

LNAI Series Editors

Randy Goebel

University of Alberta, Edmonton, Canada

Yuzuru Tanaka

Hokkaido University, Sapporo, Japan

Wolfgang Wahlster

DFKI and Saarland University, Saarbrücken, Germany

LNAI Founding Series Editor

Joerg Siekmann

DFKI and Saarland University, Saarbrücken, Germany

Longbing Cao Ana L.C. Bazzan
Andreas L. Symeonidis Vladimir I. Gorodetsky
Gerhard Weiss Philip S. Yu (Eds.)

Agents and Data Mining Interaction

7th International Workshop, ADMI 2011
Taipei, Taiwan, May 2-6, 2011
Revised Selected Papers

Series Editors

Randy Goebel, University of Alberta, Edmonton, Canada

Jörg Siekmann, University of Saarland, Saarbrücken, Germany

Wolfgang Wahlster, DFKI and University of Saarland, Saarbrücken, Germany

Volume Editors

Longbing Cao

University of Technology, Sydney, NSW, Australia

E-mail: longbing.cao-1@uts.edu.au

Ana L.C. Bazzan

Universidade Federal do Rio Grande do Sul (UFRGS), Porto Alegre, RS, Brazil

E-mail: bazzan@inf.ufrgs.br

Andreas L. Symeonidis

Aristotle University, Thessaloniki, Greece

E-mail: asymeon@issel.ee.auth.gr

Vladimir I. Gorodetsky

St. Petersburg Institute for Informatics and Automation, Russia

E-mail: gor@mail.iias.spb.ru

Gerhard Weiss

University of Maastricht, The Netherlands

E-mail: gerhard.weiss@maastrichtuniversity.nl

Philip S. Yu

University of Illinois at Chicago, IL, USA

E-mail: psyu@cs.uic.edu

ISSN 0302-9743

e-ISSN 1611-3349

ISBN 978-3-642-27608-8

e-ISBN 978-3-642-27609-5

DOI 10.1007/978-3-642-27609-5

Springer Heidelberg Dordrecht London New York

Library of Congress Control Number: 2011944829

CR Subject Classification (1998): I.2, H.3, H.4, H.2.8, F.1, H.5, J.1

LNCS Sublibrary: SL 7 – Artificial Intelligence

© Springer-Verlag Berlin Heidelberg 2012

This work is subject to copyright. All rights are reserved, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, re-use of illustrations, recitation, broadcasting, reproduction on microfilms or in any other way, and storage in data banks. Duplication of this publication or parts thereof is permitted only under the provisions of the German Copyright Law of September 9, 1965, in its current version, and permission for use must always be obtained from Springer. Violations are liable to prosecution under the German Copyright Law.

The use of general descriptive names, registered names, trademarks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Message from the Workshop Chairs

We are pleased to welcome you to the proceedings of the 2011 International Workshop on Agents and Data Mining Interaction (ADMI 2011), held jointly with AAMAS 2011.

In recent years, agents and data mining interaction (ADMI, or agent mining) has emerged as a very promising research field. Following the success of ADMI 2006 in Hong Kong, ADMI 2007 in San Jose, AIS-ADM 2007 in St. Petersburg, ADMI 2008 in Sydney, ADMI 2009 in Budapest, ADMI 2010 in Toronto, the ADMI 2011 workshop in Taiwan provided a premier forum for sharing research and engineering results, as well as potential challenges and prospects encountered in the coupling between agents and data mining.

The ADMI 2011 workshop encouraged and promoted theoretical and applied research and development, which aims at:

- Exploiting agent-enriched data mining and machine learning, and demonstrating how agent technology can contribute to critical data mining and machine learning problems in theory and practice
- Improving data mining-driven agents and systems, and showing how data mining and machine learning can strengthen agent intelligence and intelligent systems in research and practical applications
- Exploring the integration of agents and data mining toward a super-intelligent system and intelligent information processing
- Identifying challenges and directions for future research and development in agent mining, through the synergy and interaction among relevant fields
- Reporting workable applications and case studies of agent mining

The 11 accepted papers were competitively selected from 24 submissions from 13 countries. ADMI 2011 submissions cover areas from North America, Europe to Asia, indicating the boom of agent mining research globally. The workshop also included two invited talks by Peter Stone at the University of Texas at Austin, USA, and Gal Kaminka at Bar Ilan University, Israel.

Following ADMI 2009, the ADMI 2011 papers are published as an LNAI post-proceedings volume by Springer. We appreciate Springer, in particular Alfred Hofmann, for the continuing publication support.

ADMI 2011 was sponsored by the Special Interest Group: Agent-Mining Interaction and Integration (AMII-SIG: www.agentmining.org). We appreciate the guidance of the Steering Committee.

More information about ADMI 2011 is available from the workshop website: <http://admi11.agentmining.org/>.

VI Message from the Workshop Chairs

Finally, we appreciate the contributions made by all authors, Program Committee members, invited speakers, panelists, and the AAMAS 2011 workshop and local organizers.

May 2011

Gerhard Weiss
Philip S. Yu
Longbing Cao
Ana Bazzan
Andreas L. Symeonidis
Vladimir Gorodetsky

Organization

General Chair

Gerhard Weiss
Philip S. Yu

University of Maastricht, The Netherlands
University of Illinois at Chicago, USA

Workshop Co-chairs

Longbing Cao
Ana Bazzan

Andreas L. Symeonidis
Vladimir Gorodetsky

University of Technology Sydney, Australia
Universidade Federal do Rio Grande do Sul,
Instituto de Informatica, Brazil
Aristotle University of Thessaloniki, Greece
Russian Academy of Sciences, Russia

Workshop Organizing Co-chairs

Dionysis Kehagias

Informatics and Telematics Institute, Centre for
Research and Technology Hellas, Greece

Program Committee

Agnieszka Dardzinska
Ahmed Hambaba
Ajith Abraham

Andrea G.B. Tettamanzi
Andrzej Skowron
Boi Faltings

Bialystok Technical University, Poland
San Jose State University, USA
Norwegian University of Science and
Technology, Norway
University of Milan, Italy
Institute of Decision Process Support, Poland
Artificial Intelligence Laboratory, The Swiss

Chengqi Zhang
Daniel Kudenko
Daniel Zeng
David Taniar
Dionysis Kehagias
Eduardo Alonso
Eleni Mangina

Federal Institute of Technology in
Lausanne, Switzerland
University of Technology Sydney, Australia
University of York, UK
The University of Arizona, USA
Monash University, Australia
ITI-CERTH, Greece
City University London, UK
University College Dublin, Ireland

VIII Organization

Eugenio Oliveira	University of Oporto, Portugal
Frans Coenen	University of Liverpool, UK
Gerhard Weiss	University of Maastricht, The Netherlands
Giovanna Di Marzo Serugendo	University of London, UK
Hai Zhuge	Institute of Computing Technology, Chinese Academy of Sciences, China
Heikki Helin	CSC-IT Center for Science, Finland
Henry Hexmoor	Southern Illinois University at Carbondale, USA
Hillol Kargupta	University of Maryland Baltimore County, USA
Huaglory Tianfield	Glasgow Caledonian University, UK
Ioannis Athanasiadis	Dalle Molle Institute for Artificial Intelligence, Switzerland
Jason Jung	Yeungnam University, Korea
Jean-Marc Petit	INSA Lyon/Université de Lyon, France
Jiming Liu	Hong Kong Baptist University, China
Jinglong Wu	Kagawa University, Japan
Joerg Mueller	Technische Universität Clausthal, Germany
Kagan Tumer	Oregon State University, USA
Karl Tuyls	Maastricht University, The Netherlands
Katia Sycara	Carnegie Mellon University, USA
Kazuhiro Kuwabara	Ritsumeikan University, Japan
Ken Kaneiwa	National Institute of Information and Communications Technology, Japan
Kristian Kersting	University of Bonn, Germany
Leonid Perlovsky	Harvard University, USA
LUIS Otavio Alvares	Universidade Federal do Rio Grande do Sul, Brazil
Martin Purvis	University of Otago, New Zealand
Matthias Klusch	DFKI, Germany
Mengchu Zhou	New Jersey Institute of Technology Newark, USA
Michal Pechoucek	Czech Technical University, Czech Republic
Mircea Negoita	Wellington Institute of Technology, New Zealand
Mirsad Hadzikadic	University of North Carolina, Charlotte, USA
Nathan Griffiths	University of Warwick, UK
Ngoc Thanh Nguyen	Wroclaw University of Technology, Poland
Ning Zhong	Maebashi Institute of Technology, Japan
Oleg Karsaev	St. Petersburg Institute for Informatics and Automation of the Russian Academy of Sciences, Russia
Pericles Mitkas	Aristotle University of Thessaloniki, Greece

Ran Wolff	Haifa University, Israel
Renata Guizzardi	Federal University of Espírito Santo in Vitória, Brazil
Sandip Sen	The University of Tulsa, USA
Seunghyun Im	University of Pittsburgh at Johnstown, USA
Sherief Abdahla	University of Southern California, USA
Simeon Simoff	University of Technology Sydney, Australia
Sung-Bae Cho	Computer Science Department of Yonsei University, Korea
Sviatoslav Braynov	University of Illinois at Springfield, USA
Tapio Elomaa	Tampere University of Technology, Finland
Takashi Washio	Osaka University, Japan
Valerie Camps	Université Paul Sabatier, France
Victor Skormin	Binghamton University, USA
Vijay Raghavan	University of Louisiana at Lafayette, USA
Vipin Kumar	University of Minnesota, USA
Vladimir Gorodetsky	Russian Academy of Sciences, Russia
Vladimir Marik	Czech Technical University, Czech Republic
Wee-Keong Ng	Nanyang Technological University, Singapore
Wen-Ran Zhang	Georgia Southern University, USA
William Cheung	Hong Kong Baptist University, China
Yanqing Zhang	Georgia State University, USA
Yaodong Li	Institution of Automation, Chinese Academy of Sciences, China
Yasufumi TAKAMA	Tokyo Metropolitan University, Japan
Yasuhiko Kitamura	Kwansei Gakuin University, Japan
Yiyu Yao	University of Regina, Canada
Yves Demazeau	Domaine Universitaire de Saint-Martin-d'Hres, France
Zbigniew Ras	University of North Carolina, USA
Zhong Zhi-Shi	Institute of Computing Technology, Chinese Academy of Sciences, China
Zili Zhang	Deakin University, Australia

Steering Committee

Longbing Cao	University of Technology Sydney, Australia (Coordinator)
Edmund H. Durfee	University of Michigan, USA
Vladimir Gorodetsky	St. Petersburg Institute for Informatics and Automation, Russia
Hillol Kargupta	University of Maryland Baltimore County, USA
Matthias Klusch	DFKI, Germany
Michael Luck	King's College London, UK

X Organization

Jiming Liu
Pericles A. Mitkas
Joerg Mueller
Ngoc Thanh Nguyen
Carles Sierra

Gerhard Weiss
Xindong Wu
Philip S. Yu
Chengqi Zhang

Hong Kong Baptist University, China
Aristotle University of Thessaloniki, Greece
Technical University of Clausthal, Germany
Wroclaw University of Technology, Poland
Artificial Intelligence Research Institute of the
Spanish Research Council, Spain
University of Maastricht, The Netherlands
University of Vermont, USA
University of Illinois at Chicago, USA
University of Technology Sydney, Australia

Table of Contents

Part I: Agents for Data Mining

Intersections of the Future: Using Fully Autonomous Vehicles (Abstract)	3
<i>Peter Stone</i>	
Toward a Methodology for Agent-Based Data Mining and Visualization	4
<i>Elizabeth Sklar, Chipp Jansen, Jonathan Chan, and Michael Byrd</i>	
A Multi-agent Based Approach to Clustering: Harnessing the Power of Agents	16
<i>Santhana Chaimontree, Katie Atkinson, and Frans Coenen</i>	
Agent Enriched Distributed Association Rules Mining: A Review	30
<i>G.S. Bhamra, A.K. Verma, and R.B. Patel</i>	
Obtaining an Optimal MAS Configuration for Agent-Enhanced Mining Using Constraint Optimization	46
<i>Chayapol Moemeng, Can Wang, and Longbing Cao</i>	
Towards a Numerical, Agent-Based, Behaviour Analysis: The Case of Tourism	58
<i>Sébastien Corniglion and Nadine Tournois</i>	
A Comparative Study of a Financial Agent Based Simulator Across Learning Scenarios	86
<i>Filippo Neri</i>	
Agent-Based Cluster Analysis of Tropical Cyclone Tracks in the Western North Pacific	98
<i>Wei Zhang and Yuanfei Wang</i>	

Part II: Data Mining for Agents

Exploiting Domain Knowledge in Making Delegation Decisions	117
<i>Chukwuemeka David Emele, Timothy J. Norman, Murat Sensoy, and Simon Parsons</i>	
Data Mining to Support Human-Machine Dialogue for Autonomous Agents	132
<i>Susan L. Epstein, Rebecca Passonneau, Tiziana Ligorio, and Joshua Gordon</i>	

