, for higher biases it is greater than that for a random walk.

3. The value of the bias at which a directionally biased walk becomes less efficient than a random walk (from here on called the cross-over bias), progressively increases as the size of the network increases. It is around 0.74 for a $5 \times 5$ network, and 0.93 for a $15 \times 15$ network.

4. The improvement in efficiency of directional bias increases as the size of the network increases. For a directional bias of 0.5 the increase in efficiency is less than 25% for a $5 \times 5$ network, and around 60% for a $15 \times 15$ network.

Point 1 is particularly interesting as it suggests a new movement model that was not initially anticipated. Our initial motivation was to investigate movement models similar to those observed in nature, which are best represented by a von Mises distribution. As illustrated in Fig. 3(b), however, low values of bias in our movement model (including the value 0) do not correspond to a von Mises distribution. The new model, although perhaps impractical as a means of movement in nature, can nevertheless be readily implemented in network search and dissemination algorithms.

Point 2 is also interesting as it indicates that directional bias is only effective in reducing coverage time when the bias is not too large. This result was also unanticipated as directional bias in the movement of animals tends to be high. However, the areas over which such animals move would correspond to networks significantly larger than those we considered. Point 3 anticipates that the cross-over bias would be higher in such networks. This conjecture is supported by the work of Fink et al. (Fink et al. 2012) whose
micro-level simulation of a 2-dimensional network of 100 × 100 nodes shows that a directionally biased walk (approximating a von Mises distribution with high \( \kappa \)) is more efficient than a random walk.

The second part of our investigation considered the movement model of Definition 2, where occasional random steps are added to a directionally biased walk. Example plots for a network of 5 × 5 nodes and the value of \( r \) set to 0.1 (an average of one random step in every 10) and 0.2 (an average of one random step in every 5) are shown in Fig. 7 and Fig. 8, respectively. Similar plots for a 15 × 15 network are shown in Fig. 9 and Fig. 10.

The following results emerge from this analysis.

1. As may have been predicted, the addition of random steps moves the coverage time closer to that of a random walk. Hence, for bias values lower than the cross-over bias the coverage time increases, but for values higher than the cross-over value the coverage time decreases. Comparing Fig. 8 with Fig. 5 it can be seen that for a bias of 0 the coverage time has increased from around 55 to 70, and for a bias of 0.95 it has decreased significantly from around 470 to about 158. Comparing Fig. 10 with Fig. 6 it can be seen that for a bias of 0 the coverage time has increased from around 970 to 1050, and for a bias of 0.95 decreased significantly from over 1800 to around 1100.

2. The introduction of random steps increases the cross-over bias. For the 5 × 5 network, with \( r = 0.1 \) the cross-over bias increases to 0.78 (from 0.74 for no random steps) and for \( r = 0.2 \) to 0.84. For the 15 × 15 network, the cross-over bias increases from around 0.93 to a value higher than 0.95 (the last bias plotted) for both values of \( r \).

5 Conclusion

In this paper we have investigated the effect of directional bias on the coverage time of random walks on regular, connected networks. Our analysis has shown that directional bias can reduce coverage time signif-
Figure 7: Random vs directionally biased walk with probability 0.1 of a random step for a 2-dimensional network of $5 \times 5$ nodes.

Figure 8: Random vs directionally biased walk with probability 0.2 of a random step for a 2-dimensional network of $5 \times 5$ nodes.

icantly and has a greater effect the larger the network. However, this reduction occurs only when the bias (to continue in the same direction) is below a certain value we call the cross-over bias. The cross-over bias is dependent on the network size, increasing as the size of the network increases. Hence, high values of bias which work well at reducing coverage time in large networks, may be less effective, or even increase the coverage time, in smaller networks.

The cross-over bias can be increased by adding occasional random steps to a directionally biased walk — the more random steps, the higher the cross-over bias. Adding such steps, however, moves the coverage time of a directionally biased walk closer to that of a random walk (increasing coverage time for biases below the cross-over bias).

Our analysis also revealed that a movement model in which an agent changes direction by $\pi/4$ radians (in either direction with equal probability) on each step is more effective in reducing coverage time than a standard directionally biased model. Further investigation of this, and similar models, is warranted.

Our investigation was facilitated by a macro-level model of random and directionally biased walks in terms of Markov chains. This model allowed us to calculate coverage time directly, in contrast to other approaches where coverage time is calculated as the average result obtained from numerous runs of a micro-level simulation. Although introducing a margin of error due to the limitations of floating-point arithmetic, our more abstract model has provided a practical means for obtaining a deeper, more complete analysis of directional bias than in previous work.

References


Figure 9: Random vs directionally biased walk with probability 0.1 of a random step for a 2-dimensional network of 15 × 15 nodes.

Figure 10: Random vs directionally biased walk with probability 0.2 of a random step for a 2-dimensional network of 15 × 15 nodes.


Camazine, S., Deneubourg, J., Franks, N., Sneyd, J., Theraulaz, G. & Bonabeau, E. (2001), Self-


An Evaluation of POS Tagging for Tweets Using HMM Modelling

Parma Nand Rivindu Perera
School of Computer and Mathematical Sciences
Auckland University of Technology,
Auckland, New Zealand,
Email: {pnand, rperara}@aut.ac.nz

Abstract

Recently there has been an increased demand for natural language processing tools that work well on unstructured and noisy texts such as texts from Twitter messages. It has been shown that tools developed for structured texts, do not work well when used on unstructured texts hence necessitates considerable customization and re-training for the tools to be able to achieve the same accuracy on unstructured texts. This paper presents the results of testing a HMM (Hidden Markov Model) based POS (Part-Of-Speech) tagger customized for unstructured texts.

The tagger was trained on Tweter messages on existing publicly available data and customized for abbreviations and named entities common in Tweets. We evaluated the tagger firstly training and testing on the same source corpus and later did cross-validation testing by training on one Twitter corpus and testing on a different Twitter corpus. We also did similar experiments with the datasets using a CRF (Conditional Random Frequency) based state-of-the-art POS tagger customized for Tweet messages.

The results show that the CRF-based POS tagger from GATE performed slightly better compared to the HMM model at token level, however at the sentence level the performances were approximately the same. An even more intriguing result was that the cross-validation experiments showed that both the tagger's results deteriorated by approximately 25% at the token level and a massive 80% at the sentence level. This suggests vast differences in the two Tweet corpora used and emphasizes the importance of recall values for NLP systems. A detailed analysis of this deterioration is presented and the HMM trained model together with the data has also been made available for research purposes.

Keywords: Social Media, HMM POS Tagger, Twitter, Machine Learning, POS Tagging

1 Introduction

In the last five years, there has been a significant shift in the way we communicate on the internet. Instead of structured texts, there has been a shift towards loosely structured, short interactive messages. Although, this initially started with text messaging on mobile phones which had a limitation of 140 characters, it has since also proliferated into online communication on popular sites such as Facebook, Twitter, Blogs and Flickr. With such an increase in communication micro-blogging type texts, there has been an increase in demand for appropriate text processing tools for various purposes such as business intelligence and security. Extracting information from such blogs is one of the hardest problems in NLP because of their structure and switching between the type of conversation from one-to-one, multi-party and broadcast messages. NLP methods that work well for longer texts (e.g. named entity recognition, topic identification) have been found to work poorly on blogs and tweets. This has created a need to either adapt existing methods for use with microblog content or find new methods that work well in the specialised domain of micro-blog texts. In response to this need, there has been a flurry of research in recent times from the linguistic point of view in trying to understand the structure of micro-blogging texts, eg., (Monojit Choudhury, Rahul Saraf, Vijit Jain, Sudeshna Sarkar et al., 2007; Cooper et al., 2005; Finin et al, 2010), as well as computational viewpoint in trying to extract information from such texts (Gimpel et al, 2011; Ritter et al, 2010; Barbosa and Feng, 2010; Soderland et al, 1999).

Many of the commonly used IE (Information Extraction) implementations such as Lingpipe1 and An- nieLingpipe2 depend on POS tagging in order to perform the downstream tasks, hence this is a crucial step for accuracy in Information Extraction. Initial attempts at POS tagging were done using deterministic, rule-based techniques and some attempts such as (Greene and Rubin, 1971) achieved accuracies as high as 77%. However the inherent difficulties with deterministic techniques such as rule base maintenance and limitations on transportability meant a shift towards probabilistic or stochastic techniques resulting in most of the recent works primarily based on probabilistic techniques with some incorporation of rules. A necessary component of stochastic techniques is supervised learning, which requires annotated training data. Creation of such data is both expensive and time consuming.

Although there are now several sources of accurate POS annotated corpora available in the structured text genre (eg. Penn Tree Bank, Brown Corpus, and MedPost), there is still a dearth of tagged corpora for unstructured texts such as micro-blogs and tweets. Our search for publicly available POS tagged dataset for micro-blogging type texts yielded the following three sources.

1http://alias-i.com/lingpipe.
2http://gate.ac.uk/ie/annie.html
• The T-POS dataset (Ritter et al., 2011) consists of 12K tokens from Twitter messages. The corpus uses a set of 38 tags from Penn Treebank (PTB) with additional 4 tags specific to Twitter messages.
• The DCU dataset (Foster et al., 2011) consists of 14K tokens from Twitter messages. This dataset uses a conflated set of 20 tags from PTB with additional 5 Twitter specific tags.
• The ARK dataset (Gimpel et al., 2011) consists of 39K tokens from Twitter messages. The corpus uses a conflated set of 20 tags from PTB with additional 5 Twitter specific tags.

Each of the datasets described above has been used in POS tagging experiments and report accuracies of up to 92% using various forms of discriminative models. Gimpel et al (Gimpel et al., 2011) report an accuracy of 92.8% with the ARK dataset using a Conditional Random Field (CRF) estimator. Derczynski et al (Leon Derczynski, Alan Ritter, 2013) again use a CRF estimator on both the T-POS and the DCU datasets and report accuracies as high as 88.7%. Although it is generally accepted that discriminative models (eg. CRF) models perform better than generative (eg. HMM) models (Sutton and McCallum, 2001), generative models by the virtue of their design have some key advantages compared to discriminative models.

One of the key advantages that is pertinent to micro-blogging data, is that generative models are able to better handle datasets which are only partially labelled or completely unlabelled. Generative models require the computation of joint probability distribution of labels and words. The computation for the words does not require labelled data, hence, the probability distribution of words can take advantage of large amounts of unlabelled data for initial training as well as “live” data for real time systems. Secondly, in some cases, as demonstrated by Ng and Jordan (Andrew Y. Ng, 2001), generative models perform better when the input model has a smoothing effect on the features. Their results show that the advantage of generative models is even more pronounced when the dataset is small as is currently the case for labelled micro-blogging data. A third advantage of generative models is that it’s training time is insignificant compared to discriminative models, hence has an advantage in real time applications where progressive learning is required.

On the other hand discriminative models have better generalization performance when training data is abundant with the ability to account for more global features compared to a generative model. This gives discriminative models the ability to model features from any arbitrary part of a sentence, not necessarily in a linear fashion. This freedom enables it to model a much larger set of features, however it also exposes the model to the risk of overfitting the training which leads to poor generalization on unseen data. For a detailed discussion and comparison of various probabilistic models see Kalinger (Roman Klinger, Katrin Tomancek, 2007).

This paper investigates the generalization ability of two discriminative, pre-trained Twitter tagging systems and evaluates them against a generative model using three Twitter datasets, T-POS, DCU and ARK. We used the Hidden Markov Model (HMM), the basic implementation of which as adapted from LingPipe3. The HMM tagger was used to train on a subset of the tweet data in each of the datasets and the results were compared against two discriminative models, a Maximum Entropy model as implemented in the Stanford POS Tagger4 and a highly customized CRF model implemented as an extension in GATE5. We also did cross-dataset validation and observed that the performance deteriorates by approximately 20% in all the models tested. In this paper we present an evaluation of the comparative performance of the two classes of taggers as well as an analysis of the deterioration of the results in the cross validation experiment.

In summary this paper make the following contributions:
• Makes the source code and jar publicly available6 for further research or to use the HMM tagger for tagging Tweet messages.
• Presents the results of direct comparison between three types of POS taggers trained and tested on Tweet messages.
• Presents an evaluation of cross-dataset generalizability of three classes of tagging models.
• Presents a detailed analysis of the error contribution for each class of tagging model.

2 Characteristics of Tweet Data

There is much linguistic noise in micro-blogging texts as a result of the ways in which micro-blogs are written. The noise arise from different language usage characteristics, hence different strategies are required to account for them. The following are a list of the characteristics with a discussion of the challenges resulting due to the linguistic noise.

Word Capitalization Use of capitalization in micro-blogs is inconsistent. In formal texts, capitalization is used as a key feature for recognizing proper nouns (tags NNP and NNPS), however in micro-blogs capital letters are used for several other purposes such as emphasis (eg. “.tomorow YOU will need to do that”) and highlighting (eg. “.lease do me a favor and POST CHAPTER 13”). Micro-blog texts also contain a plethora of capitalization due to typos. Apart from these, noise is also introduced by a lack of capitalization of nouns which should be capitalized. Various techniques such as use of name lexicon lists (Leon Derczynski, Alan Ritter, 2013) and the use of a trained classifier to recognize a valid capitalization (Ritter et al, 2011) have been used to account for the noise due to capitalization.

Word Variations Spelling variations could be unintentional, since Microblog texts rarely get proofread, or intentional, for the purpose of compressing long words, eg. use of tmro and 2moro for tomorrow. Word compressions can take either a graphemic or a phonemic form (Monijit Choudhury, Rahul Saraf, Vijit Jain, Sudeshna Sarkar et al, 2007). Graphemic forms involve deletion vowels (eg. “msg”), deletion of repeated characters (eg. “tomorow” for “tomorrow”) and truncation (eg. “tom” for “tomorrow”). Phonemic

4http://nlp.stanford.edu/software/tagger.shtml
5http://gate.ac.uk/uiki/twitter-postagger.html
6http://staff.elena.aut.ac.nz/Parmanand/projects/TaggerDownload.html
forms involve substitution of a shorter set of characters to represent the same phoneme, e.g. “2” in “2moro”. Choudhury et al report decoding such spelling variations with an 80% success using a HMM model.

Multi Word Abbreviation Frequently, multiple words are abbreviated as single word abbreviations, eg. “lol” for “laugh out loud” and “gif” for “thank god its Friday”. These abbreviations can only be identified by use of lexicons.

Slangs Slangs such as “gonna” used for “going to” is common since it reduces the size of the sentence. Lexicons with word-to-phrase maps have been used with varying levels of success to account for such use of slangs (Gimpel et al, 2011; Derczynski et al, 2013).

Word Omission Frequently used function words such as articles and subject nouns are omitted. For example, “I went to town in a car” may be written as “went town in car”. This category of noise is easily handled by training sequence based probabilistic models and both CRF and HMM models perform well with this type of noise.

3 Data Sets Used

Currently, there are 3 sets of tagged datasets on tweet texts, two of them (TPOS and DCU) use similar PTB tag set, while the third one (ARK) uses a much smaller subset of 25 tags. The list of tags are shown in Figures 1 and 2 for TPOS/DCU and ARK respectively.

The ARK dataset, introduced in (Gimpel et al, 2011), used 17 annotators to tag a total of 1527 English Tweets containing approximately 26,000 tokens. The paper reports an inter annotator-agreement rate of 92.2%. This data was then used to train a CRF tagger model with additional 5 features to handle various types of linguistic noise such as orthography, phonetics and capitalization. The authors report an accuracy of 89.37% compared to 85.85% for a retrained Stanford tagger.

The TPOS dataset based on PTB was first introduced in (Ritter et al, 2011), contains 12K tokens from Tweet messages. The authors report a tagging accuracy of 88.3% with a CRF tagger model trained on a mixture of TPOS (12K), IRC7 (40K) and PTB (50K). The accuracy reported from this study was 88.3% compared to 80.01% for the Stanford tagger.

The DCU dataset, again based on PTB, introduced in Foster (Foster et al, 2011) contains 269 annotated sentences which had a reported inter-annotator agreement rate of 95.8%. The authors use the dataset to train and test a Support Vector Machine tagger and report an accuracy of 84.4% compared to an accuracy of 96.3% for Wall Street Journal data. This paper focusses on parsing, hence no tagging specific up-training is done to account for the linguistic noise, however the public availability of the tagged data is extremely useful for Twitter tagging research.

4 Experimental Setup

The testing was conducted using the 3 different sets of Tweet data described in section 3 so that cross validation performance could also be determined. We used the Stanford tagger (Stanford) and the enhanced Stanford tagger (Gate), both shipped with the GATE package8. The Stanford tagger has been shown to exceed accuracies of over 90% (Leon Derczynski, Alan Ritter, 2013), hence is considered to be state-of-the-art. We used the Stanford tagger as a benchmark, and did cross validation tests on its enhanced version (Gate tagger) across the three datasets and also did comparison tests against a newly introduced HMM model.

For the Stanford and Gate taggers, we used the pre-trained standard models (as opposed to faster models with lower accuracy) named english-twitter.model and gate-EN-twitter.model respectively. These two models were tested against a twitter trained HMM model, modified from the basic implementation in LingPipe9. A final evaluation was done to compare the training times between a CRF and a HMM model as implemented in LingPipe. This evaluation was done without the implementation of any additional features in both the CRF and the HMM model for the purpose of comparing the computational cost for the two models.

Table 1 gives the details of the data splits between training and testing sets used in the evaluation experiment. The training and testing splits for T-POS and DCU dataset is the same as that used for training the pre-trained GATE models reported in Derczynski et al (Leon Derczynski, Alan Ritter, 2013).

Figure 1: Alphabetical list of tags from Penn Treebank used in the TPOS and DCU dataset

Figure 2: Alphabetical list of tags used in the ARK dataset.