An Instance-Window Based Classification Algorithm for Handling Gradual Concept Drifts	156
<i>Vahida Attar, Prashant Chaudhary, Sonali Rahagude, Gaurish Chaudhari, and Pradeep Sinha</i>	
Towards a Multiagent-Based Distributed Intrusion Detection System Using Data Mining Approaches	173
<i>Imen Brahmi, Sadok Ben Yahia, Hamed Aouadi, and Pascal Poncelet</i>	
Change Point Analysis for Intelligent Agents in City Traffic	195
<i>Maksims Fiosins, Jelena Fiosina, and Jörg P. Müller</i>	
Mining Frequent Agent Action Patterns for Effective Multi-agent-Based Web Service Composition	211
<i>Xiaofeng Wang, Wenjia Niu, Gang Li, Xinghua Yang, and Zhongzhi Shi</i>	
Enhancing Agent Intelligence through Evolving Reservoir Networks for Predictions in Power Stock Markets	228
<i>Kyriakos C. Chatzidimitriou, Antonios C. Chrysopoulos, Andreas L. Symeonidis, and Pericles A. Mitkas</i>	
Pricing Analysis in Online Auctions Using Clustering and Regression Tree Approach	248
<i>Preetinder Kaur, Madhu Goyal, and Jie Lu</i>	

Part III: Agent Mining Applications

The Impact of Recommender Systems on Item-, User-, and Rating-Diversity	261
<i>W. Kowalczyk, Z. Szlávik, and M.C. Schut</i>	
Opinion Formation in the Social Web: Agent-Based Simulations of Opinion Convergence and Divergence	288
<i>Pawel Sobkowicz, Michael Kaschesky, and Guillaume Bouchard</i>	
A Data-Driven Approach for Resource Gathering in Real-Time Strategy Games	304
<i>Dion Christensen, Henrik Ossipoff Hansen, Jorge Pablo Cordero Hernandez, Lasse Juul-Jensen, Kasper Kastaniegaard, and Yifeng Zeng</i>	
Data Cloud for Distributed Data Mining via Pipelined MapReduce	316
<i>Zhiang Wu, Jie Cao, and Changjian Fang</i>	

Successful Efficient and Intelligent Retrieval Using Analytic Hierarchy Process	331
<i>Monika Arora, Uma Kanjilal, and Dinesh Varshney</i>	
A Hybrid System Based on Multi-Agent Systems in Case of e-Wedding Thailand	344
<i>Kunyanuth Kularbphettong, Phayung Meesad, and Gareth Clayton</i>	
Author Index	361

Intersections of the Future: Using Fully Autonomous Vehicles

Peter Stone

University of Texas at Austin, USA
pst^on@cs.utexas.edu

Abstract. Artificial intelligence research is ushering in a new era of sophisticated, mass-market transportation technology. While computers can already fly a passenger jet better than a trained human pilot, people are still faced with the dangerous yet tedious task of driving automobiles. Intelligent Transportation Systems (ITS) is the field of study that aims to use artificial intelligence to make transportation safer, cheaper, and more efficient. Recent advances in ITS point to a future in which vehicles themselves handle the vast majority of the driving task. Once autonomous vehicles become popular, autonomous interactions amongst *multiple* vehicles will be possible. Current methods of vehicle coordination, which are all designed to work with human drivers, will be outdated. The bottleneck for roadway efficiency will no longer be the drivers, but rather the mechanism by which those drivers' actions are coordinated. While open-road driving is a well-studied and more-or-less-solved problem, urban traffic scenarios, especially intersections, are much more challenging.

This talk will address the question: “To what extent and how can a multiagent intersection control mechanism take advantage of the capabilities of autonomous vehicles in order to make automobile travel safer and faster?” First, I will introduce and specify the problem of intersection management as a multiagent system and define a metric by which solutions can be evaluated. Next, I will propose a novel multiagent intersection control mechanism in which autonomous driver agents “call ahead” and reserve space-time in the intersection, pending the approval of an arbiter agent called an intersection manager, which is located at the intersection.

Toward a Methodology for Agent-Based Data Mining and Visualization

Elizabeth Sklar^{1,2}, Chipp Jansen^{1,3}, Jonathan Chan¹, and Michael Byrd²

¹ Brooklyn College, The City University of New York, USA

² The Graduate Center, The City University of New York, USA

³ Hunter College, The City University of New York, USA

sklar@sci.brooklyn.cuny.edu, chipp@chipp.org,

{jonmchan,mbyrd1}@gmail.com

Abstract. We explore the notion of agent-based data mining and visualization as a means for exploring large, multi-dimensional data sets. In Reynolds' classic flocking algorithm (1987), individuals move in a 2-dimensional space and emulate the behavior of a flock of birds (or "boids", as Reynolds refers to them). Each individual in the simulated flock exhibits specific behaviors that dictate how it moves and how it interacts with other boids in its "neighborhood". We are interested in using this approach as a way of visualizing large multi-dimensional data sets. In particular, we are focused on data sets in which records contain time-tagged information about people (e.g., a student in an educational data set or a patient in a medical records data set). We present a system in which individuals in the data set are represented as agents, or "data boids". The flocking exhibited by our boids is driven not by observation and emulation of creatures in nature, but rather by features inherent in the data set. The visualization quickly shows separation of data boids into clusters, where members are attracted to each other by common feature values.

1 Introduction

We are motivated to explore the notion of agent-based data mining visualization [3], taking inspiration from the *Artificial Life* and *Information Visualization* communities. Advances in computer graphics, processor speed and networking bandwidth over last decade have made possible the application of dynamic techniques for information visualization that were previously limited to high-end graphics laboratory settings. In addition, the rapid rise in popularity of certain types of data visualization environments, particularly those from the *Geographic Information Systems (GIS)* community, have made commonplace otherwise obscure techniques for examining vast multi-dimensional data sets. *Google Earth* [2,7,9] is one example, which has brought to the masses the notion of zoomable interfaces and allowed everyday computer users to explore 3-dimensional geographic data sets facilitated by, what are now considered to be, standardized

controls. Our work takes advantage of these trends by hypothesizing that when any multi-dimensional data set is projected onto 2 or 3 dimensions, it can be explored as if it were a geographic landscape. Such “data landscapes” could be studied dynamically in interesting ways if points in the landscape are represented as software agents that move in response to various stimuli.

The work presented here develops this idea, with the long term goal of providing a participatory agent-based data mining and visualization system in which the user and the system collaboratively explore a data landscape. Classical statistics are often fine (and the right choice) when the user has some notion of what is in their data set and how they want to analyze their data. Looking for a bell curve in a data set of student grades is an example where a standard statistical method, like computing mean and standard deviation, is an appropriate choice. Using *K*-means [10] when clustering data into a known number of groups is an example where a standard machine learning method is appropriate. But when the number of clusters and even the selection of features on which to cluster the data are unknown *a priori*, other techniques must be investigated. Our hypothesis is that a *participatory* system in this type of situation can take advantage of superior human skills for quick visual understanding and intuition, facilitating a user to easily identify and nudge an evolutionary data mining process into a desired direction.

This paper is organized as follows. Section 2 describes the related work in this area, which was inspired by seminal work on “flocking boids” [17]. Sections 3 details our approach. Section 4 presents results from applying our approach to a sample data set. Finally, we conclude in Section 5 with summary remarks.

2 Related Work

In 1987, Cliff Reynolds [17] produced the classic work on flocking in artificial systems. Reynolds’ model focuses on graphical aspects, and the aim was to produce a realistic (from a graphical standpoint) simulation of a group of identical agents (which Reynolds calls “boids”). Each agent is given the same behavioral rules, which includes instructions for how to react to others in an agent’s “neighborhood”. These interaction instructions consist of three independent rules. First, there is a *separation* rule, which implements collision avoidance, preventing the agents from bumping into each other. Second, there is an *alignment* rule, which causes agents to move at the same velocity and heading as those around them. Third, there is a *cohesion* rule, which encourages agents to gravitate towards a common center. When put together, the composite set of agents exhibits emergent group behavior that looks like *flocking*—in the same way that schools of fish or flocks of birds or herds of land animals move together. This work has proven to be highly influential and has inspired most of the subsequent research in the area of artificial flocking.

Proctor and Winter [16] developed the notion of *information flocking*, about 10 years after Reynolds' work. Their aim was to visualize patterns of behavior of users visiting web sites. They simulated artificial “fish” and associated a fish with each user, mining users’ clickstreams to provide input to their simulation. A user clicking on a URL was interpreted as interest in the topic(s) displayed on the page. A matrix of user interests was updated, and the fish responded by grouping together—showing users who shared similar interests.

Moere [12] provides a comprehensive overview of decentralized data visualization work conducted in the two decades since Reynolds' original work. He divides the work in the field into three categories: *information particle* animation; *information flocking*; and *cellular ant* methods. The first category involves simulating a group of *information particles*, or “infoticles” (i.e., agents), in three-dimensional space. Agents react to *forces* in the system, moving in response to them. The forces can emanate from static points in the agents' artificial landscape, acting as fixed point attractors (or repellers), as well as from dynamic points—other agents. The forces influence the velocity and direction of movement of each agent. With an infoticle, data values are assigned to each agent. The forces acting on an agent are calculated according to the similarity (or dissimilarity) of the agents' data values in relation to those of other nearby agents or fixed-point attractors in the landscape.

The second category, *information flocking*, describes the method introduced by Proctor and Winter [16], discussed above. Picarougne *et al.* [13] extend the work of Proctor and Winter by introducing the notion of an “ideal” distance between agents that is proportional to the difference in feature values between the agents. They compare to the *K*-means clustering method and state that their results are similar. Another example that extends Proctor and Winter's work suggests using the visualization in tandem with other algorithms for data classification [1]. Moere [11] offers a nice extension to the Proctor and Winter work by applying the method to time-varying datasets and using financial stock market data as an example. The data values (e.g., prices or “quotes” of a given stock) are represented by agents, and the agents' values change over time. Moere introduces two additional behavior rules: *data similarity*, where agents stay close to others with similar data values; and *data dissimilarity*, where agents move away from others with different data values. The method highlights changes in “data behavior”—as data values change over time, agents that have similarly changing data values end up clustering together.

The third category combines ideas from *artificial ant systems* [5], *ant foraging* [4] and *cellular automata* [20]. The general idea is that artificial agents (“ants”) move around a two-dimensional grid world and collect “food”—i.e., data. The aim is to bring like pieces of data together, emulating the way that ants gather food particles and deposit them in shared nests. The agents' movement is constrained by cellular-automata-like rules in which they respond only to neighboring grid cells. While an interesting alternative, many comment that this approach takes more time to reach a solution than other methods. The reason is that the agents' movements are constrained (by the cellular automata rules),

and so even if an agent “knows” where it is going, it still has to find a path to get there. The advantage, however, is that forcing the agent to explore an indirect path will lead it to encounter other agents and new data (food) sources, which may eventually result in a more robust solution (even if it takes longer to find). Handl and Meyer [8] review ant-based clustering algorithms, comparing results of different implementations and discussing approaches from an optimization perspective.

Cui *et al.* [21] employ a flocking-based algorithm for document clustering analysis. Each document object is represented as a “boid” and projects the document’s TFIDF¹ vector onto a two-dimensional space. The authors compare their flocking-based clustering algorithm with K -means clustering and Ant clustering. They found that the flocking algorithm ran faster than the Ant algorithm, while K -means ran the fastest. However, the K -means method requires an *a priori* estimate of the correct number of groups, whereas the flocking and Ant-based methods do not. This makes the flocking algorithm the better choice for a dynamically changing data set, like the one used by the authors.

Picarougne *et al.* [14] describe a biologically inspired clustering algorithm, called *FClust*, which applies similar, but not identical, techniques to those of Proctor and Winter. The authors also model agents as representations of data points moving in a two-dimensional landscape. However, instead of applying three forces to the agents’ motion, the authors apply two forces: one that attracts agents with similar data values and one that repels agents with dissimilar data values. They determine experimentally the threshold for similarity, showing visually that different threshold values produce different clustering results, as one would expect. They apply their method to several data sets and compare results to K -means clustering methods. They also describe a component of their system referred to as “interactive clustering” in which a user can select and label data elements on a visual display and the system will perform clustering on the user-selected classes. The authors define an “entropy” measure (inspired by Shannon [18]) for evaluating the stopping condition for the system.

The work we present in the remaining sections has been inspired and informed by these and other examples, all of which were found primarily in the *Artificial Life* and *Information Visualization* literature. Our approach is most similar to that of Moere [11] and Picarougne *et al.* [14]. However, we take an *agent-based* perspective. Each agent in our system can only compare its data values to others in its geographic neighborhood, much as a robot in a multi-robot team can only respond to teammates that are within range of its visual or range sensors or communication network. Where Moere and Picarougne *et al.* stress two behavior rules (essentially data similarity and dissimilarity), we stick with Reynolds’ original three behavior rules (separation, cohesion and alignment); and we introduce an additional *meta-level* flocking step in which groups that are similar emit an attractive force and tend to move towards each other. The details are described in the next section.

¹ Term Frequency, Inverse Document Frequency—a common metric used in Natural Language Processing (NLP).