The PTB data sets contains the following 3 Twitter specific tags

2. HT - topic, eg. #newyear
3. USR - a tweet username, eg. @catlovesit

The ARK dataset contains the Twitter specific tags from PTB and in addition introduces two additional tags for continuation of a tweet ( ) and emoticon (eg. :-)). It uses the following symbols for the 5 additional Twitter specific tags:

2. # - topic, eg. #newyear
3. @ - a tweet username, eg. @catlovesit
4. E - emoticon, eg. :-)
5. - continuation of a tweet, always preceded by

The testing was conducted using the 3 different sets of Tweet data described in section 3 so that cross validation performance could also be determined. We used the Stanford tagger (Stanford) and the enhanced Stanford tagger (Gate), both shipped with the GATE package8. The Stanford tagger has been shown to exceed accuracies of over 90% (Leon Derczynski, Alan Ritter, 2013), hence is considered to be state-of-the-art. We used the Stanford tagger as a benchmark, and did cross validation tests on its enhanced version (Gate tagger) across the three datasets and also did comparison tests against a newly introduced HMM model.

For the Stanford and Gate taggers, we used the pre-trained standard models (as opposed to faster models with lower accuracy) named english-twitter.model and gate-EN-twitter.model respectively. These two models were tested against a twitter trained HMM model, modified from the basic implementation in LingPipe9. A final evaluation was done to compare the training times between a CRF and a HMM model as implemented in LingPipe. This evaluation was done without the implementation of any additional features in both the CRF and the HMM model for the purpose of comparing the computational cost for the two models.

Table 1 gives the details of the data splits between training and testing sets used in the evaluation experiment. The training and testing splits for T-POS and DCU dataset is the same as that used for training the pre-trained GATE models reported in Derczynski et al (Leon Derczynski, Alan Ritter, 2013).
The first test on the HMM model was done using the ARK data set split, into training and testing set as in Gimpel et. al (Gimpel et al, 2011). (shown on the last 2 rows of Table 1). The HMM model was initially tested for n-gram (number of previous tokens used for emissions) for values ranging from 1 to 10. An n-gram value of 5 gave us the best performance, and we fixed the HMM model at this value for all the other tests.

The HMM model was first trained by varying the amounts of training data and tested on the ARK-test data which consists of 547 individual Tweets. The accuracy very quickly reached 80% at about 300 Tweets after which the the increase was slow up to a maximum value of 87% for the total amount of training data of 1827 Tweets. This compares very well with the results reported in (Gimpel et al, 2011) for a substantially feature engineered CRF model which obtained a value of 89.37%. As a comparison, the Stanford tagger had an accuracy of 85.85% on the same training and test data.

The TPOS and the DCU data sets use the PTB tagset and hence the results for these two dataset are directly comparable. The ARK data set contains a smaller subset of 25 tags by conflating some of the PTB tages. A model trained on the subset dataset cannot be cross validated on a superset dataset, however the opposite is possible by mapping the superset onto the subset tagset. Hence, the tagging results from the superset trained systems was cross validated on the ARK dataset by mapping the superset tags to the subset tags, for example all forms of verbs (VB, VBD, VBG, VBN, VBP, VBV, VBD) were mapped to a single tag, V, in the ARK dataset.

Table 2 shows the results for the cross validation tests of the four models tested on three datasets. The pre-trained Stanford and Gate taggers tested on the TPOS-test and DCU-test data sets achieved relatively high accuracies, close to the reported values in (Leon Derczynski, Alan Ritter, 2013). For example, the Gate tagger achieved a token accuracy value of 93.5% on the TPOS-test data and 89.4% on the DCU-test data. The corresponding token accuracy values for the LingPipe tagger was 82.8% for TPOS-test and 82.7% for the DCU-test data. As another comparison, the dataset were also used to train and test a CRF model from LingPipe. This model which was devoid of any domain specific feature engineering gave much lower accuracy on all data sets, shown in the last row of Table 2. The LingPipe(CRF) model was tested mainly for the purpose of comparing the training time rather than for accuracy. As an indication the training time for the CRF model on the TPOS-train data set was approximately 8 hours for 200 epochs and had to be left overnight for 1000 epochs. This compares with less than 30 seconds for the HMM model for the same training data. For text processing tasks, each word is a feature. Hence for a discriminative model such as the CRF, there needs to be multiple iterations through the whole feature set in order to be able to find discriminative features for each of the tags. This adds a huge computational cost making it unsuitable for real time systems. On the other hand, a generative model such as HMM, needs to traverse through all the features (words and tags) once and determine the probability distribution of token and word emissions. This can then be easily updated as new features are encountered making it adaptable as new data is encountered making it suitable as a progressiver learner.

The three models were then cross validated by training them with TPOS-train data and tested against the ARK-test data. Since the ARK-test data uses a smaller set of tags, the output of the models trained on TPOS-train data were first mapped to the ARK tagset before running the evaluation tests. Intuitively, this was expected to give us an even higher accuracy since the mapping is “downward”. For example, all confusions between VB, VBD, VBG, VBN and VBP from PTB tagset were mapped to V from the ARK tagset, which would be expected to drive up the accuracy since we are evaluating against a more coarse set of tags. The accuracies obtained for this cross validation are shown in Table 2. All three models being evaluated (Stanford, Gate and Ling-pipe) trained on TPOS-train data give very close results for TPOS-test and DCU-test datasets, hence for cross validation, the results of these two accuracies were averaged and then compared with the ARK-test accuracy shown in Table 3. The cross validation results show a drop of token accuracy between 21 and 25 percent, while the sentence level drop is even more substantial with values of approximately 80%. The accuracy difference between the models cannot be attributed to the feature engineering in CRF models since a similar drop was also observed in the HMM model. The details of the confusions is investigated in section 5. The individual tag accuracies in Figure 3 shows very similar tag-based performance characteristics for both Gate and HMM models. The tags between ADVP and RBR were low in number (below 10) hence the 0% accuracy for the HMM model. Apart from the NNPS (singular proper noun) tag, Gate consistently performs better or equivalent for the rest of the tags. In the case of NNPS, Gate implements several feature engineering techniques in order to identify un-capitalized proper nouns, however in the case of the TPOS-test data, this was counter-intuitive compared to the HMM model which essentially uses capitalization to identify proper nouns. The figures in 4 and 5 show the individual tag performance for the Gate and the HMM models trained on TPOS-train data. Both the models show that there is an average of 20% drop in accuracy for ARK-test data and this is due to a consistent lower performance across all the tags rather than an aberration pertaining to a subset of tags, which could have been possible in the ARK-test data. A candid analysis of the tagging between TPOS and the ARK data sets did not show any gross tagging differences, hence the performance degradation has to be attributed to differences in higher level data characteristics. This is currently being investigated.

## 5 Error Analysis of the HMM Model on the TPOS data

The TPOS-test data contains a total of 2291 tokens tagged with 44 tags which consists of 41 PTB tags and additional 3 twitter specific tags. Table 4 shows the confusion distribution for the tags which contributed more than 20% error for the HMM model tested on the TPOS data. The numbers in brackets in the 3

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Sentences</th>
<th>Tokens</th>
</tr>
</thead>
<tbody>
<tr>
<td>T-POS-train</td>
<td>551</td>
<td>10.6K</td>
</tr>
<tr>
<td>T-POS-test</td>
<td>118</td>
<td>2.2K</td>
</tr>
<tr>
<td>DCU-train</td>
<td>269</td>
<td>2.9K</td>
</tr>
<tr>
<td>DCU-test</td>
<td>250</td>
<td>2.8K</td>
</tr>
<tr>
<td>ARK-train</td>
<td>1827</td>
<td>26.6K</td>
</tr>
<tr>
<td>ARK-test</td>
<td>547</td>
<td>7.7K</td>
</tr>
</tbody>
</table>

Table 1: Dataset Details
Table 2: Percentage Accuracies for the cross validation tests

<table>
<thead>
<tr>
<th>Model</th>
<th>TPOS-test</th>
<th></th>
<th>DCU-test</th>
<th></th>
<th>ARK-test</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Tok</td>
<td>Sent</td>
<td>Tok</td>
<td>Sent</td>
<td>Tok</td>
<td>Sent</td>
</tr>
<tr>
<td>Stanford(MaxENT)(pre-trained)</td>
<td>88.7</td>
<td>20.3</td>
<td>89.4</td>
<td>36.8</td>
<td>69.6</td>
<td>6.4</td>
</tr>
<tr>
<td>Gate(CRF)(pre-trained)</td>
<td>93.5</td>
<td>33.8</td>
<td>89.4</td>
<td>37.6</td>
<td>69.5</td>
<td>6.3</td>
</tr>
<tr>
<td>LingPipe(HMM)(TPOS-train)</td>
<td>82.8</td>
<td>25.2</td>
<td>82.7</td>
<td>24.8</td>
<td>61.9</td>
<td>4.8</td>
</tr>
<tr>
<td>LingPipe(HMM)(ARK-train)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>86.9</td>
<td>26.4</td>
</tr>
<tr>
<td>LingPipe(CRF)(TPOS-train)</td>
<td>69.7</td>
<td>4.4</td>
<td>64.0</td>
<td>4.1</td>
<td>63.2</td>
<td>4.1</td>
</tr>
</tbody>
</table>

Table 3: Percentage accuracy drops for cross validation tests for the three types of taggers.

<table>
<thead>
<tr>
<th>Model</th>
<th>TPOS/DCU-test avg.</th>
<th>ARK-test</th>
<th>%age drop</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Tok / Sent</td>
<td>Tok / Sent</td>
<td>Tok / Sent</td>
</tr>
<tr>
<td>Stanford(MaxENT) (pre-trained)</td>
<td>89.05</td>
<td>21.8</td>
<td>69.6</td>
</tr>
<tr>
<td>Gate(CRF) (pre-trained)</td>
<td>91.45</td>
<td>24.0</td>
<td>69.5</td>
</tr>
<tr>
<td>LingPipe(HMM) (TPOS-train)</td>
<td>82.75</td>
<td>25.2</td>
<td>61.9</td>
</tr>
</tbody>
</table>

Figure 3: A comparison of the Gate and HMM models’ performance for individual tags, sorted in ascending order for ARK data values. The sample used is for a TPOS trained model tested on TPOS-test data.

Figure 4: Gate Model’s performance on individual tags for TPOS-test and ARK-test Data, sorted in ascending order for ARK data values.

Figure 5: HMM Model’s performance on individual tags for TPOS-test and ARK-test Data, sorted in ascending order for ARK data values.
confusion columns show the number of confused instances for each of the tags. The lowest accuracy was achieved for the JJ (adjective) tag with an accuracy of 0.57. The JJ tag was confused with NN (common noun) 17 times, NNP (singular proper noun) 11 times and RB (adverb) 4 times. From the rest of the rows in table 4 it can be seen that the JJ tag features as the highest number of false positives for the other confused tags as well. Adjectives are most difficult tags to identify as many of the nouns also function as adjectives, as for example, “valuable” and “Costa Rica”. In these examples the tokens “valuable”, “Costa” and “Rican” function as adjectives, however were identified as nouns. Out of the 30 confusions for nouns and pronouns, 60% were in the category of compound noun modifier adjectives, which were identified as either nouns or proper nouns. The rest of the adjectives were of the form where an adjective was used in a position other than a pre-modifier. For example in “...I just felt special...”, the token “special” is functioning as an adjective in a syntactical position which is usually a noun in most clauses. Hence, in sequence oriented, probability based models such as HMM, the tagging for adjective will be biased towards nouns. The bias is only corrected if the exact token was present in the training data, which is why probability based models are only as good as the range and extent of the training data. Another category of example which accounted for a high proportion of confused adjectives was the token “long” in “...all year long...”. In this case the tokens “all” and “year” are a determiner and noun respectively, hence the adjective is a post modifier of the compound noun. This is another rare syntactical position for an adjective. The majority of adjectives in this category were classified as nouns as the last token of a compound noun is frequently a noun. The rest of the adjectives were of the form where an adjective is functioning as an adjective in a syntactical position which is usually a noun in most clauses. There were only 18 VBN tags in the test data and 4 of these were confused with VBD (past tense), which is a tag that can be represented by the same set of words distinguished only by a complex combination of the rest of the tokens in the sentence. The other significant confusion worth mentioning is the confusion of the NNP (proper noun) tag confused with NN (common noun). This is due to a lax capitalisation in tweet messages. The HMM model used for the testing used capitalisation in addition to the city, corporation and name lexicons from the Stanford implementation to tag proper nouns, hence the 24 confusions out of the 181 were outside these lexicons. Tagging of the NNP tags can be easily improved by extending the existing Stanford lexicon lists for specific applications.

### 6 Concluding Remarks

Social media micro-blogging sites such as Twitter contain vast amounts of information which is extremely current and, as a result, is fast changing. This necessitates text processing tools which are both efficient as well as has the ability to do progressive learning. Since POS tagging is one of the most fundamental tasks for text processing, this paper presented the results of a comparison of two popular POS modelling techniques, the CRF and the HMM models. The state-of-the-art CRF tagger, part of the Gate package, was first tested on the TPOS corpus which was used for its development and then cross-validated with a different corpus, ARK. Both of these corpora was similarly used to test a HMM model, the basic form of which is part of the Lingpipe package. The modified HMM model together with the all the data in the appropriate formats has been made available at (URL removed and can be supplied as supplementary file) for research use.

The token accuracy for the HMM model was found to be 8% below the CRF model, but the sentence accuracy for both the models was very close, approximately 25%. The cross validation results for both the models showed a degradation of approximately 25% for tokens and a very drastic drop of approximately 80% at the sentence level. The degradation in accuracy across the two corpora implies that the two datasets had slightly different characteristics hence a model trained on one set had impaired performance on the other set. The comparative performance of the HMM and the CRF models show that the CRF model is marginally better, however the HMM model learns orders of magnitude faster and is easily adaptable to progressive learning applications. Hence, is better suited to real time applications such as processing live tweets and streaming texts.

### References

Andrew Y Ng MIJ (2001) On Discriminative vs. Generative classifiers: A comparison of logistic regression and naive Bayes


Component Modeling for SCADA Network Mapping

Ayodeji J. Akande Colin Fidge Ernest Foo

School of Electrical Engineering and Computer Science Queensland University of Technology Australia,
GPO Box 2434 Brisbane, QLD 4001,
Email: {ayodejiames.akande, c.fidge, e.foo}@qut.edu.au

Abstract

Supervisory Control and Data Acquisition systems (SCADA) are widely used to control critical infrastructure automatically. Capturing and analyzing packet-level traffic flowing through such a network is an essential requirement for problems such as legacy network mapping and fault detection. Within the framework of captured network traffic, we present a simple modeling technique, which supports the mapping of the SCADA network topology via traffic monitoring. By characterizing atomic network components in terms of their input-output topology and the relationship between their data traffic logs, we show that these modeling primitives have good compositional behaviour, which allows complex networks to be modeled. Finally, the predictions generated by our model are found to be in good agreement with experimentally obtained traffic.

Keywords: Supervisory Control and Data Acquisition (SCADA) Topology, Network mapping, Modeling, Network traffic analysis

1 Introduction

Supervisory Control And Data Acquisition (SCADA) systems are used in industrial environments to manage and control critical infrastructure such as transport systems, telecommunications, power and energy services. These systems were invented to handle communication over a large geographical distance and multiple sites. Previously, they were designed to operate in an isolated environment using proprietary protocols which ensured some level of security by obscurity. However, in recent years, these systems have evolved from being stand-alone networks and are now interconnected with both enterprise networks and the Internet with the use of Transmission Control Protocols (TCP)/Internet Control Protocol (IP) leading to wider networks. Though this has led to easy manageability, and improvements in functionality and productivity, connecting SCADA networks to the Internet and corporate networks exposes the systems to attacks with possible ruinous effects.

SCADA networks have some distinct behaviours when compared to “traditional” Information Technology (IT) networks. SCADA networks are expected to be more stable over time with fewer or no devices added to or frequently leaving the network; and in contrast to a traditional network, SCADA networks support fewer services (Barbosa et al. 2012b). SCADA network has a well-defined structure with a predictable traffic behaviour as a result of its fixed number of network devices, regular network communication patterns and limited number of protocols (Barbosa & Pras 2010).

However, increased connectivity between SCADA and other networks can lead to instability in SCADA network communication behaviour. Nicholson et al. (2012) argued that the main cause of security vulnerabilities in SCADA systems seems to be the increased connectivity and the loss of separation between SCADA and other parts of organizations’ IT infrastructures. It is crucial to have up-to-date knowledge of the communications topology of SCADA networks to ensure stability of the network. This can be achieved via network mapping.

Network mapping is the study of network connectivity. This is very useful to industries with a large dispersed network. Several published works (Besaw et al. 1994, Breitbart et al. 2000, Govindan & Tangmunarunkit 2000) have shown the use of network mapping in the study of Internet structure, detecting shared bottlenecks and detecting network intrusions using various techniques including development of algorithms for discovering the network topology (Breitbart et al. 2000) and use of statistical analysis of network packet characteristics (Bykova et al. 2001). However, this previous research is limited to standard IT infrastructure.

The contribution of our work is a new modeling technique to detect whether or not a SCADA network’s traffic is consistent with its anticipated topology. This aids in checking network misconfiguration, legacy network mapping, maintenance of growing/evolving networks, monitoring for system failure and monitoring for intrusions. In this paper, we demonstrate a modeling process to assist SCADA network mapping via network traffic analysis. Our topological modeling technique defines network modeling primitives which can be combined to create complete network traffic flow models. This technique can be used to confirm whether or not a network topology has been changed. This work investigates top-down data flow in SCADA networks and we explore a set-theory approach to developing the logical relationships needed to describe the data flow in a network. We exploit the rigid hierarchical structure of SCADA systems to produce a simple modeling approach. This is as a result of fixed number of network devices, regular network communication patterns and a limited number of protocols (Barbosa & Pras 2010).

This paper is divided into four sections. Section 1 introduces the work and its motivation. Section 2 examines related work on network traffic flow, network discovery...
and SCADA traffic monitoring. In Section 3, we describe our network modeling primitives used for modeling and mapping network topologies and we give descriptions of the compositional behavior of the modeling primitives and how these primitives are connected to form topologies. Practical case studies are presented in Section 4 and Section 5 provides the discussion. Finally, the conclusion is discussed in Section 6.