3 Approach

Our approach is built on the *information flocking* paradigm described in the previous section, with modifications highlighted below. We associate a “data boid”, or agent, with each record in an n -dimensional data set (i.e., a data set with n features in each record). In our visual environment, the agent moves around a two-dimensional geographic landscape, and its position is influenced by the relative values of its data features compared with those of its neighbors. Iterating over a number time steps, or “frames”, we animate the agents by first applying the three standard flocking rules, then *grouping* agents based on their feature value and geographic proximity, next executing a *meta-flocking* step that pulls similar groups together, and finally applying a *braking* factor (described below) as clusters of like agents converge.

The standard flocking procedure involves computing vectors representing three forces acting on each agent, followed by combining the three forces using a weighting scheme. The three force vectors are computed as follows:

- **Separation.** A steering vector is computed that points away from agents in the neighborhood with dissimilar feature values:

$$sep = \frac{1}{n} \sum_{i=1}^n d_i(P - P_i) \quad (1)$$

where n is the number of agents in the neighborhood within a specified feature distance (Δ), d_i is the geographic distance to neighbor i , P is the (x, y) position of the agent performing the computation, and P_i is the (x, y) position of neighbor i .

- **Cohesion.** A steering vector is computed that points toward the center of the group of agents in the neighborhood with similar feature values:

$$coh = \frac{1}{n} \sum_{i=1}^n P_i \quad (2)$$

where n is the number of agents in the neighborhood within a specified feature distance (Δ) and P_i is the (x, y) position of neighbor i .

- **Alignment.** A velocity vector is computed to match the average speed and heading of agents in the neighborhood with similar feature values:

$$ali = \frac{1}{n} \sum_{i=1}^n V_i \quad (3)$$

where n is the number of neighboring agents within a specified feature distance (Δ) and V_i is the velocity of neighbor i .

The vectors are weighted and summed to update the agent's position:

$$\text{velocity} = \psi \cdot \text{sep} + \alpha \cdot \text{ali} + \gamma \cdot \text{coh} \quad (4)$$

The weighting scheme amongst the three vectors is important, and there is a delicate balance between them. For the results presented here, we used weights of $\psi = 1.5$, $\alpha = 1.0$ and $\gamma = 1.0$.

The standard flocking algorithm only considers geographic distance (d) between agents, whereas we use geographic distance to determine an agent's neighborhood and distance in feature space (Δ) to determine whether agents should be attracted to or repel their neighbors. The distance in feature space is computed as follows. Multi-featured data sets frequently contain a mixture of *categorical* and *quantitative* types of features. We compute the distance between individual features, based on their type and normalized to $[0 \dots 1]$ (where 0 is most similar and 1 is most dissimilar), and then calculate the average over all features. The distance between two categorical feature values is:

$$\delta(a_i, b_i) = (a_i == b_i ? 0 : 1) \quad (5)$$

where a_i and b_i represent the values of the i -th feature for agents a and b , respectively. The distance between two quantitative feature values is:

$$\delta(a_i, b_i) = \frac{|a_i - b_i|}{\max_i} \quad (6)$$

where \max_i is the range of possible values for feature i . The overall feature distance between two agents is thus:

$$\Delta(a, b) = \frac{\sum_{i=1}^n \delta(a_i, b_i)}{n} \quad (7)$$

After the standard flocking rules are applied to all the agents, a *grouping* step takes place, followed by a *meta-flocking* step. Grouping is an organizational step in which the agents are partitioned into virtual groups based on their locations in geographic space. This is done using a recursive algorithm in which each agent looks at its neighbors (other agents within a fixed physical distance), and those neighbors look at their neighbors, and so on, adding to the group all neighbors (and neighbors' neighbors, etc.). Agents' positions do not change during the grouping step.

The meta-flocking step pulls together groups that have similar feature values. For each group, a set of feature values is calculated that represents the center of the group in feature space. For quantitative feature values, the center is the mean over all group members. For categorical feature values, the center is the mode (i.e., most popular) over all group members. A pairwise comparison is made between all groups' feature centers. A cohesion steering vector for the group is computed that points toward the geographic center of mass of other groups with similar feature values:

$$\text{coh}_g = \frac{1}{n} \sum_{i=1}^n M_i \quad (8)$$

where n is the number of groups within a specified feature distance (Δ) and M_i is the (x, y) position of the center of mass of group i . Note that this is different from the standard (agent-level) cohesion rule for two reasons. First, it only takes into account the feature distance between groups and does not use the geographic distance. Second, it compares with every group in the system, not just groups that are within close geographic distance proximity.

Finally a *braking* factor is applied to all members of groups whose feature values within the group are similar. As group membership settles, the system converges and the agents belonging to settled groups move more slowly and eventually halt.

As the clusters of agents evolve, the system computes two metrics to assess how the process is performing:

- *pctGood* computes the percentage of groups that fall within a pre-specified “goodness” threshold. The aim is for this value to converge to 1.0. The *goodness* of a group is a measure of the variance in group members’ feature values. A goodness value close (or equal) to 0 indicates that the group members are closely matched; a higher goodness value indicates a more diverse group. Goodness is calculated as:

$$goodness = \sum_{i=1}^n \left(\sum_{f \in Q} \sigma(f) + \sum_{f \in C} \left(1 - \frac{match(f)}{n} \right) \right) \quad (9)$$

where n is the number of agents in the group, Q is the set of each agents’ quantitative features, C is the set of each agents’ categorical features, $\sigma(f)$ is the standard deviation of (quantitative) feature f across all agents in the group, and $match(f)$ is the number of agents in the group with (categorical) feature f matching the mode for that feature in the group.

- *pctOverlap* computes the percentage of groups that have overlapping feature values. Since the system determines group membership dynamically, based on which agents have come within small graphical distances of each other while they float around in space, there is no guarantee that multiple groups with the same feature values will not appear. The overall aim is to find the minimal set of groups that have different feature values, since it is not desirable to have different groups that overlap in feature space. The aim is for this value to converge to 0.0.

4 Experiments and Results

We implemented our approach in a prototype system, developed using Processing [15]. The system can run in an interactive mode, where the user selects features on which to cluster, and the system responds in real-time. The user can change feature selection during clustering, and the system will adjust. To assess the efficacy of our method for clustering, we ran a series of experiments in

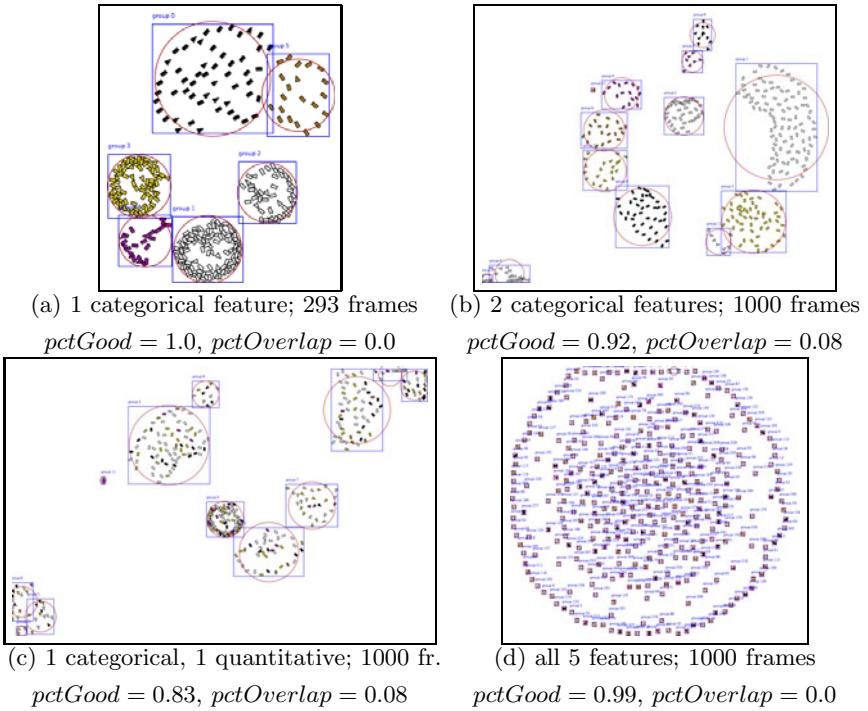


Fig. 1. Screen shots at end of 4 different runs, each clustering on different sets of features, as indicated above

a batch mode, using a 5-dimensional sample set of university enrollment data, gathered over a 4-and-a-half-year period. The data was partitioned into subsets, each representing one term; on average, there are 375 data records per term. Each record represents an instance of a student taking one class in a specific term. The fields in each record are: a unique identifier for the student (categorical feature), a course number identifying which class was taken (categorical feature), the student's gender (categorical feature), a numeric code indicating the student's ethnic origin (categorical feature), and the grade earned by the student in the class (quantitative feature). A subset of data for one term contains all the students who took classes in that term. A student who took more than one class in a given term is represented by multiple data records.

Figure 1 contains screen shots from 4 representative runs of the system, captured at the end of each run. The end of a run is either the point at which $pctGood = 1.0$ and $pctOverlap = 0.0$ or a maximum number of frames have elapsed. We used 1000 as the maximum number of frames for the runs presented here. The figure captions show the number of frames elapsed for each run, and the final values of $pctGood$ and $pctOverlap$. Figure 2 shows the improvement in $pctGood$ and $pctOverlap$ rates through the course of each corresponding run. When clustering on one categorical feature value, the system quickly converges

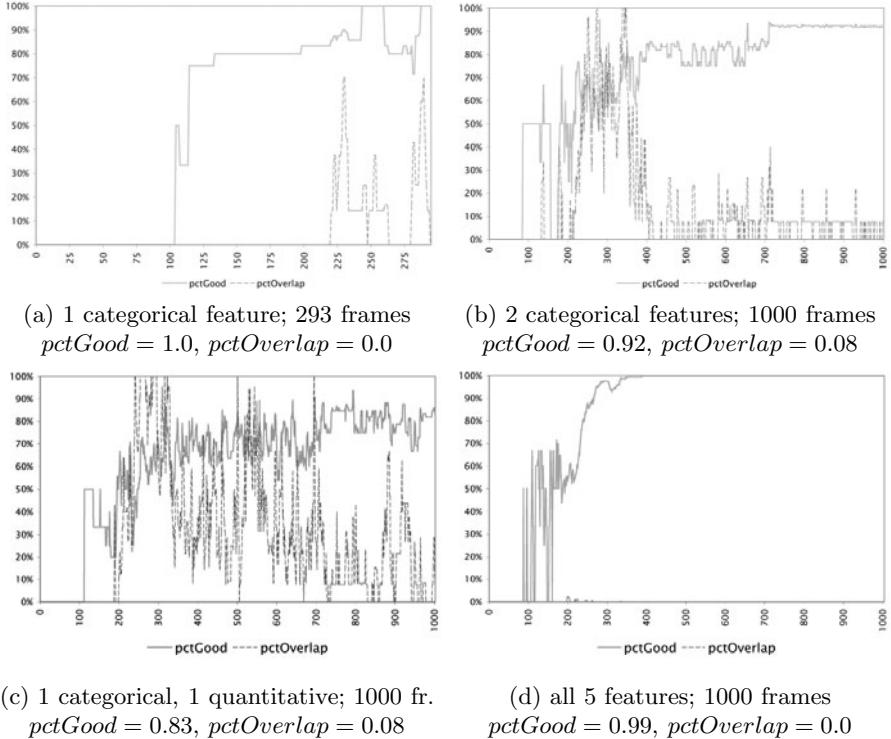


Fig. 2. Improvement in evolution metrics: $pctGood$ and $pctOverlap$

to the correct solution (in 293 frames); whereas in all the other cases, the system timed out after 100 runs. Even so, the percentage of “good” groups was high, ranging between 83% and 99%.

At the end of each run, we computed two scores to determine how well the clustering process worked: *within cluster* score and *between cluster* score. The “within cluster” score measures the disparity of the feature values in the cluster. The goal is to minimize the “within” cluster score and to maximize the “between” cluster score. The “within” cluster score is determined by calculating the feature distance (equation 7) from the representative center of the group in feature space (see the description of the meta-flocking step in Section 3) to each member of the group. The “between” cluster score is determined by calculating the average over all pairwise feature distances between representative (feature) centers of each group. Using these two metrics, we compared our results to two standard clustering algorithms from the data mining literature: *K-means* [10] and *Cobweb* [6]. We ran these algorithms using WEKA [19] on our same data set.