2 Related Work

Studying the structure and characteristics of large networks, especially the Internet, has drawn the attention of many researchers. Though there are challenges in monitoring network traffic, Lakshina et al. (2004) explained that one way of addressing this problem is by recognizing that observed traffic on different links of a network is not independent. This paper stated that a direct and fundamental way of studying network traffic is by analyzing the network’s set of Origin-Destination flows which indicates all traffic entering at a specific point and exiting the network at some other points. This concept is a starting point for our work in network traffic analysis.

Bykova et al. (2001) described how statistical analysis of network packet characteristics can be used in detecting network intrusions. This paper tried to identify how much information can be deduced about an attack by checking the packet headers but not their contents. The paper did not address how to detect inconsistencies in a network topology as a result of configuration errors, routing error or improper documentation. By contrast, our approach can detect such inconsistencies in a network topology.

In a related work, Barbosa et al. (2012a) provided a view into the characteristics of SCADA network traffic. This paper showed the similarity between SCADA traffic and Simple Network Messaging Protocol (SNMP) traffic as a regular time series which “presents baseline changes at seemingly arbitrary time intervals” (Barbosa et al. 2012a) because the majority of the sources generate data in a periodical fashion.

In another work by Barbosa et al. (2012b), they verified whether or not the models used in describing traditional network traffic can be used in SCADA traffic. The research was based on a list of network traffic invariants such as diurnal patterns of patterns, self-similarity, long-normal connection sizes and heavy-tail distributions. They compared characteristics of SCADA traffic traces analysis with traditional network traffic and they concluded that the existing traffic models can not be easily applied to SCADA traffic.

SCADA networks were designed to be reliable and fail-safe, but attacks or security breaches over the past decade indicate that risks of deliberate attacks on the systems have not been adequately considered (Nicholson et al. 2012). There are numerous sources of threats to SCADA such as human errors, equipment failures, natural disasters, terrorist groups, hostile governments and industrial spies. Most of the attacks on these systems require studying and identifying vulnerabilities in the network topology which can be detected by monitoring the network topology. Monitoring a network, especially a time critical network such as a SCADA network, is a difficult and a demanding task. It has been a difficult objective to perform whole network traffic analysis, that is, monitoring every point in the network and modeling it.

To detect any deviation from the intended or expected communications topology of a SCADA system, which may indicate that the system has been compromised, network traffic analysis is required to study the “normal” data flow and generate a model topology, thus necessitating a modeling process. In our research, we show how to model SCADA network components and compose these primitives to predict expected traffic patterns.

3 SCADA Network Modeling and Mapping

SCADA networks consist of components such as Human Machine Interfaces (HMI), Remote Telemetry Units (RTU), Programmable Logic Controllers (PLC) and other devices (see Figure 1), connected hierarchically (in a tree-like structure) to achieve scalability. Data communications in SCADA systems are based on a master/slave (server-client) architecture (Igure et al. 2006, Daneels & Salter 1999). A SCADA control unit/master station communicates with field devices via communication channels such as fiber networks, public switch telephone networks (PSTNs), satellite links and even Ethernet. The control unit transmits SCADA messages to the field sites using SCADA protocols, but the field sites do not interact directly with each other. These features distinguish SCADA systems from general-purpose IT systems.

Observing network traffic assists in learning about a network topology and its behaviour. Our modeling and mapping approach for SCADA network topology discovery begins by modeling primitive components and combining them to predict what traffic should pass each point in the network. The run time analysis process then entails matching captured packets at both ends of a network connection to compare with the model to detect inconsistencies.

In this paper, we study the network traffic behaviour assuming perfect communications in the network, however, future work will be the application of the traffic modeling process described in this paper to lossy communications as well as considering SCADA network analysis in real time. Our models focus on the communication path from the HMI to the PLC/RTU.

3.1 Modeling technique

We developed our modeling technique using set-theoretical relations to express the expected network behaviour as deduced from the physical topology. This section presents the various modeling primitives and shows their compositional behaviour.

3.1.1 Modeling Primitives

We model network components in terms of their input-output traffic flow by expressing the expected relationship between the set of packets entering the device and those
Figure 2: Atomic components for SCADA network modeling

This is a result of switching according to the addressed destination (but the switching rules are not modeled here). This is defined by:

\[ A = B \cup C \]  
\[ B \cap C = \emptyset \]  

Definition 4 (Filter) This is a component in which messages from the input interface are filtered depending on configuration-specific rules as shown by the pattern labeled F in Figure 2. The filtering is defined based on user-defined policies (although these do not form part of our model). Security networking devices such as firewalls and intrusion prevention systems demonstrate this kind of behaviour. This is defined by:

\[ A \supseteq B \]  

Definition 5 (Merger) This is a component with multiple inputs and one output as shown by the pattern labeled M in Figure 2. Messages from multiple inputs are combined and sent to another interface as an output. This is defined by:

\[ A \cup B = C \]  
\[ A \cap B = \emptyset \]  

The log of packets captured at any interface is regarded as the elements of the interface. A set is defined as a collection of unique objects, hence, every packet observed at each interface must be uniquely identifiable.

3.1.3 Primitives Interconnectivity

The atomic components above are used to express the expected behavior of a network. This requires that some of these atomic components can be linked together to form a network topology. For instance, a filter component can be combined with a buffer and their respective input-output relationships can be merged to create a new compound network component. To achieve this, the steps are:

\[ A = B \]  

Definition 1 (Buffer/Forward) This is a component with input equal to its output as shown in the pattern labeled B in Figure 2. Messages are transferred from the input interface to the output interface without any alteration or changes, so the set of entries at A is equal to the set of entries at interface B. This expresses the behavior of networking devices such as simple repeaters and it can be defined by:

\[ A = B \]  

Definition 2 (Broadcaster) This is a component with one input and multiple identical output interfaces as shown by the pattern labeled R in Figure 2. A broadcaster receives messages and the messages are repeated at all the output interfaces. This is defined by:

\[ A = B = C \]  

Definition 3 (Switch) This is a component having one input and multiple output interfaces where a choice is being made between the outputs as shown by the pattern labeled S in Figure 2. When a host sends packets through the device, the packets received at B and C are not the same.

3.1.2 Atomic Components Behavior

The behavior of atomic components expresses the expected data flow through them and can be illustrated using set-theoretical relationships between various interfaces where network traffic could potentially be monitored. In Figure 2, letters A, B, C, etc. represent the set of all packets passing a particular point in the network. The behaviours for various components are defined as follows:

Proceedings of the 38th Australasian Computer Science Conference (ACSC 2015), Sydney, Australia, 27 - 30 January 2015
Step 1: Connection of required components
This first step entails linking required components together by joining their "connectors". Connectors are represented with a dot (•) in Figure 2 and they are the observation points where packets are assumed to be captured. Figure 3 shows how two components, a switch and a buffer, are connected by joining their connectors to form a simple network topology.

Figure 3: Interconnectivity of atomic components

Step 2: Unification of the set variables
This step requires unifying the set variables of the two connected components. It is important to rename the set variables in the set equations to eliminate ambiguities. The diagram in Figure 4 shows how the variables are renamed and unified in this case.

Figure 4: Unification of set variables and renaming

Step 3: Simplification of resulting set equations
This last step involves simplifying the resulting set of equations to identify the relationships between the intended "observation points" in the network. Often this will show only end-to-end input and output points, but sometimes we will also allow for the possibility of adding a "tap" in the middle. An example of inferring the combined behaviour from the two component patterns is shown below in relation to Figure 4.

Pattern 1:
\[ A = B \cup C \quad (8) \]
\[ B \cap C = \emptyset \quad (9) \]

Pattern 2:
\[ B = D \quad (10) \]

Summary:
\[ A = D \cup C \quad (11) \]
\[ D \cap C = \emptyset \quad (12) \]

The set-theoretical relationships can be derived from the physical topology of the network and possibly other network documentation that reflect all necessary components in the network. The result describes the expected communications pattern of the network which will assist in network mapping and to verify if the actual network traffic behaviour is consistent with the assumed topology.

3.2 Network Analysis Process
Our SCADA Network Analysis process can be divided into four main procedures. The first procedure starts with developing a set of component models as described in Section 3.1.3 from physical topology diagrams and/or network documentation of the network to create the design model which expresses the expected communication topology.

The second procedure is capturing network packets from various points in the network. Network traffic is monitored at several possible points. The third procedure is analyzing the traffic matches that predicted by the design model. This process includes comparing the captured packets at the input interface with the captured packets at the output interfaces. However, a unique packet identifier is required to uniquely match captured packets between the sender and various destinations. This could be one or a combination of fields in the packet. An Internet Protocol (IP) datagram consists of a packet header and data. The packet header contains fields such as source IP addresses, destination IP addresses, protocol, identification, Type of Service, header length, header checksum, time to live, among others used in routing, fragmentation and defragmentation of packets. In a network, each IP datagram is assigned a unique value in the identification field of every packet and all fragments of the datagram have the same identification number. The identification field has 16 bits which get reused. Thus, to uniquely identify a packet in a large capture, more fields of the packet header will be required. For this paper, the combination of source IP address, destination IP address and packet IP identification numbers is used as the unique packet identifier.

Having identified the packets, the next procedure is to compare the result against the packet set relationships in the derived model to verify whether or not the network traffic behaviour is consistent with the anticipated traffic behaviour. The derived logic expresses the expected communications behaviour of the network and any deviation from it proves that the network has been compromised.

4 Experimental Set-up
As a proof of concept, two experiments were conducted, a simple network using a SCADA testbed (simulating a water treatment and distribution system) and a more complex network in a simulated environment using a graphical network simulator (GNS3). Under each experiment, two tests were conducted and the following are the assumptions made:
no loss of packets;
- data flow from top-down, that is from the HMI to the physical devices via PLCs;
- all packets are uniquely identifiable; and
- network traffic can potentially be monitored at every possible point in the network.

The first test for each experiment is to show how the model can be used to confirm that the network is consistent with the anticipated topology as stated in the network documentation while the other test shows how the model can be used to detect a change in the network.

This section describes the experimental set-up and the application of the network modeling and mapping techniques as stated in this paper.

4.1 Experiment 1

The first experimental set-up consists of two machines (a Controller and an attacker) and two PLCs. The controller provides both the HMI and Master Telemetry Units (MTU) functionality implemented using a LabView application while the attacker runs a Transmission Control Protocol (TCP) Modicon Communication Bus (Modbus) hacker program. The network includes a conventional Ethernet network as the data communications network and Modbus/TCP was utilized as the SCADA protocol. The target PLCs using a National Instruments Compact RIO provide the remote terminal unit (RTU). The controller communicates with the PLCs using a master/slave technique which is the architecture Modbus/TCP is based on. Each PLC controls a process setup shown in Figure 5 which comprises a water tank controller as may be found in a chemical plant. For this experiment, two tests were conducted; the first under a perfect environment while the other was in a compromised condition, to generate normal and attack traffic.

4.1.1 Methodology

Each test was conducted over a period of 180 seconds. The controller and attacker sent messages to the PLCs controlling the water pump and packets were generated.

For the first test, the controller attempted to turn the pump on and off at 30 second intervals while the attacker remained dormant. Figure 6 depicts the physical topology of the network and also shown in Figure 7 is a network pattern showing the expected communications topology constructed using our primitives.

Adopting the steps described in Section 3.1.3, a model for the network was derived using the physical topology to confirm if the predicted packet relationships conform to the traffic pattern generated from the captured packets. Below is the formulated relationships derived from this model:

\[
A = B \cup C \\
B \cap C = \emptyset \\
B = D \\
C = E
\]

The above equations can further be simplified as follows:

\[
A = D \cup E \\
D \cap E = \emptyset
\]

After developing the relationships representing the expected communication topology, we then tapped the network to observe traffic and captured network packets at points A, D and E using the Wireshark application. Shown in Figure 8 is a sample of captured packets at input interface A between the HMI.

The next step taken was the analysis of network packets. The captured packets were analyzed using the following steps:

- Extraction of required fields from the captured network packets such as source IP address, destination
IP address and identification number. This forms the unique packet identifier to be used in matching packets between the sender and receivers;

- Packets captured at the sender interface were matched with the captured packets at the receivers’ (PLC 1 and PLC 2) interfaces. The result represented the number of matched and unmatched packets.

A sample of a packet analyzed showing the source IP address, destination IP address and the identification number is shown in Figure 9.

The second test was conducted with the presence of an attacker. The attacker attempted to disrupt the control of the pump by using the TCP Modbus Hacker program. For the second test, the Attacker lays dormant in the time period up to 60 seconds and in the time period of 61-90 seconds, the Attacker floods PLC 1 by performing the opposite actions of the Controller in the same period. This test was performed to show how our model can be used to detect deviations from the intended communications topology in the SCADA network. All generated traffic for the two environments were captured and then analyzed. The following section presents the obtained results and analysis.

### 4.1.2 Experimental Results and Analysis

In the two tests conducted, the captured packets at the controller interface were uniquely matched with the ones at each of the PLCs’ interfaces using the chosen unique packet identifier and two tables were generated from the analysis. Table 1 shows the analysis of network packets captured under perfect conditions while Table 2 records the analysis of the network with the presence of an attacker.

<table>
<thead>
<tr>
<th>Interface</th>
<th>No of matched packets</th>
<th>No of unmatched packets</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>40620</td>
<td>nil</td>
</tr>
<tr>
<td>A and D</td>
<td>20311</td>
<td>nil</td>
</tr>
<tr>
<td>A and E</td>
<td>20309</td>
<td>nil</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Interface</th>
<th>No of matched packets</th>
<th>No of unmatched packets</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>40620</td>
<td>nil</td>
</tr>
<tr>
<td>A and D</td>
<td>16934</td>
<td>10054</td>
</tr>
<tr>
<td>A and E</td>
<td>20308</td>
<td>nil</td>
</tr>
</tbody>
</table>

The result of the first test as shown in Table 1 indicates the number of packets sent as messages from the controller via interface A to each of the PLCs, observed at each of the interfaces D and E. The result was compared to the packet relationships formulated from the physical topology of the network using our modeling primitives. According to the model, the log of entries at point A must be the union of entries observed at points B and C. In Table 1, all packets matched, so the topology is consistent with the expected behavior of the network.

On the other hand, the result of the second test as shown in Table 2 indicates that not all packets sent from the controller and observed at point A matched the captured packets observed at point E of PLC1. This suggests that the network traffic is not consistent with the anticipated network communications.

For further analysis, we generated a traffic model from the network for each test. This is done by matching the log of packets from the input interface with its output interface(s) using the unique packet identifier, as shown Figure 7, packets at point A are matched against output packets at points B and C, while D and E are the output of B and C respectively.

The created traffic models are compared to the design model generated from the physical topology and Table 3 presents the comparison.

<table>
<thead>
<tr>
<th>Design Model</th>
<th>Traffic Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test 1</td>
<td>Test 2</td>
</tr>
<tr>
<td>A = B ∪ C</td>
<td>A = B ∪ C</td>
</tr>
<tr>
<td>A ≠ B ∪ C</td>
<td>B = D</td>
</tr>
<tr>
<td>B = D</td>
<td>B = D</td>
</tr>
<tr>
<td>C = E</td>
<td>C = E</td>
</tr>
<tr>
<td>D = E</td>
<td>D = E</td>
</tr>
<tr>
<td>D ≠ E</td>
<td>D ≠ E</td>
</tr>
</tbody>
</table>

These results indicate that the state of the network does
not conform to the anticipated behaviour and the implications are as follows:

- This suggests that there are deviations from the intended communications topology, which means that the original network topology has been compromised.
- It also indicates that there might be an intrusion, error in routing or configuration error.

Section 5 discusses further the analysis of this experiment.

4.2 Experiment 2

In this second experiment, a more complex topology was designed in a simulated environment using a graphical network simulator (GNS3). The experimental set-up consisted of a router with firewall, switches, hubs and six workstations, and two tests were conducted. The network was divided into two network domains with two workstations (I and II) controlling each domain. Assigned to domain I were workstations I, III and IV while workstations II, V and VI formed the domain 2. The physical topology of the network is shown in Figure 10.

4.2.1 Methodology

Two tests were conducted and for the first test, the network was set up as shown in Figure 10 while for the second test, the network set-up was altered. The aim was to detect whether or not the alteration could be detected using our network modeling and mapping approach.

For clarity, the entire network is divided into four domains as indicated in Figure 10 and shown in Figure 11 is the model developed using our primitive components.

Traffic relationships were derived from the model of the network as shown below.

Group 1:

\[
\begin{align*}
A \cup B &= C \\
A \cap B &= \emptyset
\end{align*}
\]

Group 2:

\[
C \supseteq D
\]

Group 3:

\[
\begin{align*}
D &= E \cup F \\
E \cap F &= \emptyset
\end{align*}
\]

For this experiment, a firewall was configured on the router to filter outgoing packets from the sender and this implies that packets are checked to confirm if they conform to the set policy which states that any packets outside the specified network addresses should be dropped. Therefore, in a lossless communication environment, no packet is expected to be dropped, i.e.

\[
C = D
\]

To simplify the resulting set equations:

\[
\begin{align*}
A \cup B &= G \cup H \\
A \cap B &= \emptyset \\
A \cup B &= G \cup H \cup I \cup J \\
A \cap B &= \emptyset \\
A &= G \cup H \\
B &= I \cup J
\end{align*}
\]

In addition, based on our knowledge of the way the generated packets are addressed and the switches are set in this network configuration, we further introduce more specific constraints for this particular set up:

\[
\begin{align*}
A &= G \cup H \\
B &= I \cup J
\end{align*}
\]

Following the derivation of the network logic is network capturing. Network packets were captured at various input and output points; A, B, G, H, I and J, using the
Wireshark application. In GNS3, network interfaces can be tapped to monitor network traffic. 200 packets were sent simultaneously from the workstations (I and II) in group 1 as shown in Figure 10 to other workstations within their assigned domain.

For the second test, the switch connecting workstations V and VI was replaced with a hub. This was to investigate the impact of a change in a device on network communications.

The captured packets were analyzed by matching the entries at both inputs with the entries at the outputs. Presented in the following section is the experimental result and analysis.