Figures 3 and 4 compare the results of our agent-based data visualization method to *K-means* and Cobweb. Each vertical bar represents an average over

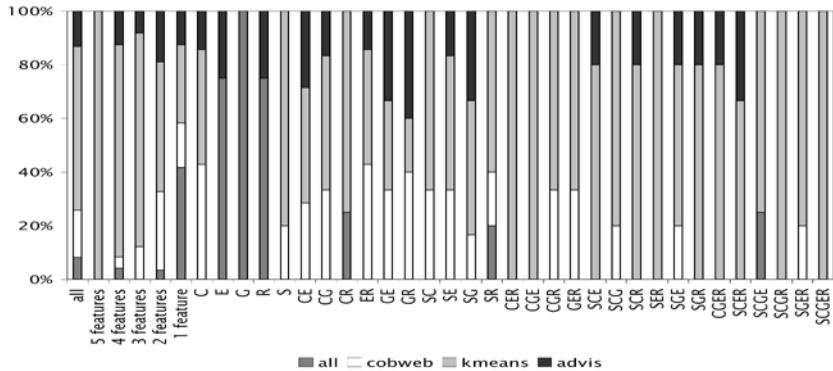


Fig. 3. “Within Cluster” score comparison

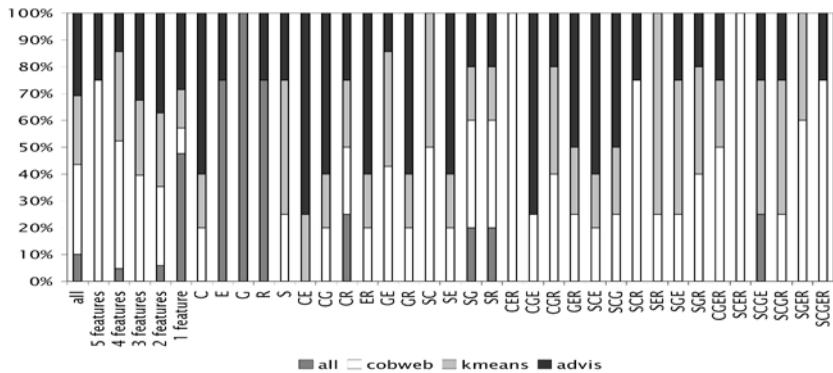


Fig. 4. “Between Cluster” score comparison

multiple runs², clustering on different combinations of features. The leftmost bar shows the average results over all combinations of features. The next 5 bars show the average results when clustering over 5, 4, 3, 2, and 1 feature(s) (in aggregate, regardless of which particular features). The remaining 31 bars show average results when clustering over specific combinations of all the various features. The shaded bands within each bar represent the distribution (by percentage) of best results for each clustering method. In general, *K*-means produces the best results for “within” cluster score more frequently than the other methods. For “between” cluster score, the results are split between Cobweb and our agent-based data flocking method. Table 1 quantifies the error rate for each method when it was not categorized as the “best” (see table caption for an explanation

² The number of runs varied between 4 and 31, depending on the number of features being clustered.

of how the error rate is computed). K -means produces the smallest error rate for “within” cluster score, whereas our method (labeled “advis”) produces the smallest error rate for “between” cluster score.

Table 1. The values in the table are are computed as follows: for each method (shown in columns), when it did not produce the best score, the error is calculated as the difference between that method’s score and the best score; this error is expressed as a percentage of the best score in order to give a sense of the proportion of the error

	K -means	cobweb	advis
within cluster	0.87%	38.23%	47.39%
between cluster	1.58%	2.18%	1.12%

5 Summary

We have described preliminary work in the development of an agent-based data mining and visualization method for clustering multi-dimensional data sets. Our technique extends early work in the area of information flocking and introduces several strategies that help the system converge on a clustering task. The first strategy involves “grouping” and “meta-level” flocking steps in which groups of data boids are identified and nudged toward each other, based on the similarity between feature values amongst the groups. The second strategy is a “braking” adjustment that causes the data boids to decrease their velocity as the system converges on good clusters. Experiments were conducted on a sample data set, showing promising results for “between cluster” scoring as compared with standard techniques. The advantage that our method has over standard techniques is the ability to adapt during clustering to changes in clustering criteria.

We plan to apply evolutionary machine learning techniques to evaluate various parameter settings, including weights (equation 4), “goodness” threshold and geographic neighborhood distance, in order to improve clustering results. Eventually, the aim is for the system to dynamically explore various combinations of features while clustering, learning to converge on the set of features that offer the best cluster scores. In addition, we also will apply our method to real-time data streams, where the feature values in the data set change while the clustering process is occurring. Because our method is highly dynamic, it should be able to respond in near real-time (depending on the relative speed at which new data comes in compared to the time it takes clusters to form).

Acknowledgments. We would like to thank the anonymous reviewers of this paper for their comments, which were not only helpful for the current revision but also provide guidance for the next stages of our work. This project was supported in part by the National Science Foundation (#CNS-0722177).

References

1. Aupetit, S., Monmarché, N., Slimane, M., Guinot, C., Venturini, G.: Clustering and Dynamic Data Visualization with Artificial Flying Insect. In: Cantú-Paz, E., Foster, J.A., Deb, K., Davis, L., Roy, R., O'Reilly, U.-M., Beyer, H.-G., Kendall, G., Wilson, S.W., Harman, M., Wegener, J., Dasgupta, D., Potter, M.A., Schultz, A., Dowsland, K.A., Jonoska, N., Miller, J., Standish, R.K. (eds.) GECCO 2003, Part I. LNCS, vol. 2723, pp. 140–141. Springer, Heidelberg (2003)
2. Butler, D.: Virtual globes: The web-wide world. *Nature* 439, 776–778 (2006)
3. Cao, L., Gorodetsky, V., Mitkas, P.A.: Agent mining: The synergy of agents and data mining. *IEEE Intelligent Systems* 24(3), 64–72 (2009)
4. Deneubourg, J.L., Goss, S., Franks, N., Sendova-Franks, A., Detrain, C., Chrétian, L.: The dynamics of collective sorting: Robot-like ants and ant-like robots. In: From Animals to Animats: 1st International Conference on Simulation of Adaptative Behaviour, pp. 356–363 (1990)
5. Dorigo, M., Maniezzo, V., Colomi, A.: The Ant System: Optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man and Cybernetics-Part B* 26(1), 1–13 (1996)
6. Fisher, D.H.: Knowledge Acquisition Via Incremental Conceptual Clustering. *Machine Learning* 2, 139–172 (1987)
7. Google: Google earth (2005), <http://earth.google.com>
8. Handl, J., Meyer, B.: Ant-based and swarm-based clustering. *Swarm Intelligence* 1(2), 95–113 (2007)
9. Lisle, R.J.: Google earth: a new geological resource. *Geology Today* (2006)
10. MacQueen, J.: Some methods for classification and analysis of multivariate observations. In: Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, pp. 281–297 (1967)
11. Moere, A.V.: Time-varying data visualization using information flocking boids. In: Proceedings of IEEE Symposium on Information Visualization, pp. 10–12 (2004)
12. Moere, A.V.: A model for self-organizing data visualization using decentralized multiagent systems. In: Prokopenko, M. (ed.) Advances in Applied Self-organizing Systems, Advanced Information and Knowledge Processing, Part III, pp. 291–324. Springer, Heidelberg (2008)
13. Picarougne, F., Azzag, H., Venturini, G., Guinot, C.: On data clustering with a flock of artificial agents. In: Proceedings of the 16th IEEE International Conference on Tools with Artificial Intelligence (ICTAI), pp. 777–778 (2004)
14. Picarougne, F., Azzag, H., Venturini, G., Guinot, C.: A new approach of data clustering using a flock of agents. *Evolutionary Computation* 15(3), 345–367 (2007)
15. Processing (2010), <http://www.processing.org/>
16. Proctor, G., Winter, C.: Information Flocking: Data Visualisation in Virtual Worlds Using Emergent Behaviours. In: Heudin, J.-C. (ed.) VW 1998. LNCS (LNAI), vol. 1434, pp. 168–176. Springer, Heidelberg (1998)
17. Reynolds, C.W.: Flocks, Herds and Schools: A Distributed Behavioral Model. In: International Conference on Computer Graphics and Interactive Systems, pp. 25–34 (1987)
18. Shannon, C.E.: A mathematical theory of communication. *The Bell System Technical Journal* 27, 379–423 (1948)
19. WEKA (2010), <http://www.cs.waikato.ac.nz/ml/weka/>
20. Wolfram, S.: Cellular automata as models of complexity. *Nature* 311, 419–424 (1984)
21. Xiaohui Cui, J.G., Potok, T.E.: A flocking based algorithm for document clustering analysis. *Journal of Systems Architecture* 52(8-9), 505–515 (2006)

A Multi-agent Based Approach to Clustering: Harnessing the Power of Agents

Santhana Chaimontree, Katie Atkinson, and Frans Coenen

Department of Computer Science, University of Liverpool, Liverpool, L69 3BX, UK
`{S.Chaimontree,katie,coenen}@liverpool.ac.uk`

Abstract. A framework for multi-agent based clustering is described whereby individual agents represent individual clusters. A particular feature of the framework is that, after an initial cluster configuration has been generated, the agents are able to negotiate with a view to improving on this initial clustering. The framework can be used in the context of a number of clustering paradigms, two are investigated: K-means and KNN. The reported evaluation demonstrates that negotiation can serve to improve on an initial cluster configuration.

Keywords: Multi-Agent Data Mining, Clustering.

1 Introduction

Data Mining and Multi-Agent Systems (MAS) are well established technologies which are finding increasing application. One of the current challenges of data mining is how to cope with the ever increasing size of the data sets that we wish to mine. A highlevel answer is to adopt and apply greater computational power. This can be achieved in a number of manners. One approach is to make use of distributed [9,11,13,26,27,31] or parallel [17,32,33] processing techniques so that several processors can be applied to the problem. This of course assumes that appropriate “farms” of processors are available. However, a more specific disadvantage is that of centralised control and lack of generality. Distributed and parallel data mining techniques typically assume a “master” process that directs the data mining task, therefore control is centralised at the master process and consequently these systems lack robustness. Further, distributed and parallel approaches tend to be directed at specific data mining applications and are difficult to generalise (because of the centralised control inherent to these systems). MAS offer an alternative to handling large quantities of data by harnessing the power of numbers of processors, with the added advantages that control is not centralised and consequently such systems can be much more robust and versatile.

A MAS is essentially a collection of software entities (agents) that are intended to cooperate in some manner so as to undertake some processing task. An important aspect of this cooperation is that the agents behave in an autonomous manner; they *negotiate* with one another to complete a given task rather than

being directed to do so by some master process. The idea of adopting MAS technology for data mining is therefore an attractive one. Multi-agent Data Mining (MADM) or Agent Assisted Data Mining (AADM) also allows for distributed data to be mined effectively without the need to first move the data into a data warehouse. This offers advantages where it is not easy or not possible for data to be first collated. MADM also supports the creation of frameworks that can be allowed to grow in an almost anarchic manner, agents can be easily incorporated into such frameworks as long as they comply with whatever protocols have been specified. The nature of these protocols remains a research issue. A number of Agent Communication Languages (ACLs) have been proposed but often these are difficult to fit to particular applications; it is therefore frequently necessary to extend or partially replace the set of performatives that are specified as part of these ACLs in order to taylor them to the specific scenario.

This paper describes an MADM framework, a set of agent communication performatives and supporting protocol, directed at unsupervised learning (clustering). The framework comprises four general categories of agent: user agents, data agents, clustering agents, validation agents (there are also house keeping agents, but these can be considered to be orthogonal to the task of data mining). Some of these agents are persistent while others are spawned as required, and have a lifetime equivalent to the duration of a given clustering task. To support the desired MADM communicates a dedicated set of performatives have been derived. A particular feature of the framework is that it supports negotiation between agents. This negotiation process allows for the enhancement of clusters once an initial cluster configuration has been established. This negotiation capability is the main contribution of the paper.

2 Previous Work

This section presents a review of some related work to that described in this paper. The section commences with some general background to MADM and then continues with a brief review of some parallel work on MAS clustering, highlighting the distinction between this work and the proposed MAS clustering approach.

There are two main paradigms for the interaction and integration between agent and data mining [5,4]: (i) data mining-driven agents which is the use of data mining to support the abilities of agents such as adaptation, coordination, learning, reasoning, etc. and (ii) agent-driven data mining, commonly known as Multi-Agent Data Mining (MADM), is the use of a collection of agents to perform data mining tasks. Surveys of agent-based distributed data mining can be found in [16,21,24].

PADMA [18] and PAPYRUS [2] are two of the earliest (late 1990s) reported multi-agent clustering systems. These systems aimed to achieve the integration of knowledge discovered from different sites with a minimum amount of network communication and a maximum amount of local computation. PADMA uses a facilitator (or coordinator) agent to direct interaction between the mining

agents. As such, it is based on a centralised architecture. PADMA agents are used to access local data and perform analysis. The *local clusters* are collected centrally to generate the *global clusters*. In addition, PADMA can be used to generate hierarchical clusters in the context of document categorisation. On the other hand, PAPYRUS differs from PADMA in that it is a distributed clustering system. PAPYRUS adopted a Peer-to-Peer model where both data and results can be moved between agents according to given MAS strategies. A more recent multi-agent clustering system is KDEC [20]. KDEC is a distributed density-based clustering algorithm also founded on the Peer-to-Peer model. In KDEC density estimation samples are transmitted, instead of actual data values so as to preserve data privacy and minimize communication between sites. A multi-agent clustering system directed at documents has been proposed in [29]; the objective here is to improve the accuracy and the relevancy of information retrieval processes. Kiselev et al. [19] proposed a clustering agent based system dealing with data streams in distributed and dynamic environments whereby input data sets and decision criteria can be changed at runtime (clustering results are available at anytime and are continuously revised to achieve the global clustering).