4.2.2 Experimental Result and Analysis

For the first test, network traffic was observed at interfaces A, B, G, I, H and J, and a unique packet identifier which is the combination of the source IP address, destination IP address and identification number, was used to match packets between the senders and the receivers. Packets sent from workstation I as observed at point A were matched against captured packets at points G and H, and likewise entries at point B sent from workstation II were matched against entries at points I and J. Tables 4 and 5 reveal the results of the matched packets for both domains.

Table 4: Experiment 2, the analysis of domain 1 for test 1

<table>
<thead>
<tr>
<th>Interface</th>
<th>No of matched packets</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>200</td>
</tr>
<tr>
<td>A and G</td>
<td>100</td>
</tr>
<tr>
<td>A and H</td>
<td>100</td>
</tr>
</tbody>
</table>

In the second test conducted, an increased number of packets were observed at points A, I and J when compared with the observed packets at points B, G and H. Table 6 shows the comparison between the observed packets at points A and B. Also observed at point B, I and J were the same number of packets as shown in Table 7.

Table 5: Experiment 2, the analysis of domain 2 for test 1

<table>
<thead>
<tr>
<th>Interface</th>
<th>No of matched packets</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>200</td>
</tr>
<tr>
<td>B and I</td>
<td>100</td>
</tr>
<tr>
<td>B and J</td>
<td>100</td>
</tr>
</tbody>
</table>

Table 6: Experiment 2, the analysis of domain 1 for test 2

<table>
<thead>
<tr>
<th>Interface</th>
<th>No of packets</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>200</td>
</tr>
<tr>
<td>B</td>
<td>205</td>
</tr>
</tbody>
</table>

Table 7: Experiment 2, the analysis of domain 2 for test 2

<table>
<thead>
<tr>
<th>Interface</th>
<th>No of packets</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>205</td>
</tr>
<tr>
<td>I</td>
<td>205</td>
</tr>
<tr>
<td>J</td>
<td>205</td>
</tr>
</tbody>
</table>

Tables 4 and 5 reveal the result of the first test showing that the number of packets detected at the senders’ interfaces are equal to the sum of the number of packets detected at the receivers’ end of the connection and accords with the network configuration which was derived, \( A = G \cup H \) and \( B = I \cup J \). The network traffic analysis thus shows that the network behaviour is consistent with the anticipated topology as stated in the physical topology of the network.

For each test conducted in this experiment, a traffic model was also generated by matching each packet cap-
tured at the input point(s) against its output point(s). As
shown in Figure 6, packets at points A and B are matched
against packets at point C, and likewise other points. The
created traffic models are compared to the design model
generated from the physical topology and depicted in Ta-
ble 8 is the comparison of the models.

Table 8: Comparison between original model and traffic
models for Experiment 2

<table>
<thead>
<tr>
<th>Design Model</th>
<th>Traffic Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test 1</td>
<td>Test 2</td>
</tr>
<tr>
<td>A ∪ B = C</td>
<td>A ∪ B = C</td>
</tr>
<tr>
<td>A ∩ B = ∅</td>
<td>A ∩ B = ∅</td>
</tr>
<tr>
<td>C = D</td>
<td>C = D</td>
</tr>
<tr>
<td>D = E ∪ F</td>
<td>D = E ∪ F</td>
</tr>
<tr>
<td>E = G ∩ H</td>
<td>E = G ∩ H</td>
</tr>
<tr>
<td>F = I ∩ J</td>
<td>F = I ∩ J</td>
</tr>
</tbody>
</table>

As shown by the final row, the change of device is re-
vealed by the change in the observed traffic.

5 Discussion

The two experiments shown in this paper have demon-
strated how network traffic analysis can be used to iden-
tify whether network flow is consistent with the expected
topology. In each experiment, two tests were performed to
check how set theoretical component models can be used to
depict data flow in a network and also used in develop-
ing a model topology.

In Experiment 1, the result of the first test confirmed
the consistency of in the network’s behaviour when com-
pared with the expected communications topology ex-
pressed using our model derived from the network physi-
cal topology, while on the other hand, the second test re-
vealed the inconsistency.

The result of the second test of Experiment 1 indicated
that the set of the received packets at both output points
D and E was not equal to the observed packets at the in-
put point, that is, \(A \neq D \cup E\). This was because not all packets sent from
the HMI were observed at PLC 2 while all the packets
sent from the same HMI to PLC 1 got to their destination.
In order to detect where the changes occurred, network
traffic was observed at every point on the network. Af-
fter monitoring point C, analysis detected the presence of
an unknown source (an attacker) connected to the network
and overriding the operations of the HMI (as shown in the
model in Figure 12 where \(F\) represents the attacker). While
matching the packets from the input (A) to the output (E)
using the unique packet identifier, there were some un-
matched packets which came from an unidentified source
IP address.

In Experiment 2, the result of the first test confirmed
that the network behaviour was consistent with the antici-
pated topology. According to the derived relationships for
the network, the captured packets at the input points (A
and B) must be the union of the packets captured at the
output points (G, H, I and J) and the result showed that
this was true.

However, the result obtained from the second test was
not consistent with the anticipated behaviour. The test re-
vealed a change in the expected communications topology.
200 packets were sent from workstations I and II each but
the observed traffic at point B indicated extra packets. Fur-
ther analysis showed that the captured packets at point I
and J were the same, that is, \(I = J\), which is contrary to
the derived logic, \(I \cap J = \emptyset\). Further investigation showed
that the changes observed in the network were as a result of
the change in a network device. Every point in the net-
work was observed and analyzed to identify the cause of
the change in the network’s traffic behaviour. The analysis
discovered a behavioural change as a result of the removal
of the switch connecting workstation V and VI which was
replaced with a hub, this led to an increase in the number
of packets observed at the interfaces. A hub broadcasts
messages and this explains why common traffic was ob-
erved at points B, I and J.

We noted that our approach serves as an essential basis
for SCADA network topology analysis. Even though
using our modeling and mapping process as described in
this paper assisted in identifying the consistency of an antici-
pated network communications topology with the network
traffic, there is more that could be achieved using this ap-
proach. The following work is to be investigated in future
using our modeling and mapping process:

- SCADA network topology discovery. Listening at
every point on a network could assist in generating
a complete topology of the network but this is im-
practicable in a real network. Therefore, to monitor a
SCADA system and develop a complete topology of
the system creates a problem: What point(s) on the
network need to be observed to generate a topology
of the network without monitoring every point?

- Lossy communication analysis. In SCADA network
monitoring, lossy network communication cannot be
disregarded. In this paper, we considered perfect
communication with some assumptions but for lossy
communications, the assumptions will be weakened
to allow for packet loss.

- Automated Analysis. In this paper, we performed
our analysis manually, however, this might not scale
properly in a network with thousands of components.
In our future work, we will consider using an auto-
mated process in solving scalability issue while still
using the same methodology.

- Real-time network analysis. Our future work will in-
vestigate how our modeling techniques could be ap-
piled in analyzing a SCADA network in real time
where complete logs of network traffic are not avail-
able. This requires the development of an auto-
modeling approach for analyzing “recent” network
traffic.
6 Conclusion

In this paper, we have presented simple models to describe data flow through a SCADA network topology based on the captured network traffic. This approach can be used in SCADA network mapping due to its simple and predictable traffic behaviour. We considered various network primitives as well as the underlying input-output relationships needed for modeling a SCADA network. This work has shown how such relationships can be developed by studying the physical topology of a network and can then be used in detecting deviations in a network’s communications from the intended communications topology, indicative of intrusions or system failures.

References


An Enhanced Model for Network Flow Based Botnet Detection

Udaya Wijesinghe, Udaya Tupakula, Vijay Varadharajan
Information and Networked Systems Security Research, Department of Computing, Faculty of Science, Macquarie University, Sydney, Australia
udyaya.wijesinghe@students.mq.edu.au
udaya.tupakula@mq.edu.au
vijay.varadharajan@mq.edu.au

Abstract
The botnet is a group of hijacked computers, which are employed under command and control mechanism administered by a botmaster. Botnet evolved from IRC based centralized botnet to employing common protocols such as HTTP with decentralized architectures and then peer-to-peer designs. As Botnets have become more sophisticated, the need for advanced techniques and research against botnets has grown. In this paper, we propose techniques to detect botnets by analysing network traffic flows. We developed templates for capturing traffic flows with more relevant attributes for botnet detection. Also we make use of the IPFIX standard for the specification of the templates. Hence our techniques can be used to detect different bot families with lesser overheads and are vendor neutral.

Keywords: botnet, network security, IPFIX, NetFlow.

1 Introduction
Malware is a tremendously serious intimidation to modern high-tech data networks. There are various types of malware. Among the malware, botnets are a clever piece of software, which carry out sophisticated synchronized activities while being resilient. The botnet is a group of hijacked computers, which are employed under command and control mechanism administered by a botmaster. Botnets are also known as a zombie army. Botnets are smart solution for cyber criminals since it makes available an effective mechanism to hide identity of the botnet herder; attempts for tracing back to origin always ends up at a hijacked device owned by innocent user that makes investigation all most vain.

The Botnets have been evolving during last few years and currently large varieties of botnet families are available. Botnet evolved from IRC based centralized botnets to employ common protocols such as HTTP with decentralized architectures and even peer-to-peer designs (Gu, Zhang & Lee 2008). Along with the advancement in protocols and architectures, botnets also used methods such as domain flux (Domain Generation Algorithms), IP flux (Single flux or Double flux) (Antonakakis et al. 2011) DNS techniques and encryption to evade detection.

As Botnets have become more sophisticated, the need for advanced techniques and research against botnets have grown. As a result of this, detecting botnet activities have become important part of modern security systems. Indeed fair amount of research has been performed during last few years in the context of botnet detection and prevention. In spite of all this effort and expense, detecting botnets remains a challenge. Furthermore, the nature of battleground is highly volatile and bot designers are developing more stealth botnet designs every day. Also proliferation of super-fast data networks, encryption technologies and anonymity techniques are making security professionals’ life harder. Botnet operators are usually financially motivated and clever enough to evade detection. Therefore, most security researchers are still struggling to find better solution even though academia and industry have conducted significant amount of research in this area. Hence, detecting botnets have become very challenging and requires an enormous effort to find sophisticated monitoring and detection methods.

IP Flow data has been used to detect various malwares including botnets in a high speed, large volume data networks. Flows are defined as group of packets, which share common characteristics such as source IP, destination IP, source port, destination port and protocol type. Each packet that is routed within the network can be examined to identify unique flows based on conjoint attributes of the packets. These attributes are considered as the IP packet identity or fingerprint of the packet and determine if the packet is distinctive or tie with other packets.

All packets with the uniform source/destination IP address and ports, protocol, interface and class of service are categorized into a flow. Subsequently number of packets and number of bytes are matched. This approach of fingerprinting of a flow is scalable as a large amount of flow information is summarized into a database. This flow information is extremely valuable for analysing malicious behaviour of high speed, high traffic volume IP networks. As we are only summarizing flow details it is relatively very small compared to actual size of data streams. Further, encryption of payload is not impacted with accuracy of flow information.

Currently most of the vendors have their own formats for capturing the traffic flows. Cisco NetFlow v5 and other parallel versions of network flows are widely used. In these versions of network flow formats, attributes, which can capture from network devices, are pre-defined; hence those network flow formats are known as fixed flow data templates. Cisco first introduced capability of selecting attributes from larger set of attributes in order to define...
customised data templates in NetFlow v9 known as flexible NetFlow. Recently IPFIX (IETF 2007) (Quittek et al. 2008) has been developed to make common standard for IP flows. IPFIX address transport protocol, security, IETF and vendor specific information elements in addition to the principles of NetFlow v9 (Cisco Inc 2012) such as separate templates and records. One of the main concerns in IPFIX is to use congestion-aware protocol in order to transfer data. So Stream Control Transport Protocol (SCTP, RFC 2960) and Stream Control Transport Protocol – Partially Reliable (SCTP-PR, RFC 3760) have defined to export data to the collectors while facilitating conventional protocols; TCP or UDP can be used to transport data however SCTP-PR is preferred.

Our Contribution: In this paper, we propose a methodology to enhance botnet detection techniques by leveraging features of flexible network flows. Our approach consists of two main steps. Firstly, we identify suitable data set templates with more relevant attributes for botnet detection from IP flows, using IPFIX. Also we customize each template according to various botnet families and attributes to enhance the effectiveness of the system. This customization overcomes the limitation of fixed flow templates in the previous approaches. Further, we improve effectiveness of existing network flow based botnet detection by above templates while significantly reducing size of the dataset. Secondly, we use IP flow data to detect botnet behaviours in unlabelled traffic. To best of our knowledge, this is a novel approach and it enhances existing IP flow based botnet detection research.

Our paper makes following major contributions;

- We develop a general network flow based botnet detection framework that is based on flexible IP flows (IPFIX) in order to overcome limitations of current network flow based botnet detection systems.

- We create a generic IP flow template that is flexible for use with existing network flow based botnet detection systems. This results in minimising the dataset overhead for capturing the flow while keeping all the necessary attributes for detecting the bots.

Our work is organized as follows. In Section 2 we propose IPFIX standards based techniques for detecting attacks from a range of bot families. Section 3 presents the implementation and analysis of our model. We will also compare our model with some of the recently proposed techniques. Section 4 presents some of the important related work and Section 5 concludes.

2 Our Model

In this section we will present high-level overview of our architecture and then describe the architecture components in detail.

2.1 High level Overview

Figure 1 illustrates the high-level architecture of our model. Generic templates, Flow collector, Filtering Engine and Botnet detection engine are some of the important components in our model. Generic templates are used for capturing flow information using IPFIX. Flow collector is a centralised server that is used for storing and organising the data captured at different devices by using the generic templates. Filtering is used to reduce dataset by filtering out unwanted data that is not related to botnets. Finally botnet detection engine, correlates flow information using machine learning techniques to find the pattern and detect bots. We will discuss each of component in detail in the section 2.2

![Figure 1: High-level system overview](image)

We use offline analysis for analysing the behaviour of different bot families. We make use of virtualisation techniques to generate traffic related to the bot or use publicly available dataset related to the bot. For example, we create a test bed with several virtual machines and infect some of the virtual machines with the bot and monitor the behaviour of the infected machine. Then we use machine-learning techniques for analysing the behaviour of the bot and identify the specific features that can be used to detect the bots. We repeat the process for different bot families and identify the specific features that can be used for detecting the bots.

We have used different machine learning techniques for analysing the communication of different bot families and made some interesting observations.

As a pre-programmed system, botnet runs automatic algorithms throughout their life cycle in order to conduct different activities. Even though botnet designer makes significant effort to randomize their activities, our analysis confirms that there are still patterns to identify them.

- Some of the bot families such as IRC-based, HTTP-based and DNS-based have specific characteristics when they react to the C&C instructions and how bots react to those instructions

- Bots such as SpyEye and some versions of Zeus have a unique flow behaviour that can be used as signature for detecting the bots.

- The communication in peer-to-peer bots such as, queries for update and instruction create many small uniformed packets than other legitimate P2P communications. Also the initial exchanges of packets have a fixed format and can be easily differentiated from legitimate traffic.

After analysing a range of bots, we develop a generic template by combining all the unique features of network flows for detecting different bot families. For example, features such as source address and destination address are common for detecting any bot. Hence our generic
template includes relevant features for detecting different bot families. Also note, if we are interested in detecting specific bots, we only need to use the specific features to detect that bot. There is no need to develop a generic template for such cases. Now let us consider how the generic template can be used to detect different types of bots.

We make use of the IPFIX standard for defining the generic template. In the current state of art, different vendors have proprietary protocols such as Cisco-NetFlow (Cisco Inc 2012), Juniper-j-flow or cflowd (Juniper Networks Inc 2011), Huawei-Netstream (Huawei Technologies Co Ltd 2012) for capturing the flow traffic. Hence by defining the generic template using IPFIX, it can be applied on the devices that belong to different vendors.

The generic templates are applied to different devices such as layer 3 switches and routers. These devices analyse the traffic flowing through them and report the traffic information to the flow collector. Filtering is used to minimise the captured dataset by eliminating redundancies and unwanted traffic that is not related to botnets. For example, core routers can report the network flow information of communication between internal hosts and external hosts. Gateway routers can report network flow information of external hosts. Hence we only store the internal host communication information reported by the core routers and the external host communication reported by the gateway routers. Botnet Detection Engine is used for analysing the filtered traffic and detecting bots. Since generic templates are used for capturing the flow information, the captured data includes data for detecting different bots. Hence different bots can be detected by analysing subset of the features captured in the network traffic flows. The machine learning techniques that were initially used for identifying the specific features in each bot are again used for detecting the attacks from the flow dataset.

We have compared our model with some of the related techniques. There are several advantages with our approach. Our approach has minimal dataset overhead since the templates are developed from the attributes that are extracted using machine-learning techniques for efficient detection of the bots. Our model can detect a range of botnet families using various activities throughout their lifecycle including C&C interactions, recruiting new bot members and synchronized attacks. In Section 3.3, we provide a detail comparison of our model with related techniques.

2.2 Architecture Components

In this Section we will describe the components of our framework.

2.2.1 Generic templates

Templates are used for capturing network traffic flows from different devices for detecting the bots. In this paper we consider IPv4 and IPv6 traffic flows. Flows are defined as group of IP packets, which share common, attributes such as source IP, destination IP, source port, destination port, protocols type, class of service and interface of the network element. Based on these attributes each packet going through a network element can be classified into unique flows. For example, these attributes enable us to determine if the received packet belongs to an existing flow or a new flow.

Various network device vendors have developed their own IP flow capturing techniques such as, Cisco NetFlow, Juniper-j-flow or cflowd, and Huawei-Netstream. However such flow classification schemes have several limitations since they use proprietary protocols, have fixed format for flow classification and are not compatible with products from different vendors. Hence the related techniques such as Botfinder (Tegeler et al. 2012), Disclosure (Bilge et al. 2012), BotSniffer (Gu, Zhang & Lee 2008), BotTrack (Francois et al. 2011), (Zhao et al. 2012), (Zang et al. 2011), (Strayer et al. 2008), which make use of these tools for flow analysis, are not efficient.