The above systems use agents to facilitate data privacy and support distributed data clustering (mining). There is little reported work on an agent based clustering systems that support intra-agent negotiation in order to refine (enhance) cluster configurations. In [1] Agogino et al. proposed an agent-based cluster ensemble approach to generate a best cluster configuration. Reinforcement learning, to maximise a utility function with respect to the original clustering results, is used to achieve the desired best clustering. However, the approach proposed in the paper operates in a different manner by using negotiation among agents to improved the quality of clustering result. It is argued that this approach harnesses the true power of agents.

The MADM framework described in this paper is founded on earlier work by the authors directed at multi-agent based clustering. This work is reported in [7,8]. The distinction between this early work and the current work is that the previous work did not feature any agent negotiation and thus, it could be argued, did not harness the full potential of MAS.

3 The MADM Framework

The proposed MADM framework, as noted above, comprises four categories of agent:

1. User agents.
2. Data agents.
3. Clustering agents.
4. Validation agents.

User agents are the interface between end users and the MADM environment. The agents are responsible for obtaining the input from the user, spawning clustering agents in order to perform the clustering task and presenting the derived

clustering result. To the above list of specific MADM agents we can also add a number of housekeeping agents that are utilised within the MADM framework.

Data agents are the “owners” of data sources. There is a one-to-one relationship between data agents and data sources. Data agents can be thought of as the conduit whereby clustering agents can access data.

Clustering agents are the “owners” of clusters. Groups of clustering agents can be thought of as representing a clustering algorithm. With respect to this paper the K-means [22] and K-Nearest Neighbour (KNN) [10] clustering algorithms have been adopted; however, our collections of clustering agents could have been configured to perform some alternative form of clustering (for example hierarchical clustering). A number of clustering agents will be spawned, as required, by a user agent in order to perform some clustering task. Thus, each clustering agent represents a cluster and is responsible for selecting a record from a data set and determining whether that record would belong to its cluster or not. The number of clustering agents, therefore depends on the number of clusters (K). In the case of the K-means algorithm the number of clusters is predefined; thus, by extension, the number of clustering agents that will be spawned will also be predefined. In the case of the KNN approach only one initial clustering agent will be spawned; then, as the KNN algorithm progresses further clustering agents may be created. Details concerning the operation of K-means and KNN with respect to the proposed MADM framework will be presented in Section 5. Clustering agents collectively have two principal functions: (i) initial generation of a “start” cluster configuration, and (ii) cluster refinement.

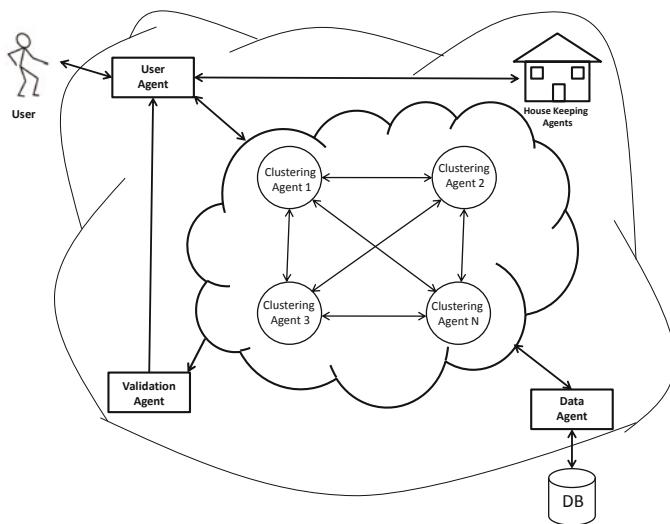


Fig. 1. Proposed MADM Framework

Validation agents are a special type of agent that perform validation operations on clustering results. Each validation agent is the “owner” of a technique for measuring the “goodness” of a given cluster configuration. In the current system validation agents consider either cluster cohesion or cluster separation or both.

A possible configuration for the proposed MADM framework, incorporating the above, is presented in Figure 1. The Figure includes a User Agent, a collection of Clustering Agents, a Data Agent, a Validation Agent and some house-keeping agents. The directed arcs indicate communication between agents. Note that communication can be bidirectional or unidirectional and that the Data Agent directly communicates with each Clustering Agent. Intra-communication between Clustering Agents takes follows a protocol that permits negotiation about cluster exchange to take place.

The framework has been realised using the Java Agent Development Environment (JADE) [3]. The house keeping agents provided by JADE, include the AMS (Agent Management System) agent and the DF (Directory Facilitator) agent. The AMS agent is responsible for managing and controlling the lifecycle of other agents in the platform, whereas the DF agent provides a “yellow pages” service to allow agents to register their capabilities.

4 Agent Communication within the Framework

The agents within our framework need to be able to communicate to carry out their tasks. JADE provides a communication mechanism that makes use of the FIPA ACL performatives [12]. However, as has been discussed previously in [23], the FIPA ACL has limited applicability to dialogues not involving purchase negotiations. Given that we wish to permit the agents in our system to engage in dialogues about the exchange of items between cluster configurations, we need a more expressive language to support this communication. As such, we have defined and implemented a set of performatives to enable our agents to engage in negotiations about the suitability of moving items/merging between clusters. At the highest level, the performatives are categorised as follows: holding a dialogue; performing the clustering task, negotiating about the movement of items between clusters; informing others about the clustering results.

The performatives are defined axiomatically in terms of the pre-conditions that must hold for an agent to be able to use the performatives and the post-conditions that apply following the use of the performatives. The agents use the performatives as part of a protocol that governs the exchange of information between the agents. For reasons of space we do not include here the details of the semantics of the performatives, but instead describe the communication protocol that the agents follow when using the communication language. We indicate in *italics* the performatives used at each stage.

A dialogue opens (*mining request*) which triggers a mining request to the data, cluster and validation agents to join. Once the recipient agents have entered the dialogue (*join dialogue*), the clustering task is performed (*inform data*). On

completion of the initial clustering, the agents evaluate the cohesion and separation of their clusters and as a result may broadcast to other agents that items be moved between clusters (*propose item move*). The recipients of the proposal can then reply by accepting (*accept item move*) or rejecting the proposed move (*reject item move*). We also permit retraction of a proposed item move (*retract item move*) if it is subsequently found to yield an unsuitable configuration. When the items have been moved between clusters and the agents are happy to accept the results, the clustering agents inform the validation agent of the new configurations (*inform cluster*). The overall cluster configuration is then sent to the user agent (*inform cluster configuration*), after which moves are made for agents to exit the dialogue (*leave dialogue*) and subsequently end it (*close dialogue*).

The way in which the reconfiguration of clusters happens is dependant upon the clustering algorithm used, as will be described in the next section.

5 Operation

The operation of the proposed MADM clustering mechanism is described in this section. We have elected to describe the operation in a procedural manner as we believe this will facilitates understanding (although readers should be clear that the operation is anything but procedural). As noted in the foregoing, Clustering Agents are spawned by a User Agent according to the nature of the end user's initial "clustering request". Fundamentally there are two strategies for spawning Clustering Agents: the K-means strategy and the KNN strategy. In the K-means strategy the user pre-specifies the number of clusters, K , that are required; in which case K Clustering Agents are spawned. In the case of the KNN strategy the MADM process decides how many clusters are required, thus initially only one Clustering Agent is spawned (more may be generated later as required).

Table 1. Bidding phase founded on K-means Spawning Strategy

Phase I: Bidding using K-means

Input: Dataset ($D = \{d_1, d_2, \dots, d_n\}$), the desired number of clusters (K)

Output: An initial clustering configuration

1. User Agent spawns K Clustering Agents ($C = \{c_1, c_2, \dots, c_K\}$)
 2. Each Clustering Agent sends a data request to the indicated Data Agent
 3. Data Agent sends first K records ($\{d_1, d_2, \dots, d_K\}$) to the K Clustering Agents; d_1 to c_1 , d_2 to c_2 , and so on.
 4. Each Clustering Agent calculates its cluster centroid
 5. $\forall d_i \in D$ ($i = K + 1$ to n)
 6. $\forall c_j \in C$ ($j = 1$ to K)
 7. $bidDistance = d_i - centroid c_j$
 8. Allocate d_i to c_j so as to minimise $bidDistance$
-

The operation of the proposed MADM clustering mechanism comprises two phases: a bidding phase and a refinement phase. During the bidding phase

Clustering Agents compete for records by “bidding” for them in an “auction” setting where the Data Agent acts as the auctioneer. For each record the Clustering Agent that poses the best bid wins the record and includes it in its cluster (see Sub-sections 5.1 and 5.2). During the refinement phase each Clustering Agents tries to pass unwanted records (records that no longer fit within its cluster definition) to other agents. This can also be conceptualised in terms of an auction; each Clustering Agent acts as a local auctioneer and tries to sell its unwanted records by inviting other clustering agents to bid for them. The operation of the refinement phase is entirely independent of the spawning strategy adopted. However the operation of the bidding phase differs according to the nature of the spawning strategy.

The rest of this section is organised as follows. In Sub-sections 5.1 and 5.2 the operation of the bidding phase with respect to the K-means and KNN spawning strategies are described. Sub-section 5.3 then describes the operation of the refinement phase.

Table 2. Bidding phase founded on KNN Spawning Strategy

<i>Phase I: Bidding using KNN</i>
Input: Dataset ($D = \{d_1, d_2, \dots, d_n\}$), threshold t
Output: An initial clustering configuration
1. User Agent spawns a single Clustering Agent (c_1)
2. $K = 1$
3. $\forall d_i \in D$ ($i = 2$ to n)
4. $\forall c_j \in (j = 1$ to $K)$
5. $bidDistance = \text{nearest neighbour to } d_i$
6. IF $\exists c_j \in C$ such that $bidDistance < t$, allocate d_i to c_j so as to minimise $bidDistance$
7. ELSE $K = K + 1$, spawn Clustering Agent c_K , allocate d_i to c_K

5.1 Biding Phase Founded on the K-means Spawning Strategy

The operation of the bidding process with respect to the K-means strategy is presented in Table 1. K clustering agents are spawned to represent the clusters (line 1). Each Clustering Agent is then allocated a single record, by the identified Data Agent, and this record is used to represent the centroid of each cluster (lines 2 to 4). The clustering agents then bid for the remaining records in D (lines 5 to 8). In the current implementation the $bidDistance$ equates to the “distance” of d_i to the nearest neighbour of d_i in the cluster. At the end of the process the K clustering agents will collectively hold an initial cluster configuration. Note that, in common with standard K-means clustering, the “goodness” of this initial configuration is very much dependent on the nature of the first K records selected to define the initial clusters. It should also be noted that the operation

of K-means bidding strategy described here should not be confused with the K-means clustering algorithm. The K-means bidding strategy is a “one pass” process, the K-means clustering algorithm features an iterative process.

5.2 Biding Phase Founded on the KNN Spawning Strategy

The biding phase founded on the KNN spawning strategy, as noted above, commences with a single Clustering Agent (C_i). The operation of this bidding process is presented in Table 2. Note that the process requires a *nearest neighbour threshold*, t , as a parameter. The threshold is used to determine the “nearest neighbour”. If the *bidDistance* is less than the threshold, this record in question is allocated to the “closest” cluster (line 6). If there is no “closest” cluster a new Clustering Agent is spawned (line 7). Note that the chosen value of t can significantly affect the number of Clustering Agents that are spawned. The authors proposed a method to identify the most appropriate value for t in [6].

5.3 Refinement (Negotiation) Phase

The refinement process is presented in Table 3. The refinement process is driven using individual cluster cohesion values and an overall cluster configuration separation value. We wish to generate a configuration which minimises the cohesion values and maximises the separation value. The refinement phase commences (line 1) with each Clustering Agent determining its own cluster cohesion value and the set of Clustering Agents collectively determining a separation value. Cluster cohesion (the compactness of a cluster) can be determined, by each clustering agent, simply by considering the distance between the members of its cluster. In order to determine the degree of separation (the distinctiveness of each cluster with respect to each other cluster) agents must communicate with one another. For this latter purpose each Cluster Agent defines its cluster in terms of its centroid and these values are then transmitted to all other clustering agents so that an overall separation value can be determined. With respect to the framework described in this paper the Within Group Average Distance (WGAD) and the Between Group Average Distance (BGAD) metrics were adopted to determine cluster cohesion and separation respectively (alternative methods could equally well have been adopted). These metrics are described in Section 6. The cohesion and separation values are sufficient if the cohesion values are below a specified cohesion threshold and the separation value is above a pre-specified separation threshold. If this is not the case the cluster associated with each Clustering Agent (c_i) will be split into two sub-clusters: a *major sub-cluster* and a *minor sub-cluster*. The splitting is achieved by applying a standard K-means algorithm (with K set to 2) to the records held by c_i . The cluster with the smallest number of records is then designated to be the minor sub-cluster and these records are then put up for auction. The auction proceeds in a similar manner to that described for the bidding phase founded on the K-means strategy (see Sub-section 5.1) with the distinction that the WGAD value is used as the *bidDistance*. This process repeats until satisfactory cohesion and separation

values are reached. Note (line 8) that on any given iteration, if a record cannot be allocated to a class (because its WGAD value is too high) it is allocated to an *outlier cluster*.