The fixed nature of capturing IP flows limited the capability of security analysis while avoiding important features from IP flow data. For example, conventional fixed IP flows only report 20 attributes: Source IP address, Destination IP address, IP address of the next hop router, SNMP index of the input interface, SNMP index of the output interface, packet in the flow, total number of L3 bytes in the flow’s packets, system uptime at start of the flow, system uptime at the last packet of the flow received, layer 4 source port number, layer 4 destination port number, cumulative OR of TCP flag, IP protocol, IP type of service, AS number of the source, AS number of the destination, source address prefix mask bits, destination address prefix mask bits, and two unused attributes (pad 1 and pad 2).

The fixed attributes consume 48 bytes for each of the flow record. From our analysis, we identified that most of the attributes that are used in the fixed flow templates for classification of the flows are not useful for detecting the bots. Furthermore, there is no way to include new attributes that can be used for efficient classification of the flows for bot detection. Hence the main disadvantages of using fixed IP flow is that it narrows down the security analysis that can be performed on the captured traffic and it also results in unnecessarily increasing the size of data set.

Recently, IETF introduced open standard for flexible IP flows named as IPFIX. This IP fixed abstracts most of the features from NetFlow version 9 to define standard for IP flows. However unlike Netflow version 9, IPFIX added variable length fields. Furthermore, IPFIX standard allow users to select wider range of attributes from more than 400 different data points (RFC7012, 2013). Hence IPFIX enables the users to define the templates with any of the attributes for traffic classification.

Our system makes use of IPFIX for developing customized templates for detecting botnets. There are two main benefits with the customized IP flow templates used in our model. First, customized templates facilitate to reduce size of the data set by removing unwanted attributes from fixed data set. This is a very important benefit of our proposed system, as this will significantly reduce the size of data sets for analysis and save space and computing power. Secondly, proposed templates...
provide more room for botnet detection since new templates come up with additional important attributes in addition to traditional network flow data set. For example, customized templates allow us to develop new data templates based on botnet families, adding new features from IP header, adding new features from pay load (DNS attributes), adding new statistical attributes (average pay load size) in order to detect botnets. Further, this will facilitates to detect DGA and fast flux attacks by introducing DNS attributes in to network flow data set.

Recall that we have used machine-learning techniques for identifying the attributes that are efficient for detecting the bots. We have used these attributes for developing generic templates by using only the attributes that are efficient for detecting different bot families. Once customized templates have been developed, it is applied to different network devices. Since we make use of IPFIX for developing the template, it can be used on devices that belong to different vendors. Now the network devices classify the flows based on the attributes specified in the template and send this information to the flow collector.

2.2.2 Flow collector

The flow collector is used to collect the flow records from different devices. As flexible IP flow comes with variable size filed set and data set, flow collector enables storing according to the requirement of flow templates. The network devices send their data along with data templates to the flow collector. The flow collector software facilitates the security administrators to define storage template to match with the flow templates for storing the data. Then collector builds a raw dataset based on information from templates, by storing received data in correct order. Data is arranged into directories based on type of traffic, flow direction, year, month, date and time. Flow collector also provides additional features such as compression of the data to save the storage space and ability to perform search on the stored data. Another important feature of flow collector is to facilitate usage of various analytical tools on the stored data to eliminate unwanted traffic and detect botnets. This provides more room to use different filtering and analytical tools to access data and make data available for various analysis methods.

2.2.3 Filtering engine

The main purpose of the Filtering Engine is to reduce data set since generic templates can also capture the flows that are not related to botnet. Hence before forwarding the collected flows to the next stage for further processing and detection of attacks, we reduce the dataset by removing unnecessary data stored in the flow collector. This will make data set more manageable and save computing power in next steps while providing capability to deal with high volume data networks. Similar to (Strayer et al. 2008) we use a five step filtering mechanism for botnet detection system. However there are also some differences in the filtering used in our model. Below we summarise how our model differs to Strayer.

Firstly, Strayer mainly targets detection of IRC based botnet. Hence his technique makes use of only the TCP flows for the detection of botnet and filters out all other traffic. Our model targets a range of bot families. Hence we make use of TCP and UDP flows for the detection of bots. For example, significant amount of botnets manipulate legitimate DNS traffic to tunnel C&C communication. So filtering out of non-TCP flows is not suitable for DNS based botnet families.

Secondly, Strayer uses a filter to remove the nuisance port-scanning chaff in order to reduce dataset. The main idea behind this is if a flow contains only SYN or RST flag that means TCP flow is not established and un-established flows are no value for C&C communication. However, un-established flows represent some botnet activity such as DDOS or random packet throwing to mislead detection tools. So instead of filtering out un-established flow we categorize flows with only SYN or RST flags in order to detect botnets. Further, flexible IP flows allow us to customize flow template to collect flow data based on TCP flags while fixed flows only allow us to collect cumulative TCP flags.

Thirdly, Strayer introduced a filter to remove high volume data flows as C&C communication is not supposed to generate high bitrate data flows. Also software updates and peer-to-peer data transfer generate high volume data flows. Hence we use similar filter to distinct legitimate peer-to-peer traffic from P2P or hybrid botnets’ C&C channels.

Fourthly, Strayer uses filter to remove flows with large IP packets specifically packets size above 300 bytes. This filter can only be used for detecting IRC botnets. However some botnets use large data packets to send stolen data to its C&C server. Hence we consider all the packets for our analysis. However, if specific pattern exists for some of the bots we modify this filter in order to identify such a communication by selecting correct size of the packets. Further, usage of IPFIX in our model allows us to get payload and header size of the packets separately. We use this attribute for fine-tuning of the filter.

Finally, Strayer used filter to remove flows with 2 or less packets or very brief time windows to remove port-scanning flows. In our model we make use of this filter to identify vulnerability or open port scanning by the botnets to find new victims.

Another important feature in our model is to define additional filters for C&C traffic behaviours of different botnet families. Such filtering enables to filter flows related to particular botnet family and facilitate further analysis effectively as it will significantly reduce data set.

2.2.4 Botnet Detection Engine

Once data set has been selected for comprehensive analysis by series of filters describe in previous stage, data set is fed to botnet detection engine. First step of the engine is to classify or cluster the flows in data set in order to identify similarities of the flows. The machine learning techniques, which are initially used to identify the specific features for each bot, are used in this stage to detect the specific bots. There are three main behavioural patterns; bot behaviour, botnet behaviour and temporal behaviour are used in our system.
In the bot behaviour, we analyse flows generated from one bot or single machine to identify its C&C communication or attack graphs. In botnet behaviour, we analysis flows generated by group of bots or machines, in order to detect botnet activities. Further, in above method we horizontally analyse flows generated in a network to find suspicious pattern related to botnet communication. Finally, in temporal or vertical method, we analysis flows generated by bots or botnets over the period of time in order to detect patterns. Once we identify, similarities or correlations, then we compare them with the patterns we already identified through machine learning.

We have analysed a range of bot families such as IRC, HTTP, peer-to-peer and hybrid bots and made several observations that enable for the detection of bots. For example, in case of C&C bots our analysis was mainly focused on detecting the bots during different phases in botnet life cycle (Feily, Shahrestani & Ramadass 2009); initial infection, secondary infection, connection, malicious command and control, and update and maintenance. In the initial infection phase attacker exploits vulnerabilities on victims and gains basic control over victims. Then secondary infection phase is used to further download and install malicious script and binaries to get full control of the victim. Once secondary infection is complete, bots make connection to its C&C server in order to become a member of botnet. Then bot will receive command and control from C&C server to conduct malicious coordinated activities. Finally bots update its binaries to get more functionality or evade detection. Below we summarise some of the important observations from that analysis of range of bot families:

In case of direct C&C related bots such as IRC and HTTP every bot needs to find its own C&C controller to be a member of centralized botnet. In case of peer-to-peer and Hybrid bots such as Kademila, Chord and GameOver (Zeus v3) each bots need to find its servant bot or proxy bot to get C&C instruction and become a member of P2P botnet. Our analysis confirms that these botnets are using hard coded static IP lists or/and DNS service to locate its C&C server, from which it has to receive the control commands and updates. This generates a pattern, which is used for the detection of bots. Conversely, to evade from discovery and close it down, the C&C servers’ use distinctive methods like Domain flux (Domain Generation Algorithm - DGA), and IP flux to alter their DNS name or the IP addresses connected with FQDN. However, this makes the botnets to connect to different C&C servers instead of connecting to a specific one. As a result, the bots perform a large number of DNS lookups and scan a large volume of addresses to find the C&C server. These patterns are used in our model to track the botnet.

The second observation related to C&C bots is that the bots need to perform frequent communication with the C&C server. This is essential to keep control and update the botnet by the bot master. Once C&C server has been discovered, the bots take updates and commands from the botmaster about what type of action has to be executed. The action can be whatsoever from sending spam emails to intensifying a DDoS attack. Another important massage type is keep alive packets that are sent from the bot to the C&C server. Moreover, some bots (such as Zeus and SpyEye) report back to C&C server with the information they steal from compromised computers. Even though botnet designers are working hard to randomize those communications to evade detection there are some vertical or/and horizontal correlation on their communication. For example, some of the Zeus bots (Zeus v1.3) send updates at fixed interval of 20 minutes.

All the botnets desire to spread over its network and recruit more bots into its botnet, which ultimately benefits to surge the strength of a botnet and consequently that of the attack carried out too. Hence the bots scan for other machines in its network for vulnerabilities. If a vulnerable machine is found, they will run exploits to compromise the machine. When scanning the network for possible machine to infect, bots generate a burst of small packets. So this activity makes a sudden increase in the number of packets without a major increase in the traffic volume that could be used to detect bots.

![Figure 2: Theoretical framework](image)

The bot detection engine also looks for DDoS activities such as, outbound TCP SYN packets having an invalid source IP address. The reason for these large number of TCP SYN packets could mean that some of the internal hosts in the network is part of a botnet and are participating in a DDoS attack. Some botnets such as (Grum, Bobax, Cutwail and Donbot) generate email spamming. Email spamming involves sending enormous amount of spam emails advertising fake products intended at financial gains. When the hosts in a network are part of a botnet involved with spamming, they send huge number of emails to the outside world and mostly using some external email server. So, unusual SMTP activity from the network to the outside is another significant network activity that is used for tracking the bots.
3 Implementation and Analysis

Our approach consists of two main steps. Firstly, we make an effort to identify best suitable template in order to collect flow data set. As there is no publicly available IPFIX dataset that can use for our study, we have to setup our own controlled lab environment to get a flexible IP flow dataset that includes malicious traffic sample. In such a case we obtain two separate datasets, dataset from analysis of different botnets in our lab environment and dataset from University network over a period of two days (about 5 GB), which contains everyday usage traffic patterns. We merged both the datasets using TCP replay tool to produce test dataset. We have used 10-fold cross-validation technique for training and testing of our model and achieved high detection accuracy.

We have analysed different bot families such as HTTP, IRC, and peer-to-peer bots to find best suitable template. First we have used a template with 57 attributes (see Figure 2) for capturing the traffic flows of each bot then used machine learning techniques to identify the best attributes for detecting each bot family. Theoretical framework of the system is shown in Figure 2.

We have used several machine learning techniques such as Bayesian Network, Neural Network, Support vector Machine, Gaussian and Nearest Neighbour classifier for identifying the best attributes for each bot since none of the machine learning technique was found to be effective for detecting all the bot families. For example, Decision Tree classifier is found to be effective for detecting peer-to-peer botnet by analysing the flow intervals. SVM and Bayesian networks were found to be effective for detecting C&C communication of the centralized botnets such as HTTP and IRC botnets.

3.1 Lab Environment

Figure 3 and Figure 4 illustrate the design of the lab, which has been used to analyse different botnet samples and collect IPFIX data sets. In this case, since Cisco devices were used in the lab environment, we made use of NetFlow 9 that is compatible with IPFIX standard.

3.1.1 Building a Flow Collector/Analyser using SiLK toolset.

We make use of SiLK for the implementation of the flow collector and used Python for developing tools to analyse the data stored in the SiLK server. SiLK is part of the Network Situational Awareness (NetSA) developed by the team of security professional at Software Engineering Institute (SEI) CERT division (NetSA 2014). The NetSA group constructs unified, standards compliant flow collection and analysis tools with open source license agreement. SiLK stands for the “System for Internet-Level Knowledge” and is cooperated of network traffic analysis tools divided into two main categories – receiving and packing flow data, and analysis. It supports efficient collection of network flow data, storage, and analysis, which lets us to very speedily query large amount of traffic data sets. We have built a SiLK v3.8.2 Collector/Analyzer on Fedora Linux v20.

The SiLK packing system collects IPFIX data and stores it in a more space efficient “packed” format binary flat file. Mainly, we have used “rflowpack” from SiLK which receives the flow record, converts the data to the SiLK Flow record format, categorises the flow records, and writes the records to hourly flat-files organised in a time-based directory structure. The Python tools are then used for querying and analysing the data stored in the server.

3.1.2 Building the Bots

As shown in Figure 3, we have run Windows XP virtual machines on Proxmox VE (KVM) hypervisor to build bots infected with the version of Zeus, SpyEye and Warbot. Below we present an overview of the bots.

Win32/Zeus: Zeus is also known as ZBot/WSNPoem, prominent HTTP based attempts first discovered in 2007 (Andriesse et al. 2013). Zeus is specially designed for stealing banking information, by using web injection technology. It outbreaks by abusing vulnerabilities in the browser security to alter web pages and manipulate financial transactions by changing or adding malicious details. Form grabbing is a technique of seizing web form data in many web browsers. Later versions of Zeus were released as commercial toolkit to build and administrate a
botnet. It consists with a control panel that is used to command and control the botnet. It also has a tool to create the bot executable for infecting different machines.

**Win32/SpyEye:** SpyEye is http-based botnet, which appeared in 2009 (Sood, Enbody & Bansal 2013). The architecture is similar to Zeus but comes with advance features. As Zeus, SpyEye was also released as commercial crime ware kit distributed through underground web forums. It also contains a builder module for creating the bot executable with config file and a Web control panel for command and control (C&C) of a bot net. Further, it is capable of keylogging, encrypted configuration file, daily email backup and detect and kill Zeus bot files.

**Win32/Warbot:** Warbot (YouTube 2013) specially design for DDoS attacks, in addition it let botmaster to run other files on the victim machine so it has a Loader capabilities too. This bot is cracked and available to download online.

We have infected the lab machines with different bot executable files and also reverse engineered to analyse the bot behaviour. We have been able to determine the communication patterns of the bots with the C&C server by information captured from the compromised devices. The Builder tool is used to build the encrypted dynamic configuration file and the bot executable file. For example, the Zeus Builder first checks if the system is already infected with the Zeus and the version of the infected file. Figure 5 shows the system information reported by the builder.

3.1.3 **Building the C&C Server**

We have used separate Windows XP virtual machine to built C&C Server. The Control Panel is an open source PHP application that is running on an IIS or Apache web server. Some additional software and MySQL user with appropriate permissions is also used in the setup. When the system is ready, the Control Panel code was copied into the web server directory. The install page can then be accessed from a browser. Once this form is completed the remainder of the setup is done automatically. Figure 6 shows the screen shots of C&C server setup window for Zeus botnet.

The router in Figure 4 is configured to capture the flows using NetFlow v9 and send them to SiLK flow collector (rwflowpack). Then the SiLK packing system stores it in a more space efficient “packed” format binary flat file in SiLK Flow record format while categorizing the flow records, and writing the records to hourly flat-files. Filtering Engine is used to reduce the data set by filtering out the data that is not useful for detecting bots. The filtered dataset is clustered and further analysed to find correlations between flows. Finally these correlations is used to identify botnet activates and victimized hosts in the network.

![Figure 6: Zeus Control - Installation configuration](image)

3.2 **Results**

Figure 7 shows the generic template that was developed to detect different bots for IPv4 protocol. We have only used the best attributes for detecting different bot families such as Zeus, SpyEye and Warbot. Figure 8 shows a sample flow record that is captured by the generic template and forwarded to the flow collector. The total length of the dataset is only 30 bytes. This is of significant advantage compared to related techniques since the size of the dataset for these cases is 48 bytes in fixed templates. Also, several attributes that are used in the fixed template are not efficient for detecting the bots. Furthermore the related techniques can only detect specific bot families. Our model can detect bots from different bot families.

![Figure 7: Generic template](image)

IPFIX can also be used to capture specific bytes from the application layer. Hence the generic templates in our model can be easily extended to detect various bot families who use ICMP, SNMP, IPv6 or DNS as
communication channels to get C&C by considering the application layer attributes, IPv6 attributes, ICMP attributes and DNS attributes in IP Flows.

The template in Figure 7 does not make use of the application layer data for detecting the botnets. We can also use additional filed for the generic template for high detection accuracy of specific bots. Figure 9 shows the generic template with additional fields related to packet size that can use for detecting specific bots such as Zeus by considering their unique C&C characteristics.

![Figure 8: Sample flow record from generic template](image)

![Figure 9: Generic template with additional attributes](image)

Figure 10 shows the sample flow record that is captured by the extended template and forwarded to the flow collector.

We have tested the performance impact of the Cisco router, by network flows. Figure 11 shows the average (five runs) overhead on the router for capturing the traffic flows with fixed IP flows (NetFlow v5) and our generic template. Hence our generic template has minimal overhead on the routers and the detection accuracy is higher since only the attributes that are efficient for the bot detection are used in the template.

### 3.3 Discussion

Our model has several advantages for detecting a range of bots. Let us consider some of the advantages.

Most of the related techniques make use of fixed templates for the classification of the flows and/or can only detect specific bot families. Techniques, which make use of the fixed templates, can be applied to devices from specific vendors. Also the fixed templates are not efficient since many of the attributes that are used for the flow classification are not usable for detecting the bots. Furthermore it is not possible to extend these templates with efficient attributes for detecting the attacks. Hence the techniques, which make use of the fixed templates, have higher dataset overhead and are less efficient. Currently, there are several bot families such as IRC, HTTP, and peer-to-peer bots. Hence the techniques that can detect only specific bots are not much use.

![Figure 11: CPU Overhead](image)

Our model makes use of only the attributes, which are efficient for detecting the bots for developing the templates. Since we make use of IPFIX for the specification of the template, it can be applied on the devices that belong to different vendors. Furthermore it has minimal dataset overhead since it only requires 30 bytes for a flow record and detecting a range of bot families. The overhead on the routers is also minimal compared to the fixed templates. Table 1 summarises the advantages of our model with related techniques.

### 4 Related Work

In this Section we will present some of the related techniques and compare them with our model.

BotSniffer (Gu, Zhang & Lee 2008) proposed network based anomaly detection techniques for detecting the bots. This system is capable of identifying C&C servers as well as infected hosts in a network. Detection is based on the fact that command and control communication within the same botnets most likely exhibit spatial-temporal correlation and similarity such as coordinated communication, propagation, DDoS and fake activities. However, this methodology limited to detect centralized botnets (IRC and HTTP).
Botfinder (Tegeler et al. 2012) proposed a methodology that senses bot in a network, using NetFlow v5 dataset. Botfinder leverages the discovery that C&C communication of a particular bot family trail specific regular pattern. This technique uses average time between start times of two subsequent flows in the trace, average duration of a connection, average number of bytes transferred to the destination, average number of packets transferred to the destination and Fourier transformation over the flow start times in the trace to detect C&C activities. This approach also limited to detect centralized botnets (IRC and HTTP).

(Zhao et al. 2012) analyze NetFlow v5 characteristics to detect peer-to-peer botnet by introducing time windows for NetFlow v5 dataset. However, this approach has relied on the selection of correct size of the time window to cover activities of a bot. So they proposed two phases in their framework; training phase to find correct size for time window and detection phase to detect botnets from active flows. They have used 4 attributes from the NetFlow v5 dataset, which are source IP, source port, destination IP, destination port, L4 protocol. Then they have calculated 8 more attributes from NetFlow data that are, number of reconnects for a flow, number of flows from this address over the total number of flows generated per hour and attributes with in selected time window such as average payload packet length, variance of payload packet length, number of packets exchanged, number of packets exchanged per second, the size of the first packet, the average time between packets while examine behavior of peer to peer botnets such as Strom, Nugache and Waledac. This approach limited to peer-to-peer botnet and need to have complete network flow in order to effectively detect peer-to-peer bots.

(Zhao et al. 2013) examines the feasibility of detecting botnet activity without seeing a complete network flow by classifying behavior based on time intervals as extension to their previous study. Also the paper shows experimentally that it is possible to identify the presence of botnet activities with high accuracy even with very small time windows. Authors mainly addressed the limitation of payload inspection that resource intensive behavior, inability to deal with encryption and violation of privacy by using flow data to analysis. Detection method explained here was based on Network flow v5 while detecting real time botnet activities by inspecting the characteristics of these flows in small windows. This approach is also specific to peer-to-peer botnet detection. The system called DICSLOSURE (Bilge et al. 2012) present a large-scale botnet (Centralized) detection method based on NetFlow v5 data and improved machine learning techniques. Authors identified unavailability of network data sets and terabits per seconds line speed. Authors consider following limitations of NetFlow in order to build their system. The NetFlow doesn’t contain payload of the packet but aggregated metadata of the packet flows. Another limitation of the NetFlow is it can only consider one direction of the flow. NetFlow sampling used in large networks makes another limitation for malware detection. The system (DICSLOSURE) rely on NetFlow attributes; the source IP address, The destination IP address, Source port, Destination Port, Start and finish time of the flow, number of packets and number of bites transferred. Flow size, client access pattern and temporal behavior have been used to distinguish C&C communication over benign traffic.

(Strayer et al. 2008) constructed NetFlow v5 based botnet detection method for IRC bots. This technique first eliminates traffic that is unlikely to be a part of a botnet, classifies the remaining traffic into a group that likely to be part of a botnet, then correlates the likely traffic to find common communications patterns that would suggest the activity of a botnet. Strayer uses a five step filtering mechanism for filtering the traffic. We have already presented a detail discussion and comparison with this work in Section 2.2.3.

(Zang et al. 2011) introduced a fine flow classification method to detect centralized botnet using RTT (Round Trip Time) of the IP packets in NetFlow version 5 data set and K-mean clustering algorithms. Also authors highlighted that the difficulty of preparing universal system to detect all type of botnets. (Tarng et al. 2011) proposed a six-step methodology to identify peer-to-peer botnets by analysing network flow traffic and the payload characteristics. As the authors rely on payload characteristics this method not possible to be applied on high volume data networks. BotTrack (Francois et al. 2011) extend the Google’s linkage analysis algorithm named PageRank, by adding clustering process based on DBSCAN algorithm in order to detect peer-to-peer botnet which are not generating large amount of communication traffic. NetFlow v5 data is used to define linkages in each

<table>
<thead>
<tr>
<th>Paper</th>
<th>HTTP</th>
<th>IRC</th>
<th>P2P</th>
<th>Generic</th>
<th>ICMP</th>
<th>IPv6</th>
<th>DNS</th>
<th>SMTP</th>
<th>Record Size (per record)</th>
<th>IPFIX compatible</th>
</tr>
</thead>
<tbody>
<tr>
<td>Botfinder (2012)</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>48 bytes</td>
<td></td>
</tr>
<tr>
<td>Disclosure (2012)</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>48 bytes</td>
<td></td>
</tr>
<tr>
<td>Xiaonan Zang et al (2011)</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>48 bytes</td>
<td></td>
</tr>
<tr>
<td>Timothy Strayer (2008)</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>48 bytes</td>
<td></td>
</tr>
<tr>
<td>David Zhao et al (2012)</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>48 bytes</td>
<td></td>
</tr>
<tr>
<td>Huihu Hou et al (2012)</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>48 bytes</td>
<td></td>
</tr>
<tr>
<td>BotTrack (2011)</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>48 bytes</td>
<td></td>
</tr>
<tr>
<td>Wernhaur Tarng et al (2011)</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>whole packet</td>
<td></td>
</tr>
<tr>
<td>BotSniffer (2008)</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>48 bytes</td>
<td></td>
</tr>
<tr>
<td>Our Method</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>30 bytes</td>
<td>✔</td>
</tr>
</tbody>
</table>

Table 1: Comparison with related techniques
host and tracking communication patterns. Linkage analysis and clustering techniques apply together, to explore group of hosts showing similar behavioural patterns. This method is limited to detect peer-to-peer botnets.

From the above discussion and considering Table 1, it is clear that the related techniques make use of the fixed template available in Netflow v5 and can only detect specific bot families. Also high overheads are in their approaches in the aspect of data set size and CPU utilization in order to capture the IP flows. We have shown that our model can detect a range of bots and has minimal overhead.

Popular commercial systems such as Arbor-Peakflow, Radware-Defencenow and Cisco Guard are able to detect botnets in attack phase (ex. DDOS), however our system can detect, bots during different phases in botnet life cycle (ex. infection, command and control, update and attack).

5 Conclusion:

In this paper we have proposed traffic analysis techniques for detection a range of bot families. Most of the related techniques make use of the fixed IP flows available in different products for detecting specific bot families. Our model makes use of IPFIX for designing a generic template to detect a range of bot families. We have also shown that generic template has minimal overhead on the routers. In the future work we will extend this model to deal with new types of botnets.

6 References


YouTube 2013, Warbot Tutorial, 06-May-2014, <https://www.youtube.com/watch?v=U8yrodbHg1Q%3E.


Lego© Mindstorms: Merely a Toy or a Powerful Pedagogical Tool for Learning Computer Programming?

Jacqui Chetty  
University of Johannesburg  
jacquie@uj.ac.za

Abstract  
Research world-wide indicates that computer programming is a difficult subject to teach and learners find such modules challenging. The result is that attrition rates are high. South Africa experiences similar problems. These problems are exacerbated by learners entering tertiary institutions under-prepared. Such institutions are under much pressure to transform learners’ academic capabilities. Transformation can be realised through alternative pedagogical approaches as it is well known that these approaches can influence learning outcomes in a positive manner.

This paper documents an innovative pedagogical approach to teach computer programming through the use of robotics. Lego Mindstorms robots have been used to further develop learners’ problem solving skills as well as encourage learners to “think out of the box”. The pedagogical approach supports Vygotsky’s philosophy regarding the Zone of Proximal Development. A case study was conducted and learners were expected to complete different projects using the Lego Mindstorms EV3 robots. The aim of each project was twofold. Firstly, to further reinforce fundamental computer programming concepts that had been partially developed. Secondly, to increase interaction between learners and generate higher motivation and interest in computer programming. A qualitative analysis was performed subsequent to the case study. The issues regarding the pedagogical approach are discussed and feedback from learners is analysed. The results are positive and encouraging.

Keywords: Lego Mindstorms robots, innovative pedagogical approach, programming, problem solving

1 Introduction

The idea of games as a pedagogical approach to teaching-and-learning computer programming is not new (Lawhead 2002). This said, using games as a learning tool is advocated as games have the potential to positively contribute to successful learning (Piteira 2011).

Lego Mindstorms robots is one such game that provides an innovative teaching tool for building learners computer programming skills. Amongst others, the game provides two necessary elements for learning, namely understanding and motivation (Piteira 2011).

It provides a platform for learners to build, reinforce and practice fundamental computer programming concepts, while adding an element of fun. Lego Mindstorms scaffolds learners’ learning because it uses action instead of explanation; accommodates a variety of learning styles and skills; reinforces mastery skills; provides an opportunity to practice; and affords an interactive, decision making context.

This paper investigates the benefits of using Lego Mindstorms as a pedagogical tool to reinforce learners’ fragile knowledge related to the fundamental concepts of computer programming. As motivation often predicts how successful a learner is academically (Jenkins 2000, Serrano-Camara 2014), the paper also investigates whether Lego Mindstorms provides a high level of motivation.

The paper is structured as follows: section 2 presents a review of literature related to computer programming and includes difficulties faced by learners learning to program. Pedagogical approaches in use today are also discussed, as well as Lego Mindstorms as a pedagogical tool. Section 3 describes the methodology approach used and section 4 evaluates and analyses results produced from the methodological approach. Section 5 highlights observations noticed by the educator. Section 6 concludes the paper.

2 Background

2.1 Difficulties faced by learners

The skills expected for computer programming are complex. These skills include the ability to:

- Solve problems (Development-OECD 2004, Mead 2006);
- Articulate a problem into a programming solution (Garner 2005, Lahtinen 2005);
- Construct mechanisms and explanations (Soloway 1986);
- Combine syntax and semantics into a valid program (Winslow 1996);
- Understand larger entities of a program instead of smaller details (Lahtinen 2005);
- Apply fundamental computer programming concepts (Robins 2003, Garner 2005);
- Understand abstract concepts (Lahtinen 2005); and
- Properly estimate their level of understanding (Lahtinen 2005).

Many of these skills require learners to think in an abstract manner, or require higher order thinking skills (HOTS). Unfortunately, learners in South Africa often do not acquire such skills at primary and secondary educational level (Jansen 2012). This means that when such learners are presented with a subject, such as computer programming, they struggle to deliberate concepts in an abstract manner (Mason 1999). For
example, although learners may understand how to solve a particular problem mathematically, they may not be able to articulate the problem into a programming solution by applying fundamental computer programming constructs learnt in the classroom.

2.2 Abstract reasoning

Abstract reasoning is one of the most important mental tools that learners must have to become competent computer programmers (Corney 2012). Lister (2011) proposes that learners may possess limited skills in the early stages of their lives, but such skills should develop and mature, given that learners are educated and receive formal training (Lister 2011). However, research indicates that very few learners’ are able to engage with and solve programming problems that involve abstract reasoning. As seen in Table 1, which illustrates Lister’s Neo-Piagetian theory, learners’ progress as programmers, in a similar fashion to the developmental process described above.

Given the importance of abstract reasoning Kramer (2007) asks whether it is possible to improve learners’ abstract reasoning through education and training. He also advocates that unless learners’ abstract reasoning is well developed, they should not be allowed admission into computing courses.

In the South African context, Kramer’s advice cannot be implemented, as the mandate from the government is to be inclusive of previously disadvantaged learners’. Nelson Mandela felt very strongly about this and insisted that learners have the right to tertiary education (Jansen 2012). Unfortunately, many learners’ attend schools that are over populated and under resourced (Mason 1999). Consequently, tertiary educators cannot adopt the approach of testing for abstract reasoning ability as many learners’ do not meet the requirements. Therefore, innovative pedagogical approaches that favour the development of abstract reasoning have to be investigated and implemented. Lego Mindstorms is one such tool that is investigated in this study. The next section explores the different pedagogical approaches used for computer programming.

2.3 Pedagogical approaches to computer programming

Pedagogical approaches relate to the manner in which teaching-and-learning takes place in order to facilitate desired learning outcomes (Pears 2009). There are many pedagogical approaches to teaching-and-learning (Boyer 2008, Pears 2009). However, the traditional teacher-centric pedagogical approach is still the most popular (Nicolaides 2012).

The teacher-centric approach consists of activities, such as lecturing, questioning and demonstration. The lecturer is the expert who transfers their knowledge across to learners (Xiao and Hub 2006). This approach is used extensively to teach not only computer programming modules, but also other disciplines of study. Although the teacher-centric approach is the most popular approach, there are other pedagogical approaches that are unique to teaching-and-learning computer programming.

The learner-centred pedagogical approach involves philosophies that have been around for many decades and even a century. These include social constructivism, peer-

<table>
<thead>
<tr>
<th>Stage</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sensor motor</td>
<td>Students who are able to trace code with less than 50% accuracy are at this stage.</td>
</tr>
<tr>
<td>Pre operational</td>
<td>Students are able to competently trace code but they are not able to derive meaning or explanation regarding the outcome of the code.</td>
</tr>
<tr>
<td>Concrete operational</td>
<td>Students are able to trace and describe code but they are only able to do so in familiar situations. A student would not be able to successfully transfer their skills from one context to another.</td>
</tr>
<tr>
<td>Formal operational</td>
<td>Students are deemed to be competent programmers at this stage.</td>
</tr>
</tbody>
</table>

Table 1: Lister’s (2011) Neo-Piagetian stages

Regardless of the pedagogical approach used in the classroom, learning computer programming can take on many forms. Firstly, teaching a particular language, such as Java, where the structure, syntax and semantics of the programming language itself is taught (Pears 2009). Most textbooks are structured according to the constructs of a particular programming language. For example, learners may learn how to make use of variables by applying the eight primitive data types known to Java.

Secondly, teaching problem solving techniques applicable to computer programming is another approach. The idea is that if a learner is able to solve one type of problem, that learners should be able to solve other problems of a similar nature (Winslow 1996, Pears 2009). Very precise computer programming structures are taught within this context. For example, instead of learning how to make use of variables by applying the eight primitive data types known to Java, learners can be taught by learners developing pseudo code or flowcharts.

Thirdly, teaching programming through the introduction of graphical user interface (GUI) tools, such as Scratch, Greenfoot or Alice provide a simulated computer programming environment that is user-friendly. It provides ease-of-use when trying to develop computer programs (Maloney 2010).

Lastly in this paper, teaching learners how to read, trace and debug existing programs (Patton, Miliszewska 2007) before they embark on writing their own programs is also very effective. Tracing a computer program reveals underlying concepts to learners that they most probably would not have thought of themselves. Learners then learn to mimic these revolutionary ideas and make them their own.

All of the above-mentioned approaches can embrace the use of peer collaboration. Peer collaboration is the process of dividing learners into groups of between five and eight, or pairs. They proceed to solve problems and develop computer programs collaboratively (Preston 2006, Teague 2007).
Figure 1: An EV3 Lego Mindstorms robot

2.3.1 Lego Mindstorms as a Pedagogical Approach

Lego Mindstorms robots have become a popular pedagogical tool to teach and learn introductory computer programming concepts (Lawhead 2002, Lui 2010). In effect, Stein (Stein 1998) challenges the computer science teaching community to move from the premise that computation is calculation to the notion of computation is interaction. Robots would be a natural way to explore such a concept.

Lego Mindstorms form part of Lego education and can be bought through a representative responsible for retailing such toys. As illustrated in Figure 1, the Mindstorms consist of building components, a programmable brick (EV3), active sensors and motors. There is software for which both GUI and command line interfaces are available. These simple tools provide an opportunity for educators to transform classrooms into rich laboratory or software studios, where learners can experience learner-centred learning, collaborative learning and peer-to-peer programming experimentation. This environment provides an opportunity for learners to “put their programming skills to the test” as what they program comes to life through the Lego Mindstorms robot. They can visually understand “what works” and “what does not work” and “why”.

Lego Mindstorms robots provide an opportunity for learners to understand fundamental computer programming concepts that are, by their very nature, abstract (deRaadt 2008). These concepts are not analogies with the real world (Piteira 2011) and learners find it challenging to relate to real-world problems. Moreover, traditional pedagogical approaches to teaching computer programming exacerbate this problem (Rountree 2002).

Introducing Lego Mindstorms robots provides a unique opportunity to transform a classroom environment in which (Piteira 2011): learners are given an opportunity to “grapple” with real-world problems; the Lego Mindstorms robot becomes a learning tool that can scaffold learners; fragile knowledge of abstract programming concepts can be reinforced; and learners are given an opportunity to experiment, explore and enjoy programming. This creates a degree of motivation.