Table 3. Algorithm for refinement phase

Algorithm Phase II: Refinement

Input: A set of clusters

Output: An improved clustering result

1. For all clusters calculate cluster cohesion and separation values
 2. DO WHILE there exists cluster cohesion value > cohesion threshold
or cluster separation value < separation threshold
 3. $\forall c_i \in C$ ($i = 1$ to K)
 4. Split a cluster c_i into two sub-clusters, c_{major} and c_{minor} using K-means
 5. $\forall d \in c_{minor}$
 6. $\forall c_j \in C$ ($j = 1$ to K and $j \neq i$)
 7. $bidDistance = WGAD_j$ (see Section 6)
IF $\exists c_j \in C$ such that $bidDistance < cohesion threshold$, allocate d to c_j so as to minimise $bidDistance$
 8. ELSE Allocate d to “outlier cluster”
 9. IF no “successful” bids end loop
-

6 Cluster Configuration Metrics

In order for agents to bid for examples some appropriate metric must be adopted. The process for deciding when and how to move a record also requires recourse to some metric, as does deciding when to split and merge clusters. Essentially there are three ways of measuring the “goodness” of a proposed cluster configuration. We can measure intra-cluster cohesion, we can measure inter-cluster separation or we can adopt both metrics. This section presents a review of some of the metrics that may be used to measure the goodness of a cluster configuration and the metrics used in the context of the work described in this paper.

In context of the work described the authors have adopted the Within Group Average Distance (WGAD) [28] and the Between Group Average Distance (BGAD) metrics:

$$WGAD = \frac{\sum_{i=1}^{|C|} dist(x_i, centroid)}{|C|} . \quad (1)$$

$$BGAD = \sum_{i=1}^{i=K} dist(centroid_i, c) . \quad (2)$$

where $|C|$ is the number of objects in a cluster (i.e. the size of a cluster), $centroid$ is the cluster centroid, $dist(x_i, centroid)$ is the distance between object x_i and the

cluster centroid, and $dist(centroid_i, c)$ is the distance between cluster centroid $centroid_i$ and the overall centroid of cluster configuration (c). K is the number of clusters.

The lower the WGAD value the greater the cohesiveness of the cluster, whereas the higher the BGAD value the greater the separation of the clusters from one another. We wish to minimise the WGAD and maximise the BGAD to achieve a best cluster configuration. The target WGAD value, the cohesion threshold, is the average WGAD across the identified set of clusters multiplied by a factor p ($0 < p < 1.0$). The target BGAD value, the separation threshold, is the BGAD value for the initial cluster configuration multiplied by a factor q ($1.0 < q < 2.0$). Our experiments indicate that values of $p = 0.8$ and $q = 1.2$ tend to produce good results.

Table 4. Comparison of the result accuracy provided by K-means task distribution before and after cluster configuration improvement

No.	Data Set	Num Classes	Accuracy		Cohesion	Separation
			Phase I	Phase II	threshold	threshold
1	Iris	3	0.89	0.97	1.41	2.03
2	Zoo	7	0.76	0.78	1.94	1.95
3	Wine	3	0.68	0.70	204.95	296.73
4	Heart	2	0.55	0.59	33.87	106.28
5	Ecoli	8	0.76	0.80	0.42	0.54
6	Blood Transfusion	2	0.76	0.76	794.16	4582.29
7	Pima Indians	2	0.65	0.66	65.08	290.60
8	Breast cancer	2	0.79	0.85	283.16	1729.93

7 Evaluation

To evaluate our approach we experimented with a selection of data sets taken from the UCI machine learning repository [14]. We compared the operation of our MADM approach with the well known K-means [22] and KNN [15] clustering algorithms. In each case we recorded the accuracy of the clustering operation, with respect to the known (*ground truth*) clustering. For the K-means algorithm the number of desired clusters must be pre-specified in order to spawn an appropriate number of clustering agents. For this purpose we have used the number of classes given in the UCI repository. The t parameter used for KNN was selected, for evaluation purposes, according to the nature of the input so as to be compatible with the specified number of clusters. The results using K-means and KNN are reported in Tables 4 and 5 respectively. The columns in the tables describe: the number of identified clusters (classes), the accuracy after the initial Phase I clustering (i.e. the accuracy that would be achieved using K-means or KNN without any further negotiation), the accuracy after refinement (Phase II), and the calculated cohesion and separation thresholds. Accuracy values are calculated as follow:

$$Accuracy = \frac{\sum_{i=1}^{i=K} C_i}{m} . \quad (3)$$

Where K is the number of clusters, m is the number of records and C_i is the size (in terms of the number of records) of the majority class for cluster i . It should be noted firstly that the ground truth is only used here to evaluate the outcomes. Secondly it is the relative performance that is important, better absolute clustering accuracies with respect to some of the above datasets, have been reported in the literature.

Table 5. Comparison of the result accuracy provided by KNN task distribution before and after cluster configuration improvement

No.	Data Set	Num Classes	t		Accuracy Phase I	Accuracy Phase II	Cohesion threshold	Seperation threshold
			1	2				
1	Iris	4	0.99	0.84	0.93	1.90	1.93	
2	Zoo	7	2.24	0.82	0.73	2.42	2.55	
3	Wine	3	164.31	0.65	0.65	281.49	282.98	
4	Heart	2	115.29	0.55	0.64	21.96	62.51	
5	Ecoli	7	0.42	0.64	0.67	0.38	0.26	
6	Blood Transfusion	2	3500.83	0.76	0.76	1222.54	3090.16	
7	Pima Indians	2	149.08	0.65	0.65	133.77	152.08	
8	Breast cancer	2	826.75	0.63	0.84	205.51	847.98	

From the tables it can be seen that in the majority of cases (shown in bold font) agent negotiation serves to enhance the initial clustering, with the K-means approach tending to outperform the KNN approach. Interestingly, in the case of the Zoo data set when using the KNN approach, negotiation had an adverse effect; the research team conjecture that this is something to do with the large number of classes compared to the number of records. The zoo data set has a very small number of records, 101 in total, thus an average of 14.4 records per class. Closer inspection of the data set indicates that three of the classes have less than 10 records. In the remaining cases the situation remained unchanged, either because a best configuration had been identified immediately, or because the WGAD and BAGD values could not be reduced). It should also be noted that the number of class values given in Table 4 (column three) are the ground truth values; the KNN approach does not always produce the correct number of classes and hence this is why the accuracy values are not always as good as in the case of the K-means approach.

8 Conclusion

A MADM framework to achieve multi-agent based clustering has been described. A particular feature of the framework is that it enables agents to negotiate so

as to improve on an initial clustering. The framework can be used in a number of ways. Two approaches were considered: K-means and KNN. Evaluation using a number of datasets taken from the UCI repository indicates that in most cases (and especially when using the K-means approach) a better clustering configuration can be obtained as a result of the negotiation process. The results obtained thus indicate the benefits that can be realised as the result of the negotiation process.

Future work is directed at two areas. First the enhancement and further testing of the current capabilities of the framework so as to address as many clustering mechanisms as possible (for example hierarchical clustering [30] or branch and bound [25] approaches). A system that can effectively and efficiently compare the operation of many different clustering techniques will provide a useful research tool in its own right. The second is stress testing of the framework using large data sets (10,000 records plus) to determine: (i) what the capacity of the framework is in terms of the size of the data to be clustered (defined in terms of the number of clusters, the number of attributes and the number of records) and (ii) the complexity of the system in terms of the messaging overhead. The messaging overhead is considered significant in the context of distributed and parallel data mining directed at the mining of very large data sets. However, in the case of MADM, the author do not consider this to be significant as the objective of MADM is also to provide an easily extendable framework, that features decentralised control, and whose components behave in an autonomous manner.

References

1. Agogino, A., Tumer, K.: Efficient agent-based cluster ensembles. In: Proceedings of the 5th International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2006, pp. 1079–1086. ACM, New York (2006)
2. Bailey, S., Grossman, R., Sivakumar, H., Turinsky, A.: Papirus: A system for data mining over local and wide area clusters and super-clusters. IEEE Supercomputing (1999)
3. Bellifemine, F., Bergenti, F., Caire, G., Poggi, A.: JADE: a java agent development framework. In: Bordini, R.H. (ed.) Multi-agent Programming: Languages, Platforms, and Applications, p. 295. Springer, New York (2005)
4. Cao, L., Gorodetsky, V., Mitkas, P.A.: Agent mining: The synergy of agents and data mining. IEEE Intelligent Systems 24(3), 64–72 (2009)
5. Cao, L., Gorodetsky, V., Mitkas, P.A.: Guest editors' introduction: Agents and data mining. IEEE Intelligent Systems 24(3), 14–15 (2009)
6. Chaimontree, S., Atkinson, K., Coenen, F.: Best Clustering Configuration Metrics: Towards Multiagent Based Clustering. In: Cao, L., Feng, Y., Zhong, J. (eds.) ADMA 2010, Part I. LNCS, vol. 6440, pp. 48–59. Springer, Heidelberg (2010)
7. Chaimontree, S., Atkinson, K., Coenen, F.: Clustering in a Multi-Agent Data Mining Environment. In: Cao, L., Bazzan, A.L.C., Gorodetsky, V., Mitkas, P.A., Weiss, G., Yu, P.S. (eds.) ADMI 2010. LNCS, vol. 5980, pp. 103–114. Springer, Heidelberg (2010)
8. Chaimontree, S., Atkinson, K., Coenen, F.: Multi-Agent Based Clustering: Towards Generic Multi-Agent Data Mining. In: Perner, P. (ed.) ICDM 2010. LNCS, vol. 6171, pp. 115–127. Springer, Heidelberg (2010)

9. Coenen, F., Leng, P., Ahmed, S.: T-trees, vertical partitioning and distributed association rule mining. In: Proceedings of the 3rd IEEE International Conference on Data Mining, ICDM 2003, pp. 513–516. IEEE Computer Society, Washington, DC, USA (2003)
10. Dasarathy, B.V.: Nearest neighbor (NN) norms: NN pattern classification techniques. IEEE Computer Society Press, Las Alamitos (1991)
11. Dasilva, J., Giannella, C., Bhargava, R., Kargupta, H., Klusch, M.: Distributed data mining and agents. *Engineering Applications of Artificial Intelligence* 18(7), 791–807 (2005)
12. FIPA: Communicative Act Library Specification. Tech. Rep. XC00037H, Foundation for Intelligent Physical Agents (2001), <http://www.fipa.org>
13. Forman, G., Zhang, B.: Distributed data clustering can be efficient and exact. ACM SIGKDD Explorations Newsletter 2, 34–38 (2000)
14. Frank, A., Asuncion, A.: UCI machine learning repository (2010), <http://archive.ics.uci.edu/ml>
15. Fukunaga, K.: Introduction to Statistical Pattern Recognition. Academic Press, New York (1972)
16. Giannella, C., Bhargava, R., Kargupta, H.: Multi-agent Systems and Distributed Data Mining. In: Klusch, M., Ossowski, S., Kashyap, V., Unland, R. (eds.) CIA 2004. LNCS (LNAI), vol. 3191, pp. 1–15. Springer, Heidelberg (2004)
17. Kargupta, H., Chan, P. (eds.): Advances in Distributed and Parallel Knowledge Discovery. MIT Press, Cambridge (2000)
18. Kargupta, H., Hamzaoglu, I., Stafford, B.: Scalable, distributed data mining using an agent based architecture. In: Proceedings the 3rd International Conference on the Knowledge Discovery and Data Mining, pp. 211–214. AAAI Press (1997)
19. Kiselev, I., Alhajj, R.: A self-organizing multi-agent system for online unsupervised learning in complex dynamic environments. In: Proceedings of the 23rd AAAI Conference on Artificial Intelligence, pp. 1808–1809. AAAI Press (2008)
20. Klusch, M., Lodi, S., Moro, G.: Agent-Based Distributed Data Mining: The KDEC Scheme. In: Klusch, M., Bergamaschi, S., Edwards, P., Petta, P. (eds.) Intelligent Information Agents. LNCS (LNAI), vol. 2586, pp. 104–122. Springer, Heidelberg (2003)
21. Klusch, M., Lodi, S., Moro, G.: The role of agents in distributed data mining: Issues and benefits. In: IAT 2003: Proceedings of the IEEE/WIC International Conference on Intelligent Agent Technology, p. 211. IEEE Computer Society, Washington, DC, USA (2003)
22. MacQueen, J.B.: Some methods for classification and analysis of multivariate observations. In: Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability, pp. 281–297 (1967)
23. McBurney, P., Parsons, S., Wooldridge, M.: Desiderata for agent argumentation protocols. In: Castelfranchi, C., Johnson, W.L. (eds.) Proceedings of the 1st Int. Conf. on Autonomous Agents and Multi-Agent Systems (AAMAS 2002), pp. 402–409. ACM Press, New York (2002)
24. Moemeng, C., Gorodetsky, V., Zuo, Z., Yang, Y., Zhang, C.: Agent-based distributed data mining: A survey. In: Cao, L. (ed.) Data Mining and Multi-agent Integration, pp. 47–58. Springer, US (2009)
25. Newman, M.E.J., Girvan, M.: Finding and evaluating community structure in networks. *Phys. Rev. E* 69(2), 1–15 (2004)
26. Park, B.H., Kargupta, H.: Distributed data mining: Algorithms, Systems, and Applications. In: Data Mining Handbook, pp. 341–358. IEA (2002)