<table>
<thead>
<tr>
<th>Category</th>
<th>Freq</th>
<th>Freq %</th>
</tr>
</thead>
<tbody>
<tr>
<td>Improve problem solving skills</td>
<td>21</td>
<td>100%</td>
</tr>
<tr>
<td>Testing programming abilities using real-world problems</td>
<td>5</td>
<td>23.8%</td>
</tr>
<tr>
<td>Fun way to develop programming skills</td>
<td>21</td>
<td>100%</td>
</tr>
<tr>
<td>Collaboration with peers</td>
<td>21</td>
<td>100%</td>
</tr>
<tr>
<td>Critical thinking</td>
<td>11</td>
<td>52.4%</td>
</tr>
<tr>
<td>Motivation factor</td>
<td>10</td>
<td>47.6%</td>
</tr>
<tr>
<td>Made new friends</td>
<td>7</td>
<td>33.3%</td>
</tr>
</tbody>
</table>

Table 2: Classification of categories

2.4 The motivation factor

Research indicates that emotions, such as hope, anger, relief, anxiety and boredom are significantly related to motivation, learning strategies, cognitive resources, self-regulation, and academic achievement, as well as personality and classroom antecedents (Pekrun 2002). According to Jenkins (Jenkins 2000) motivation in particular is a crucial component related to learners success. Although motivation is difficult to quantify, Jenkins has identified expectancy and value as two factors, which when multiplied can predict learners motivation (Jenkins 2001). Expectancy is related to the extent to which learners feel that they are able to succeed. Value is related to what they expect to gain. For example, confident learners who feel that they are able to succeed will attach a value or goal related to high marks. They will most likely score high in the area of motivation as: \( motivation = expectancy \times value \) (Jenkins 2001).

A motivated learner would therefore experience emotions related to hope, enjoyment and pride, whereas an unmotivated learner would experience emotions related to anger, frustration, anxiety and boredom. Lego Mindstorms robots provides an opportunity for learners to experiment and explore. The idea of learning-through-play is an effective tool to create personal motivation and satisfaction of learning (Piteira 2011).

3 Methodology

In this paper, a qualitative analysis regarding learner reflections of Lego Mindstorms robots was undertaken in order to identify how the robots assisted learners in improving their fundamental computer programming skills. This was explored within a case study of an introductory computer programming course. The research questions that were asked relates to: firstly, did the learners reflections indicate that the robot programming further improved their partially developed fundamental computer programming concepts? Secondly, are learners more motivated to become computer programmers due to their exposure to Lego Mindstorms robots?

3.1 Research Method

An instrumental case study was a suitable approach for answering the research questions as it allows a particular case to be examined to provide insight into the extent to which robot programming can scaffold learners’ learning of fundamental computer programming concepts (Denzin 1994). Case studies capture the complexities of a
phenomenon; such detailed observations can very rarely be captured in surveys or experimental designs (Hartley 2004, Falkner 2014). Although the sample size is small, the number of participants in the study being 21, it is sufficient for qualitative research that ranges from 1 to 99, with an average sample size of 22 (Troskie-de Bruin 2013). The case study will be repeated again later in the year for a further 42 learners’.

The study has adopted only a qualitative approach. The data were subjected to grounded theory analysis, starting with the process of open coding, before proceeding to axial coding. Grounded theory involves establishing a coding framework and it differs from other types of qualitative analysis in that a specific, structured coding framework is not employed (Charmaz 2006). The first stage in grounded theory development is open coding, where the data is broken down into distinct segments in order to obtain the full collection of ideas and concepts present in the data, without regard to how it will be used. Subsequently, axial coding is employed, where the coding framework developed during the open coding stage is refined and reorganised into specific categories, informed by theoretical frameworks and comparison within the data (Henning 2004).

The case considered in this research project forms part of an introductory computer programming course at a university in Johannesburg, South Africa. Learners within this course have already completed a 6 month (semester A) course related to the development of problem solving skills. Furthermore, they are currently in their second 6 months (semester B) learning a computer programming language, namely C#. Their computer programming competencies are still developing. The learning objectives for both courses include awareness and application of simple data structures, related algorithms and algorithm complexity, as well as initial experiences in small-scale problem solving and software development.

The Lego Mindstorms robots project has been included as a supplementary course. One hundred and twenty six learners were approached about the project and a small demonstration was conducted to illustrate to them some of the skills that they would learn. Learners were informed that the Lego Mindstorms project was not mandatory. A total of 63 learners volunteered to be part of the project. Learners were divided into 3 groups of 21 each, where each group was further sub-divided into 7 groups of 3. Each group of 21 were presented with a variety of Lego Mindstorms projects that they had to complete over a period of 4 weeks (a 2 hour session per week).

The Lego Mindstorms course includes explanations of various components, instructions for building a standard EV3 robot, guides to program the robot and perform the basic tasks, and exercises for practicing. Each session included suggestions for learners to explore further. Figure 2 illustrates an educator demonstrating an action.

3.2 Coding Framework

The basic unit of analysis in this project were coding units (Charmaz 2006). During the open coding stage, sections of text, such as a word, phrase, sentence or paragraph, were coded while the selection represented a single idea or concept related to using robots as an innovative pedagogical approach. During the axial coding stage, the researcher refined the established codes into categories, merging codes where appropriate and identifying specific categories as derived from the data. The qualitative software ATLAS.ti was used (Troskie-de Bruin 2013) to code the units as well as derive specific categories. Table 2 illustrates the specific categories as defined by the axial coding. Table 3 illustrates the extent to which learners felt that the Lego Mindstorms robots provided them with an opportunity to reinforce fundamental programming concepts that they had already learnt.

4 Qualitative Analysis

The analysis so far indicates that firstly, learners recognise the benefit of Lego Mindstorms as a tool to further reinforce fragile fundamental programming concepts. Secondly, Lego Mindstorms motivates learners to become better programmers and a passion for programming is developed. As illustrated in Table 2 the following categories emerged.

4.1 Solving real-world authentic activities

Researchers and experts world-wide agree that an authentic learning activity represents a problem that has real-world relevance, is ill-defined, and needs to be completed over a period of time (Herrington 2006, Lombardi 2007, Brannock 2013, Herrington 2013).

Real-world relevance relates to problems that match “every day” tasks of professionals in practice. Such problems are normally “messy” or ill-defined. Ill-defined problems are problems that when described to learners are open to interpretation, as opposed to problems that are developed by following step-by-step solutions. Instead of being highly prescriptive, ill-defined problems provide an opportunity for learners to identify the steps needed to complete the activity (Herrington 2006). As ill-defined problems are more complex, learners need a longer period of time to complete such activities. A longer time period allows learners to reflect on the choices they make regarding the solution. This enhances their metacognitive skills (Lombardi 2007).

Authentic learning activities provide an opportunity for learners to construct new knowledge instead of reproducing existing knowledge. In order to achieve this, learners are provided with multiple sources from which they can draw information, examine the problem from many angles, distinguish relevant information from irrelevant information, and formulate a solution (Lombardi 2007). These tasks are usually completed in collaboration, and learners are given the opportunity to discuss problems, ideas and solutions, thus learning from one another, before completing the task.

Learners in particular embraced and acknowledged the idea of using the Lego Mindstorms robots to program in the real-world. For example, learners were asked to develop a “car” using their robot. The car had to be able to navigate from point A to point B, while avoiding obstacles and adhering to traffic robots (red – stop; green – go). As learners understood the real-world problem in the context of their own lives, this made it easier for them to develop and implement the coding instructions.

Some of the responses from learners included:
programming concept | “yes” | “no”
---|---|---
Problem solving | 21 | 
Breaking problems into small steps | 20 | 1
Practice algorithmic skills | 20 | 1
Assist in understanding how to solve problems better algorithmically | 21 | 
Assist in coding programs that require input-processing-output | 19 | 2
Assist in coding programs that require methods | 21 | 
Assist in coding programs that require repetition | 21 | 

Table 3: Responses regarding Mindstorms reinforcing learning
- Yes I enjoyed it because of the observation of using concepts of programming logic in physical situations
- It’s a fun programming logic, whereby you can solve problems in different ways through that robot. It is realistic, because you can just imagine the robot in real-life situations

Another learner acknowledged the notion of using the Lego Mindstorms robot to benchmark his personal programming skills.
- Yes, I really did. When I joined the project I was curious and I wanted to test my abilities by trying and taking an opportunity that was presented to us. I am glad I did because now I know where I stand and I know I can

Asking learners to solve problems that involved authentic tasks that they could related to, provided essential scaffolding so that gap between “understanding the problem” and determining “how to solve it” was not too wide.

4.2 Critical thinking

Computer programming is a subject that is rife with abstract concepts. For learners to better understand such concepts, it may be adventitious to provide them with meaningful problems to solve. As per the Neo-Piagetian theory any improvement in abstract reasoning skills results in improved computer programming competencies. Furthermore, allowing learners to solve such problems in a collaborative manner may further enhance abstract reasoning.

Vygotsky (1978) suggested this as any higher order thinking skill (including computer programming skills) evolves in the construction of joint social activities (collaborative tasks), prior to developing into skills that can be applied to independent problem solving (Beck 2013). The social spaces or “zones of proximal development” are critical if higher order thinking skills are to be achieved (Vygotsky 1978).

For many of the learners at the university, critical thinking and deeper level learning is difficult to achieve. However, the learners acknowledged the notion of critical thinking when learning how to program the robots.

Figure 2: Demonstration in action
- It challenged our thinking and encouraged us to make use of critical problem solving skills to accomplish our tasks
- It was different from the usual work we do in the labs. It provided the platform to test what we learnt in class – it tested our analytical skills

4.3 Collaborative learning

The idea of constructing knowledge through a social setting can be a powerful educational and learning tool (Vygotsky 1978, Ben-Ari 1998, Kozulin 2003, Stetsenko 2010). This type of learning involves two important concepts, namely social constructivism and collaboration. Social constructivism is a philosophy and a learning theory that has established itself in the last few decades (Karagiorgi 2005). It attempts to take the individual into account by viewing them as unique, having their own personal and subjective experiences (Karagiorgi 2005). The individual imposes meaning on the world by constructing knowledge based on past experiences, goals, curiosities and beliefs (Cole 1992). In this way, the individual adapts and constructs knowledge as they make sense of their world.

It is important to note that the newly constructed knowledge may not necessarily be ‘correct’ as the individual’s past experiences may be based on ‘misperceptions’ or ‘misunderstandings’ (Karagiorgi 2005). When a misperception occurs the individual will either ignore the conflict; construct a better model of prior understanding; or reflect on the existing model and create a new way of thinking about that model (Perkins 1999). However, ‘misperceptions’ may be dealt with in another way, one in which the individual works in collaboration with others.

It is widely acknowledged that there are many educational advantages that can be derived from learners working in collaboration with one another (Brown 2005, Preston 2006). The expression “I learn what I believe as I hear myself speak” is very powerful. The biggest advantage is that learners are more successful when learning occurs in the midst of others. Learners learn from one another as they discuss problems and formulate solutions (Ben-Ari 1998, Lombardi 2007).

Collaborative learning is one of the most powerful characteristics of Lego Mindstorms. Learners remark that:
I made new friends who helped me on my programming skills and now I can take shortcuts to make the best out of programming.

I was a person who liked to isolate herself but now I can’t even count how many friends I have, so it boosted my social life.

We were discussing and bringing our knowledge into one

Collaboration with others is a powerful pedagogical tool. Such collaboration often leads to increased motivation among learners.

4.4 Motivating learners and Lego Mindstorms

Hirumi (Hirumi 2008) argues that games are effective tools for learning because, amongst other advantages, games create personal motivation and satisfaction. One of the main advantages of using games, such as Lego Mindstorms to motivate learners, is due to the inherent characteristics associated with the games, namely energy, direction, persistence and equifinality – all aspects that stimulate and motivate learners (Serrano-Camara 2014).

Motivation forms part of a conceptual model or framework known as the self-determination theory. This theory emphasises the importance of the development of internal human resources for personal development and self-regulation. Such self-determination refers to something because it is interesting or enjoyable, such as the Lego Mindstorms. Such games provide a natural wellspring of learning and achievement that can either be encouraged or discouraged by facilitators practices. Such motivation results in high quality learning and creativity (Ryan 2000).

The Lego Mindstorms robots inspired and motivated learners to think about programming in a very real way. They reported that:

Because now I want to program bigger and better robots

It motivates me because it showed me that what I am studying is relevant to today

Better programmer, makes me want to perfect the robot and make it do exactly what I want it to do

5 Observations

The following observations came to light during the study:

Learners collaboration increased significantly, relationships where formed not only with peers, but also with tutors and educators;

Although learners had not been exposed to Lego Mindstorms prior to the case study, they quickly understood what was expected of them. Instead of being intimidated they embraced the experience;

There was a high level of excitement as learners completed complex programming tasks. This in turn, lead to the promotion of a high level of motivation;

There was an element of competitiveness amongst learners’ as they competed against each other. This further added to the motivation factor; and

All of the above factors lead to an increase in learners’ confidence.

6 Conclusion

In the analysis, a variety of categories related to whether Lego Mindstorms robots provided a platform that reinforces fragile computer programming concepts, were identified. Learners’ reflections illustrate that Lego Mindstorms robots provide them with an opportunity to:

Improve their problem solving / algorithmic skills;

Engage with authentic real-world activities to test their programming abilities;

Stimulate creative thought and develop critical thinking;

Collaborate with peers and learn socially; and

Motivate them to achieve and become creative in their thinking.

Lego Mindstorms robots is a platform that allows the focus of learning to shift from traditional learning to one where learners can engage in real-world problems. It allows them to develop dynamic programming skills in a language independent environment. In today’s rapidly changing world, providing innovative ways to scaffold learners and challenge them to think in other ways is essential.

7 Acknowledgements

The financial assistance of the National Research Foundation (NRF) in South Africa under Grant number 84244 is hereby acknowledged. Any opinion, findings and conclusions or recommendations expressed in this material are those of the authors and the NRF does not accept any liability thereto.

The assistance of Dr E.C. Anderssen, who was responsible for the classroom facilitation, the management thereof, and support of the Lego Mindstorms case study is also hereby acknowledged.

8 References


Hourglass Systems

Yuki Osada1  Mark Reynolds2  Tim French3
School of Computer Science
The University of Western Australia,
Email: yuki.osada@research.uwa.edu.au1
mark.reynolds@uwa.edu.au2
tim.french@uwa.edu.au3

Abstract

In this paper, we first describe a simplified scenario observed in mining operations, where trucks are loaded and emptied at different locations. We will then look at ways that we can model this subsystem using timed automata so we may be able to apply model checking techniques on such systems. Specifically, we will look at modifications to the Hourglass Automata and Interrupt Timed Automata to model the truck loading, unloading scenario, and we will show that reachability is maintained.

Keywords: Model Checking, Timed Automata, Hybrid Automata.

1 Introduction

Automata are mathematical objects that are capable of modelling many real-world systems. By modelling systems with automata, it allows us to run formal verifications on such systems to ensure correctness, to reason about the system properties, to compute or determine properties of the system, or to synthesise a controller. The main class of automata we are interested in is timed automata as they allow the modelling of systems where the behaviour can depend precisely on when other actions were taken. These systems whose correctness depends critically on the time an action is performed are called real-time systems.

Formal verification of real-time systems is an important task in industry since a fault in a design or schedule can cost millions. For example, scheduling in industrial processes is a multi-million dollar operation across a wide range of industries from mining (13) through transport logistics to chemical and mechanical production (14). A well-known industry tool used for the model checking of timed automata is UPPAAL (12). Standard UPPAAL is limited to decidability problems, but there are also a number of UPPAAL extensions which integrate features from various timed automata extensions. UPPAAL-CORA is one of these that introduce Cost Optimal Reachability Analysis on linearly priced timed automata, which are timed automata with an extra price variable that may be updated during state transitions or for staying in a state.

Hybrid systems, as defined by Alur et al. (5), are finite automata with a set of real-valued variables of varying rates of change. These variables can appear in state transition guards and location invariants in the system. Timed automata, as described by Alur and Dill (3), is a subset of hybrid systems that model time using a finite set of real-valued clocks with constant, unit rate of growth, and they have been studied extensively due to their usefulness in industrial modelling. These clocks can have a number of operations associated with them that may be applied during a state change. The main operation that can be applied to them is the reset, where a set of clocks can be reset to zero.

A common verification problem for automata is the reachability problem. The reachability problem is: given an initial state, we want to determine whether there exists an accepting run, which is a run from the initial state to a final state. If no such run exists, then a final state is not reachable. For timed automata, Alur and Dill (1) proved that the reachability problem is decidable and PSPACE-hard, so for any standard timed automaton, we are able to determine if some state is reachable. However, this reachability property can be lost by introducing extra functionality. The decidability of reachability for different clock update functions in timed automata have been looked at by Bouyer et al. (8). They show that introducing the updates $x := c$ and $x := y$ are decidable, while $x := x + 1$ and $x := y + c$ are decidable for 3 or more clocks only when no diagonal constraints are used.

The stopping of clocks is a feature that is required to express some real-time systems where some real-valued memory is needed. Cassez and Larsen (6) define an extension of timed automata called the stopwatch automata, which allow the stopping and starting of clocks, and they show that this results in an expressive power equal to linear hybrid automata. Reachability in linear hybrid automata is known to be undecidable, so this shows that stopping time can cause undecidability in general. However, Bérard et al. (10) show it is possible to have a restricted form of stopping time that maintains decidability of reacha-
bility with the interrupt timed automata, where clock levels are used to restrict values that they can be compared to when running. Additionally, hourglass automata with two hourglass clocks are capable of having stopped clocks (15).

2 Preliminaries

Timed automata are finite automata coupled with a finite set of real-valued variables, called clocks, which all increase at unit rate at all locations. Each of these clocks may be reset to zero immediately after a transition, and these transitions can have clock constraints associated to them. The clock constraints on transitions are called guards, and unless the constraints in the guard are met, the transition cannot be taken. Similarly, constraints on locations are called invariants, and the constraint must be met to stay in that location. The location invariants and transition guards may only consist of comparisons between clock values and non-negative integer constants. We restrict it to integer values since any rational constant can be multiplied by a large enough value that they all become integers.

A clock valuation is a map from clock variables to non-negative real-values. States in timed automata are written as location and clock valuation pairs, \((s, v)\). Since there can be an infinite number of clock valuations, we can have an infinite number of states in timed automata. There are two types of state transitions in a timed automaton:

- a delay transition \((s, v) \rightarrow_d (s, v + t)\) for some \(t \geq 0\), where \(v + t\) is the clock valuation \(v\) with all clock values incremented by \(t\).

- an action transition \((s, v) \rightarrow_a (s', v')\) for some action \(a\), where \(s'\) is a location directly connected to \(s\) in the automaton and \(v'\) is the valuation \(v\) after the transition updates are applied.

The only clock update available in the standard timed automata is the reset operation, \(x := 0\).

For every clock \(x \in X\), we will let \(c_x\) be the largest integer constant that \(x\) is compared against in the clock constraints. Once \(x\) exceeds \(c_x\), the valuation of \(x\) can be considered to be \(\infty\) as its value would be indistinguishable from any value above \(c_x\). While the values are still distinguishable, we can split the fractional and integral components as comparisons in clock constraints are all with integers. So for some \(t \in \mathbb{R}^+\), let \(fr(t)\) be the fractional component of \(t\), and \([t]\) be the integral component of \(t\). This gives us \(t = [t] + fr(t)\).

2.1 Hourglass Automata

Hourglass automata (15) are extensions of timed automata that can have clocks that go backwards as well as forwards, and these clocks are bound to the range \([0, c_x]\). Additionally, the guards and invariants in a hourglass automata are limited to comparing a clock \(x\) to its bounds: 0 and \(c_x\). Comparing clocks or comparing clocks to other constants does not make sense for hourglasses. Additionally, clocks in an hourglass automata can potentially be stopped to simulate the process of placing an hourglass on its side.

2.1.1 Syntax

We introduce clocks that can behave like hourglasses. We now define the key operation we can apply to hourglasses, which is their ability to be flipped, resulting in the clock progressing backwards.

Definition 2. A flip of a clock is an operation that multiplies the rate of change of the real-value of the clock by \(-1\). All clocks initially have a rate of change of 1, so the flip operation alternates the rate of change between 1 and \(-1\).

Lastly, we can toggle the progress of time for an hourglass clock by placing them on their side or back upright.

Definition 3. Toggling a clock switches the clocks state between paused and running. Hourglass clocks start in the running state, so toggling the clock will stop it and another toggle would start it again. Toggling the state does not affect the direction of the rate of change, so starting the clock from the paused state will let the clock progress in the set direction.

The rate of change of a clock can be tracked by extending the state of the automaton to include \(d \in D : X \rightarrow \{-1, 1\}\) that maps clocks \(x \in X\) to a value in the set \([-1, 1]\), which encodes the direction that time is progressing for each clock. This multiplies the number of states by \(2^{|X|}\), where \(|X|\) is the number of clocks in the system. We can further extend this to record the stopped clocks by mapping clocks to a value in \([-1, 0, 1]\), resulting in the number of states being multiplied by \(4^{|X|}\). The \(-1\) is needed to store the direction that time should progress in when toggled back on. Additionally, flipping a stopped clock, should switch the direction that time progresses in when the clock is started again.

With the above two definitions, we can now define our hourglass automata.

Definition 4. An hourglass automaton is a tuple \(A = \langle \Sigma, S, s_i, S_f, X, I, T \rangle\), where:

- \(\Sigma\) is a finite set of actions,
- \(S\) is a finite set of locations,
- \(s_i \subseteq S\) is the initial location,
- \(S_f \subseteq S\) is the set of final locations,
- \(X\) is a finite set of hourglass clocks,
- \(I : S \rightarrow C(X)\) is a mapping from locations to clock constraints (the location invariants), where a clock constraint \(\phi \in C(X)\) maps a clock \(x \in X\) to a constraint such that \(\phi(x) = v(x) \prec c\) or \(\phi(x) = c \prec v(x)\) or \(\phi = \phi_1 \wedge \phi_2\), where \(c \in \{0, c_x\}\). \(<\in \{<, \leq\}\), \(\phi_1, \phi_2 \in C(X)\), and \(v : X \rightarrow \mathbb{R}\) maps clocks to their clock valuation.
- \(T \subseteq S \times \Sigma \times C(X) \times 2^X \times 2^X \times S\) is a set of transitions, where the 6-tuple \((s, a, \phi, \mu_{\text{flip}}, \mu_{\text{toggle}}, s')\) is a transition from location \(s\) to location \(s'\) with the label \(a\). This transition is enabled when the constraint \(\phi\) is met, and taking this transition will flip a set of clocks \(\mu_{\text{flip}} \subseteq X\), and toggle the progress of time in a set of clocks \(\mu_{\text{toggle}} \subseteq X\).
In previous work (15), it is noted that we can simulate the flip operation with a clock update of the form \( x := c_x - x \), where \( x \in X \) and \( c_x \) is its upper bound. This removes the need to have a negative rate of change. Additionally, it was proven that reachability is decidable for two clocks or less. This was done by adding reverse diagonal constraints, \( fr(v(x)) + fr(v(y)) = 1 \), onto the original region graph used to prove general decidability for timed automata. This does not however show that hourglass automata with three or more clocks is undecidable since this region graph is capable of expressing more than just an hourglass automata. In particular, we can compare clocks against one another, which is not a feature of hourglass automata.

In Hourglass automata the state of the clock cannot change maximum time. Once the sand runs out of the hourglass the state of the clock cannot change backwards, or an accumulated resource running out.

Figure 2: The region graph of timed automata, augmented with reverse diagonal constraints marked as 3. As time progresses, the system will move from 1 to 4 in order, then back to 1. The clock update will flip the position in a unit square. The reverse diagonal constraints are needed to capture the flipping on the normal diagonal constraints.

2.2 Interrupt Timed Automata

Interrupt timed automata (10) are extensions of timed automata that can have at most a single clock active at a time, and each clock is given a unique interrupt level, which influences how each clock can interact with the other clocks. At a given interrupt level, the clocks from lower levels are suspended with rate zero while those from higher levels are undefined, and thus given an arbitrary value of zero. Transition guards are linear constraints, using only clocks from the current or lower level, and the defined clocks can be updated by linear expressions of clocks from levels lower than it. Along with the levels, locations have a timing policy which indicate whether time may, may not, or must elapse.

2.2.1 Syntax

Definition 5. An interrupt timed automaton is a tuple \( A = (\Sigma, AP, S, s_i, S_f, X, \lambda, lab, T) \), where:

- \( \Sigma \) is a finite set of locations,
- \( s_i \) is the initial location,
- \( S_f \subseteq S \) is the set of final locations,
- \( pol : S \rightarrow \{ \text{Lazy, Urgent, Delayed} \} \) is the timing policy of locations,
- \( X = \{ x_1, \ldots, x_n \} \) consists of \( n \) interrupt clocks,
- \( \lambda : S \rightarrow \{ 1, \ldots, n \} \) is a mapping that associates with each location its level and we call \( x_{\lambda(s)} \) the active clock in state \( s \)
- \( lab : S \rightarrow 2^{AP} \) labels each state with a subset of \( AP \) of atomic propositions,
- \( T \subseteq S \times C(X) \times (\bigcup x_{\lambda(s)} \neq 0 \) (involving clocks from levels less than or equal to \( k \)). The guard \( \varphi \) is a conjunction of constraints \( \sum a_i x_j + b > 0 \) for \( i \leq k \).
- \( \lambda \) is either of the form \( \sum a_i x_j + b \) or \( C_i = x_i \), and for \( i > k \), \( C_i = 0 \).

The reachability problem on the interrupt timed automata is shown to be decidable by building a finite automaton as a generalized class graph. It is shown (10) that it is in \text{NEXPTIME} when the number of clocks is not fixed, and in \text{PTIME} when it is fixed. This is an interesting subclass of hybrid automata since it allows the stopping of clocks while also allowing comparisons between clocks.

3 Hourglass System

An ore mining operation is a complex system with many interacting parts. We will look at the loading and unloading of a single truck, where the load size is not fixed. The truck will be loaded from a fixed shovel and unloaded at a fixed crusher. A truck in an open-pit mining operation will generally move between an ore source and the crusher, where the ore is not fixed. The truck will be loaded from a fixed ore source and unloaded at a fixed crusher. A truck in an hourglass operation will generally move between a fixed source and a fixed destination, where the load size is fixed. This is an interesting subclass of hybrid automata since it allows the stopping of clocks while also allowing comparisons between clocks.

- Hourglasses can represent time only up to a fixed maximum time. Once the sand runs out of the hourglass the state of the clock cannot change.
- Hourglasses cannot be compared directly to one another. Typically there is no way to tell whether one hourglass has more time remaining than another, (other than course estimations based on appearance).
- Hourglasses can be flipped. Flipping an hourglass results in time running backwards, or an accumulated resource running out.

A limitation of this model is that we restrict the loading time and unloading time to be equal.

Successfully being able to model such a subsystem could allow us to apply model checking techniques or optimise properties for systems that have may have

\footnote{This is reminiscent of the normalisation process used in (3).}
Figure 3: An open-pit mine will generally have a crusher where ore is dropped off and some shovellers where ore is loaded onto a truck.

such a sub-process. For example, we could check if a certain state is reachable, or if a certain output target can be reached within a time constraint. Alternatively, we could look for optimal truck loads for optimal operation. This may be maximum throughput, or maximum output within a time constraint.

3.1 Hourglass Automata

We can model the loading and unloading of a single truck system with two clocks and four locations.

Locations:
1. Moving from crusher to shovel. (initial state)
2. Loading ore onto truck at shovel.
3. Moving from shovel to crusher.
4. Dropping off ore at the crusher.

Clocks:
1. Track distance to crusher. (bounded by $\max(Distance_{crusher\rightarrow shovel}, Distance_{shovel\rightarrow crusher})$)
2. Track truck load. (bounded by $MaxLoadTime$, initially toggled off)

Transitions:

1 $\rightarrow$ 2:
- Guards: $Clock_1 = Distance_{crusher\rightarrow shovel}$
- Updates: $toggle(Clock_2), toggle(Clock_1), flip(Clock_1)$

2 $\rightarrow$ 3:
- Guards: $Clock_2 \leq MaxLoadTime$
- Updates: $toggle(Clock_2), flip(Clock_2), toggle(Clock_1)$

3 $\rightarrow$ 4:
- Guards: $Clock_1 = 0$
- Updates: $toggle(Clock_2), toggle(Clock_1), flip(Clock_1)$

4 $\rightarrow$ 1:
- Guards: $Clock_2 = 0$
- Updates: $flip(Clock_2), toggle(Clock_2), toggle(Clock_1)$

From the above description, we find that each location has one state.

3.2 Interrupt Timed Automata

We can model the loading and unloading of a single truck system with four locations, each with their own clock.

Locations:
1. Moving from crusher to shovel. (initial state)
2. Loading ore onto truck at shovel.
3. Moving from shovel to crusher.
4. Dropping off ore at the crusher.

Transitions:

1 $\rightarrow$ 2:
- Guards: $Clock_1 = Distance_{crusher\rightarrow shovel}$
- Updates: Increment level

2 $\rightarrow$ 3:
- Guards: $Clock_2 \leq MaxLoadTime$
- Updates: Increment level

3 $\rightarrow$ 4:
- Guards: $Clock_3 = Distance_{shovel\rightarrow crusher}$
- Updates: Increment level

4 $\rightarrow$ 1:
- Guards: $Clock_4 = Clock_2$
- Updates: Drop level back to base

We see that each state follows the previous state, going in a loop, and is thus finite. This gives us an hourglass automaton with clocks to represent the given scenario. The hourglass automata is presently known to be decidable for two clocks or less, which is sufficient for expressing the above process.
4 Conclusion and Future Work

In this paper, we first described a simplified hourglass system scenario where:

- material is loaded onto a truck at the shovel at a fixed rate,
- truck transports the material to the crusher,
- material is unloaded off the truck into the crusher at a fixed rate.

We then looked at how this subsystem could be modelled using hourglass automata or interrupt timed automata. Both classes of automata were able to model this subsystem, but the hourglass automata allowed a more natural representation where one clock represented the truck’s load while another clock would be used to model the truck’s movement. If the load size is fixed, then this can be done with standard timed automata since we won’t need to keep track of how much was loaded while at the shovel when unloading at the crusher. However, for the given example scenario, it may be sufficient to use a more discrete system since the loading time could just be an integer multiple of a shovel making one scoop.

The current models are able to simulate this simplified process, but if we want to make further use of these models to calculate properties like throughput or worst case execution time, we need to capture other global information. For example, to capture total time, we would need another standard clock (or “price” variable) running in parallel. Similarly for capturing the total ore unloaded at the crusher.

It is often the case that the loading rate and unloading rate in a system is different, so a future extension would be to look at the possibility of modelling the same scenario, but with different rates for loading and unloading. This is a harder scenario since if we simply extend hourglass automata to be multi-rate such that each clock has a fixed absolute rate of change, we end up needing to compare two hourglass clocks to synchronise them after loading to allow a secondary clock with a different rate to simulate the unloading. Without being able to compare clocks, we can’t do this synchronisation of two clocks of different rates. It is also known that reachability for standard multi-rate timed automata with clock comparisons is not decidable (5). Without this difference in rates, the applicability of these models could be more limited, and it may be more suited to modelling passengers on a bus or similar, where loading and unloading rates are expected to be approximately equal.

Another extension would be the possibility of modelling multiple trucks running in parallel. This could be split into levels, where we first allow trucks to have no interaction with each other other than some shared information like the total amount of material dropped off at the crusher. Next, we could introduce the queuing of trucks. These are not difficult when the load size is fixed, but with a non-fixed load size, trucks could interact with one another differently.

References


Author Index

Akande, Ayodeji James, 91
Anam, Sarawat, 17
Chetty, Jacqui, 111
Damavandinejadmonfared, Sepehr, 59
Dean, Jonathan, 47
Ding, Xi, 65
Fidge, Colin, 91
Foo, Ernest, 91
French, Tim, 119
Gong, Xiangyang, 65
Höller, Johann, 9
Herbert, Jeff, 27
Kang, Byeong, 47
Kim, Yang Sok, 17
Li, Qin, 73
Litchfield, Alan, 27
Liu, Qing, 17
Moir, Christopher, 47
Nand, Parma, 83
Osada, Yuki, 119
Parry, David, 63
Perera, Rivindu, 83
Quirchmayr, Gerald, 9
Reynolds, Mark, 119
Sanders, J W, 73
Smith, Graeme, 73
Stabauer, Martin, 9
Thompson, Jason, 37
Tian, Ye, 65
Torabi, Torab, 37
Tupakula, Udaya, 101
Varadharajan, Vijay, 59, 101
Wang, Wendong, 65
Wijesinghe, Udaya, 101
You, Hanxu, 3
Zhang, Lanshan, 65
Zhu, Jie, 3
Recent Volumes in the CRPIT Series

**ISSN 1445-1336**

Listed below are some of the latest volumes published in the ACS Series _Conferences in Research and Practice in Information Technology_. The full text of most papers (in either PDF or Postscript format) is available at the series website http://crpit.com.

<table>
<thead>
<tr>
<th>Volume Number</th>
<th>Title</th>
<th>Editors</th>
<th>ISBN</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Volume 157</td>
<td>Database Technologies 2013</td>
<td>Edited by Hua Wang, University of Southern Queensland, Australia and Rui Zhang, University of Melbourne, Australia.</td>
<td>978-1-921770-22-7</td>
<td>Contains the proceedings of the Twenty-Fourth Australasian Database Conference (ADC 2013), Adelaide, Australia, 29 January – 1 February 2013.</td>
</tr>
<tr>
<td>Volume 159</td>
<td>User Interfaces 2013</td>
<td>Edited by Ross T. Smith, University of South Australia, Australia and Burkhard C. Wünsche, University of Auckland, New Zealand.</td>
<td>978-1-921770-24-1</td>
<td>Contains the proceedings of the Fourteenth Australasian User Interface Conference (AUIC 2013), Adelaide, Australia, 29 January – 1 February 2013.</td>
</tr>
<tr>
<td>Volume 160</td>
<td>Parallel and Distributed Computing 2013</td>
<td>Edited by Bahman Javadi, University of Western Sydney, Australia and Saurabh Kumar Garg, IBM Research, Australia.</td>
<td>978-1-921770-25-4</td>
<td>Contains the proceedings of the Eleventh Australasian Symposium on Parallel and Distributed Computing (AusPDC 2013), Adelaide, Australia, 29 January – 1 February 2013.</td>
</tr>
<tr>
<td>Volume 162</td>
<td>Health Informatics and Knowledge Management 2013</td>
<td>Edited by Anthony Wirth, University of Melbourne, Australia.</td>
<td>978-1-921770-27-2</td>
<td>Contains the proceedings of the Sixth Australasian Workshop on Health Informatics and Knowledge Management (HIKM 2013), Adelaide, Australia, 29 January – 1 February 2013.</td>
</tr>
<tr>
<td>Volume 164</td>
<td>The Web 2013</td>
<td>Edited by Helen Ashman, University of South Australia, Australia and Yanz Z. Sheng, University of Adelaide, Australia and Andrew Trotman, University of Otago, New Zealand.</td>
<td>978-1-921770-29-6</td>
<td>Contains the proceedings of the First Australasian Web Conference (AWC 2013), Adelaide, Australia, 29 January – 1 February 2013.</td>
</tr>
<tr>
<td>Volume 166</td>
<td>Computer Science 2014</td>
<td>Edited by Bruce Thomson, University of South Australia and Dave Parry, AUT University, New Zealand.</td>
<td>978-1-921770-31-9</td>
<td>Contains the proceedings of the Australian System Safety Thirty-Seventh Australasian Computer Science Conference (ACSC 2014), Auckland, New Zealand, 20 – 23 January 2014.</td>
</tr>
<tr>
<td>Volume 169</td>
<td>User Interfaces 2014</td>
<td>Edited by Burkhard C. Wünsche, University of Auckland, New Zealand and Stefan Marks, AUT University, New Zealand.</td>
<td>978-1-921770-34-0</td>
<td>Contains the proceedings of the Fifteenth Australasian User Interface Conference (AUIC 2014), Auckland, New Zealand, 20 – 23 January 2014.</td>
</tr>
<tr>
<td>Volume 173</td>
<td>Health Informatics and Knowledge Management 2014</td>
<td>Edited by Anthony Wirth, University of Melbourne, Australia.</td>
<td>978-1-921770-37-0</td>
<td>Contains the proceedings of the Seventh Australasian Workshop on Health Informatics and Knowledge Management (HIKM 2014), Auckland, New Zealand, 20 – 23 January 2014.</td>
</tr>
</tbody>
</table>