27. Provost, F.: Distributed data mining: Scaling up and beyond. In: Advances in Distributed and Parallel Knowledge Discovery, pp. 3–27. MIT Press (1999)
28. Rao, M.: Clustering analysis and mathematical programming. *Journal of the American Statistical Association* 66(345), 622–626 (1971)
29. Reed, J.W., Potok, T.E., Patton, R.M.: A multi-agent system for distributed cluster analysis. In: Proceedings of the 3rd International Workshop on Software Engineering for Large-Scale Multi-Agent Systems (SELMAS 2004) W16L Workshop - 26th International Conference on Software Engineering, pp. 152–155. IEE, Edinburgh (2004)
30. Xu, R., Wunsch, D.: Clustering. Wiley/IEEE Press (2009)
31. Younis, O., Fahmy, S.: Distributed clustering in ad-hoc sensor networks: a hybrid, energy-efficient approach. In: 23rd Annual Joint Conf. of the IEEE Computer and Communications Societies, INFOCOM 2004, vol. 1, pp. 629–640 (2004)
32. Zaki, M.J., Ho, C.-T. (eds.): KDD 1999. LNCS (LNAI), vol. 1759. Springer, Heidelberg (2000)
33. Zaki, M.J., Pan, Y.: Introduction: Recent developments in parallel and distributed data mining. *Distributed Parallel Databases* 11, 123–127 (2002)

Agent Enriched Distributed Association Rules Mining: A Review

G.S. Bhamra¹, A.K. Verma², and R.B. Patel³

¹M.M. Institute of Computer Technology and Business Management, MMU,
Mullan-133203, Haryana, India
bhamra_gs@gmail.com

²Department of Computer Science & Engineering, TIET, Thapar University,
Patiala-147004, Punjab, India
akv_1001@yahoo.com

³Department of Computer Sc. & Engineering, DCR University of Sc. and Tech.,
Murthal, Haryana -131039, India
patel_r_b@yahoo.com

Abstract. Distributed Data Mining (DDM) is concerned with application of the classical Data Mining (DM) approach in a Distributed Computing (DC) environments so that the available resource including communication networks, computing units and distributed data repositories, human factors etc. can be utilized in a better way and on-line, real-time decision support based distributed applications can be designed. A Mobile Agent (MA) is an autonomous transportable program that can migrate under its own or host control from one node to another in a heterogeneous network. This paper highlights the agent based approach for mining the association rules from the distributed data sources and proposed an another framework called Agent enriched Mining of Strong Association Rules (AeMSAR) from Distributed Data Sources. As agent technology paradigm of the DC has gained lots of research in the recent years, therefore, making an alliance of agent and Association Rules Mining(ARM) will help mining the Association rules in a Distributed environment in a better way.

Keywords: Data Mining, Distributed Data Mining, Association Rule Mining, Mobile Agent.

1 Introduction

The exponential growth in data collection in business and scientific fields results into a need to analyze and mine useful knowledge from it. Data Mining (DM) is the process of automatically extracting some interesting data patterns or trends representing knowledge, implicitly stored in large databases [1, 2]. In a classical knowledge discovery technique in distributed environment, a single central repository called Data Warehouse (DW) is created and then DM tools are used to mine the data and extract the knowledge [3]. This approach, however, is ineffective or infeasible for a number of reasons [4] like (a) Storage, Computational and Communication cost involved to handle and store the data form the ever increasing and updated distributed

resources; (b) Undesirable central collection and utilization of privacy-sensitive data by the business organizations; (c) Performance and Scalability; (d) Resource constraints issues of distributed and mobile environment are not considered properly in central DW based DM [5].

In a DC environment, distributed data sources add a new dimension. It is evident, therefore, that traditional data mining model is insufficient while dealing with today's distributed environments and applications mining the privacy-sensitive data. Advanced analysis of distributed data for mining the useful knowledge is the obvious step in today's ubiquitous and DC scenario. DM environment is a combination of users, data, hardware and the mining software. DDM addresses the impact of distribution of all these components on the data mining process.

This paper makes an effort to explore the possible synergy between the Multi Agent System (MAS) and DDM technology [45]. It particularly focuses on Distributed Association Rules Mining (DARM), a problem finding number of applications in distributed information retrieval and many other domains. This paper illustrates the ideas and reviews existing work in this area.

Rest of the paper is organized as follows. Section 2 describes the DDM architecture and the issues involved, Distributed Association Rule Mining is discussed in Section 3, Mobile Agent technology and its advantages and the role of mobile agents in DDM is described in section 4, Existing agent based frameworks for ARM are explored in section 5, architecture of the proposed framework is explained in section 6 and finally article is concluded in Section 7.

2 Distributed Data Mining

Parallel Knowledge Discovery (PKD) and Distributed Knowledge Discovery (DKD) are the direct outcome of the issues involved in centralized DW based DM. DDM is the related pattern extraction problem in DKD. It is concerned with application of classical DM procedures in a DC environment trying to make the best of the available resources. In DDM, DM takes place both locally at each geographically distributed site and at a global level where the local knowledge is merged in order to discover global knowledge [3, 4, 6, 7]. The main aim of DDM is to get the global company-wide knowledge for decision making (planning, marketing and sales etc.) from the local operational data at distributed sites [7]. Broadly, DM environment is a combination of users, data, hardware and the mining software. DDM addresses the impact of distribution of all these components on the data mining process. DDM techniques are scalable in that its performance does not degrade much with the increase of data set size and the number of distributed sites involved. The overall performance of such system remains stable.

Byung-Hoon Park and Hillol Kargupta [3] have given a DDM architecture where processing at the different distributed nodes generates several local models which are then aggregated to form a global model representing the global knowledge. DM

operations are performed based on the type and availability of the distributed resources. Grigoris Tsoumakas and Ioannis Vlahavas [4] have shown different phases in a typical architecture of a DDM approach. In the first phase local distributed databases are analyzed. Then, the discovered knowledge is transmitted to a merger site, where all the distributed local models are integrated. The global knowledge is then transmitted back to update the distributed databases. In some cases, instead of having a merger site, the local models are broadcasted to all other sites, so that each site can compute the global model in parallel.

Fu, Y. [7] suggested some of the issues that need to be addressed while developing DDM systems and algorithms. These issues involved data fragmentation, data replication, adaptation on heterogeneity of the databases, integration of the interestingness property of the local data, privacy preservation of the local data etc.

3 Distributed Association Rule Mining

Frequent itemsets are the set of items purchased more frequently in transactional databases (basket data). Associations, correlations, and many other interesting relationships among itemsets can be generated using frequent itemsets. It has an elegantly simple problem statement: to find the set of all subsets of items that frequently occur together in database records or transactions. Although this task has a simple statement, it is CPU and input/output (I/O) intensive, mainly because the large amount of itemsets that are typically generated and large size of the datasets involved in the process[2,8,9].

3.1 Preliminaries and Definitions for Frequent Itemsets Mining (FIM)

Frequent Itemsets Mining (FIM) can be formally defined as follows:

Transactional Data Set $DB = \{T_i, i = 1 \dots D\}$: Set of total D number of transactions (tuple), where each T_i is assigned an identifier (TID).

Itemset $I = \{d_i, i = 1 \dots n\}$: Set of total n data items in DB

k -Itemset $P = \{d_i, i = 1 \dots k\}$: Set of k data items in a particular transaction T , $P \subseteq I$.

Support of an itemset $s(P) = \frac{\# \text{ of } T \text{ containing } P}{D} \%$: The frequency of occurrence of itemset P in DB , where $\# \text{ of } T \text{ containing } P$ is the support count (sup_count) of itemset P .

Frequent k -Itemset: An itemset that appear in DB frequently ,i.e., If $s(P) \geq \text{min_sup}$ (given minimum support threshold), then P is a Frequent k -Itemset.

Candidate frequent k-Itemsets C_k : the list (or set) of all frequent k-itemsets without the constraint of minimum support threshold, min_sup.

List (or set) of frequent k-itemsets L_k : List of frequent k-itemsets after pruning the candidate set C_k by applying the constraint of min_sup.

3.2 Preliminaries and Definitions for Distributed Frequent Itemsets Mining (DFIM)

DFIM is the task of mining Global Frequent k-Itemsets (L_k^{GFI}) from the distributed Local Frequent k-Itemsets (L_k^{LFI}) in a distributed environment. The frequent itemset mining in a distributed environment can be formally defined as follows:

Distributd Sites $S = \{S_i, i = 1 \dots n\}$: Set of n number of sites in a distributed system.

Central Site $S_{CENTRAL}$: Central Site where Global Knowledge is computed.

Local Transactional Data Set DB_i : Transactional Data Set at the local site S_i with size D_i

Distributed Transactional Data Set $DB = \{DB_i, i = 1 \dots n\} = \bigcup_{i=1}^n DB_i$: Distributed horizontally partitioned Transactional Data Set across multiple sites with size

$$D = \bigcup_{i=1}^n D_i$$

Local Frequent k-Itemsets $L_{k(i)}^{LFI}$: Local Frequent k-Itemsets at site S_i

List of support count $L_{k(i)}^{LFISC}$: List of support count of every items in $L_{k(i)}^{LFI}$.

Total Frequent k-Itemsets $L_k^{TFI} = \bigcup_{i=1}^n L_{k(i)}^{LFI}$: Total Frequent k-Itemsets and is the union of all the local frequent k-itemsets.

Global Frequent k-Itemsets $L_k^{GFI} = \bigcap_{i=1}^n L_{k(i)}^{LFI}$: Global Frequent k-Itemsets and is the intersection of all the Local frequent k-itemsets.

Local support count X_{sup}^i : Support count of an Itemset X in DB_i .

Global support count X_{sup} : Support count of an Itemset X in DB -

3.3 Association Rules

Association Rules, first introduced in [10], are used to discover the associations among items in a transactional database to find the purchase patterns of the customers in order to design the goods shelves.

Let P be a k-itemset and Q be a j-itemset, $P \subset I$, $Q \subset I$, and $P \cap Q = \emptyset$. Association Rule (AR) is an implication of one of the following forms $P \Rightarrow Q$ (support, confidence) where:-

- P is the antecedent part and Q is called the consequent part of AR.
- support and confidence are two measures to find interesting Association Rules.
- support

$$s(P \Rightarrow Q) = p(P \cup Q) = \frac{\# \text{ of } T \text{ containing both } P \text{ and } Q}{D}$$

% : the probability of both P and Q appearing in T, we can say that $s\%$ of the transactions support the rule $P \Rightarrow Q$, $0 \leq s \leq 1.0$

- confidence

$$c(P \Rightarrow Q) = p(Q|P) = \frac{s(P \Rightarrow Q)}{s(P)} = \frac{\text{sup_count}(P \Rightarrow Q)}{\text{sup_count}(P)} : \text{ Conditional probability of } Q \text{ given } P, \text{ we can say that when itemset } P \text{ occurs in a transaction there are } c\% \text{ chances that itemset } Q \text{ will occur in that transaction, } 0 \leq c \leq 1.0$$

Strong Association Rules: An Association rule $P \Rightarrow Q$ is said to be strong if

1. $s(P \Rightarrow Q) \geq \text{min_sup}$, i.e., support of the rule is greater than or equal to the given minimum threshold support and
2. $c(P \Rightarrow Q) \geq \text{min_conf}$, i.e., confidence of the rule is greater than or equal to the given minimum threshold confidence

3.4 Association Rule Mining(ARM)

ARM is the task to find all the strong association rules from the frequent itemsets. The ARM can be viewed as a two-step process[12].

1. Find all frequent k-itemsets(L_k)
2. Generate Strong Association Rules from L_k
 - a) For each frequent itemset $l \in L_k$, generate all non empty subsets of l .

- b) For every non empty subset s of l , output the rule “ $s \Rightarrow (l - s)$ ” if $\frac{\text{sup_count}(l)}{\text{sup_count}(s)} \geq \text{min_conf}$, where min_conf is minimum threshold confidence.

3.5 Distributed Association Rule Mining (DARM)

Distributed Association Rule Mining (DARM) task is to find all the strong association rules in a distributed environment. The DARM can also be viewed as a two-step process.

1. Find the Global frequent k-itemset (L_k^{GFI}) from the distributed Local Frequent k-Itemsets $L_{k(i)}^{LFI}$ from partitioned datasets.
2. Generate globally strong Association Rules from L_k^{GFI}

The state of the art in parallel and distributed association rule mining can be found in a survey report by Mohammed J. Zaki [32].

The notations and terminologies used for DARM are similar to regular (non-distributed) mining with some extensions described by Gongzhu Hu and Shaozhen Ding [35,36] and U. P. Kulkarni et. al. [41,42].

4 Mobile Agents for DDM

Software agents refer to intelligent program that performs certain tasks on behalf of the user and acts as a personal assistant. Software agents endowed with the property of mobility are called mobile agents. Mobile Agent (MA) [14,15,16,17,18] is an autonomous transportable program that can migrate under its own or host control from one node to another in a heterogeneous network to perform a task. In other words, the program running at a host can suspend its execution at an arbitrary point, transfer itself to another host (or request the host to transfer it to its next destination) and resume execution from the point of suspension. Once an agent is launched, it can continue to function even if the user is disconnected from the network. They implement a computational metaphor that is analogous to how most people conduct business in their daily lives: visit a place, use a service, and then move on. . When an agent reaches a server, it is delivered to an agent execution environment. Then, if it possesses necessary authentication credentials, its executable parts are started. To accomplish its task, the mobile agent can transport itself to another server in search of the needed resource/service, spawn new agents, or interact with other stationary agents. Upon completion, the mobile agent delivers the results to the sending client or to another server.

Various attributes of a MA are: 1) *Identification* to identify a MA and its dispatching station, 2) *Itinerary* consisting of the number and order of the hosts to be visited by MA, 3) *Data unit* containing the required data, 4) *Code unit* containing the

transportable code, 5) Execution state, i.e., output of the serialization process [19], and 6) *External state* for intermediate results.

The MA paradigm provides the following key advantages:

- MAs reduce the network traffic and save the network bandwidth by allowing only shared resultant data to be carried over the network.
- Some of the processor and memory intensive tasks of a program can be offloaded to other more powerful and not heavily loaded platforms due to agent's mobility [21].
- Protocol in a distributed system can be encapsulated within a MA and any up gradation can be easily delivered and deployed to all locations using MA.
- Mobile agents operate asynchronously. Once a mobile agent is dispatched from the home machine, the home machine can disconnect from the network. The mobile agent executes autonomously without the intervention of the home machine. The home machine can reconnect at a later time and collect the agent.
- Mobile agents react dynamically and autonomously to the changes in their environment, which makes them robust, task selector and fault tolerant. A properly implemented MA may have the intelligence to bypass the faulty hosts or seek temporary shelter on a reliable host.
- MAs enable the use of portable, low-cost, personal communications devices such as PDAs, laptops etc. to perform complex tasks even in a suddenly disconnected network. The agent acts as a network application's surrogate.
- Mobile agents can learn from experiences and adapt themselves to the environment. They can monitor traffic in large networks and learn about the trouble spots in the network. Based on the experiences of the agent in the network the agent can choose better routes to reach the next host.
- MA system allows an application to scale well, since the number of participating hosts can be increased without any significant impact on the complexity of the application. The parent agent can also clone several child agents to implement concurrent operations, and raise running efficiency.
- MAs are naturally heterogeneous and goal oriented in nature. They cooperate, negotiate and collaborate with other agents to make a smart system.

Agent Driven Data Mining (ADD) a.k.a Multi-Agent Data Mining (MADM) seeks to harness the general advantages of Multi Agent System (MAS) to the application domain of DM like extendibility of the DM framework, sharing of the resources and experience, enhanced end user accessibility, information hiding and addressing of the privacy and security issues, i.e., preservation of intellectual property rights [30]. Since MAS are also distributed systems, combining DDM with MAS for data intensive applications is appealing [38]. A number of DDM solutions[23,24,25,26,27,28,29,37] are provided in recent years using various techniques such as distributed association rules, distributed clustering, Bayesian learning, classification (regression), and compression, but only a few of them make use of intelligent agents [22].

5 Agent Based Frameworks for ARM

This section is the core work of this paper reviewing the state of the art in agent based DARM. Only limited literature is available regarding the architecture or framework for this domain. Five important papers highlighting this domain are reviewed below.

Kamal Ali Albashiri et. al. [30,31,32,33,34] introduced an extendable agent enriched data mining(AEDM) system implemented in Java Agent Development Environment(JADE) called Extendible Multi-Agent Data mining System (EMADS). The system's operation is described and illustrated in terms of two Knowledge Discovery in Data (KDD) scenarios: meta association rule mining (Meta ARM) and classifier generation. Meta Mining is defined as the process of combining the individually obtained results of N applications of a data mining activity. Each local frequent itemsets T-tree is collected and a merged T-tree is generated for global frequent itemsets. EMADS provides a sound foundation for both KDD research and application based AEDM. The central feature of the framework is that, for addressing and communication mechanisms between agents, it avoids the use of agreed meta-language formats by supporting a system of mediators and wrappers coupled with Agent Communication Language (ACL) such as Foundation for Intelligent Physical Agents (FIPA) ACL. The broad advantages offered by the framework are:-

- The system is easily extendible, so that further data agents and mining agents can simply be added to the system.
- It can enable and accelerate the deployment of practical solutions to data mining problems.
- It offers decentralized control, distribution of computational resources, and interoperability.
- Flexibility in assembling the communities of autonomous service providers, including the incorporation of exiting applications.
- Minimization of the efforts to create new agents, and to wrap existing applications.
- Great support for end user accessibility to express DM request without having detailed knowledge of the individual agents.
- Preservation of intellectual property rights by providing privacy and security.
- System operation methodology includes meta-ARM and classifier generation.
- Communication and cooperation between agents are brokered by facilitators.
- Framework avoids the use of agreed meta language format for addressing and communication by supporting a system of mediators and wrappers coupled with ACL.
- Implementation in Java Agent Development Environment (JADE) framework.

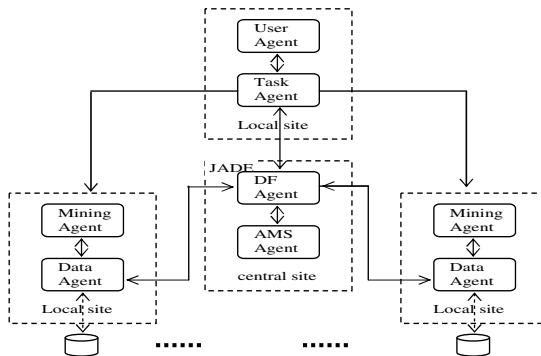


Fig. 1. Meta ARM Model Architecture [31]

Figure 1 shows the Meta ARM model architecture of EMADS framework. The system consists of the following components:-

1. *Central site*: There is one organization site (mediator host) having the following sub components-:
 - a. *AMS (Agent Management System) Agent*: It is a special JADE agent that is automatically started when the organization site is launched and provides the Naming Service (i.e. ensures that each agent in the platform has a unique name) and represents the authority in the platform. It acts as mediator to manage all routine communication and registration.
 - b. *DF (Directory Facilitator) Agent*: It is a special JADE agent that is automatically started when the organization site is launched and provides a Yellow Pages service by means of which an agent can find other agents providing the services required in order to achieve its goals. It also acts as mediator to manage all routine communication and registration.
2. *Local sites*: There are several local sites (sites of individual hosts) having at least one agent that is a member of the organization. the other sub components are-:
 - a. *DBMS (Data Base Management System)* to store the detailed data of local sites.
 - b. *Data and Mining Agents*: These are responsible for data accessing and carrying through the data mining process; these agents work in parallel and share information through the task agent. Data mining is carried out by means of local data mining agents (for reasons of privacy preservation). The basic function of these agents is to generate local item sets (local model) from local data and provide this to the task agent in order to generate the complete global set of frequent itemsets (global model)
 - c. *Task agent*: It co-ordinates the data mining operations, and presents results to the user agent.
 - d. *User agent*: It communicates with the task agent to assign the task and get the results back.

Gongzhu Hu and Shaozhen Ding, [35,36] proposed an agent based approach to mine the association rules from the distributed data sets across the multiple sites while preserving the privacy of the local data sets. This approach relies on the local systems to

find the frequent itemsets that are encrypted and the partial results are carried from site to site. Various types of mobile agents with specific functionalities and communication schemes are employed in this structural model to accomplish the task. The system consists of the following components:-

1. *Agent Master (Server)*: There is one agent server that communicates with local hosts through six types of agents created and dispatched by it. The agents are defined as:-
 - a. *Encrypt Secure Union Agent (ESUA)*: This agent tries to coordinate the n hosts to participate encryption of each locally frequent k-itemset at host i ($F_{i(k)}^l$).
 - b. *Decrypt Secure Union Agent (DSUA)*: The DSUA travels through every host to pursue decryption.
 - c. *Encrypt Sum Agent (ESA)*: ESA carries Rule Set containing pairs of item_label, count. It travels through all the hosts to obtain the encrypted support count.
 - d. *Decrypt Sum Agent (DSA)*: DSA carries the array of RuleSet extracted from the returned ESA. It travels through all the hosts and let each host subtract the random number they generate when dealing with the ESA.
 - e. *Broadcast Agent (BA)*: When the DSA agent comes back to agent master the globally frequent k-itemsets (F_k) can be calculated from the decrypted RuleSet then BA is used to carry F_k to each host to update their knowledge.
 - f. *Over Agent (OA)*: It is used to notify all the hosts that algorithm has terminated.
2. *Local Hosts*: There are several local hosts where agents can visit and perform their tasks of local association rules mining, encryption and decryption etc.

The broad advantages offered by the framework are:-

- It preserves the privacy of the local data sets by encrypting the frequent itemsets by means of secure union and secure sum.
- System operation methodology includes ARM.
- The system is easily extendible, so that further agents can simply be added to the system.

Cristian Aflori and Florin Leon [39] discovered ARs in a distributed database using intelligent agents and applying loose-couple incremental approach for effective distributed association rules mining.

Yun-Lan Wang et. al. [43] proposed mobile-agent-based distributed knowledge discovery architecture called, knowledge discovery management system (KDMS) for data mining in the distributed, heterogeneous database systems. The architecture contains some knowledge discovery sub-systems (sub-KDS). KDMS is located in the management system of the distributed database system. The sub-KDS is located in each site of the distributed database. Based on the architecture of the distributed knowledge discovery system a flexible and efficient mobile-agent-based distributed algorithm (IDMA) for association rules is presented, in which the global association rules and all the local association can be mined at the same time. The thinking in IDMA is that one mobile agent is dispatched by the KDMS to each sub-KDS to

mining all the local large itemsets. According to the relations between the local and the global large itemsets, most global large itemsets can be ascertained. If some itemsets cannot be decided they are global large or not, the counter mobile agents can be dispatched to relative sites to get the support count, so all the global frequent itemsets can be mined.

U. P. Kulkarni et. al. [41,42] suggested the improvements over the method proposed by You-Lin Ruan et al [40] and uses the advantages of MAs over client/server based approaches for better bandwidth usage and network latency. The methodology adopted in [40] uses two steps.

1. Mining Local Frequent Item sets (LFI) at each site in parallel and send them to central site to calculate Global Frequent Item sets(GFI).
2. Central site calculates the Candidate Global Frequent Item sets-CGFI and send them to all sites. Each local site computes the count of such CGFI and sends them back to central site to complete the process of finding GFI.

The basic objective of the algorithm proposed by U.P.Kulkarni et. al.[40,41] is to reduce the time required to compute GFI. The proposed algorithm performs two tasks parallel.

1. Local sites send LFIs to central site and also to all their neighbors.
2. Calculation of GFI/CGFI at central site and counts of CGFI at local sites is done as a overlapped operation. That is, local sites need not wait for central site to send CGFI. Thus total time taken is reduced drastically.

6 Proposed Framework for DARM

This section gives an overview of the designed and implemented framework called Agent enriched Mining of Strong Association Rules (AeMSAR) from Distributed Data Sources. Results are being verified and validated and would be the incorporated into the forthcoming paper. This multi-agent framework is designed on the top of the Platform for Mobile Agent Distribution and Execution (PMADE)[5][6][7].

Detailed architecture is shown in figure 2. This framework consists of one central site $S_{CENTRAL}$ where the global knowledge is computed and n distributed sites $\{S_i, i = 1 \dots n\}$ where horizontal partitioned transaction datasets $\{DB_i, i = 1 \dots n\}$ are stored. Synthetic Transactional Data sets are generated and stored at each distributed site using a tool called TDSGenerator[] $S_{CENTRAL}$ acts as the agent launching station from where mobile agents are dispatched carrying some information and returned back with the results. Mobile as well as Stationary agents are stored in the Agent Pool at this site. A Central Security Agency (CSA) at this site assigns a legal certificate to every mobile agent before its launch and when that agent reaches at a node in its itinerary authenticity of this certificate is verified again so that no malicious agent can attack local node. We have identified five agents in the architecture, three of these are MAs and other two are stationary intelligent agents to perform different tasks. Mobile Agents are- ***Local Frequent Itemset Generator Agent (LFIGA)***, ***Local Knowledge Generator Agent (LKGA)***, ***Total Frequent Itemset Collector Agent***

(*TFICA*). These agents maintains dynamic itinerary, whenever required this can be updated at any node at any time in the itinerary. These agents maintain two containers (Result Container and State Container) one for transporting result data across the network and other for state variables and their intermediate values. Stationary Agents are- ***Global Frequent Itemset Generator Agent (GFIGA)*** and ***Global Knowledge Generator Agent (GKGA)***.

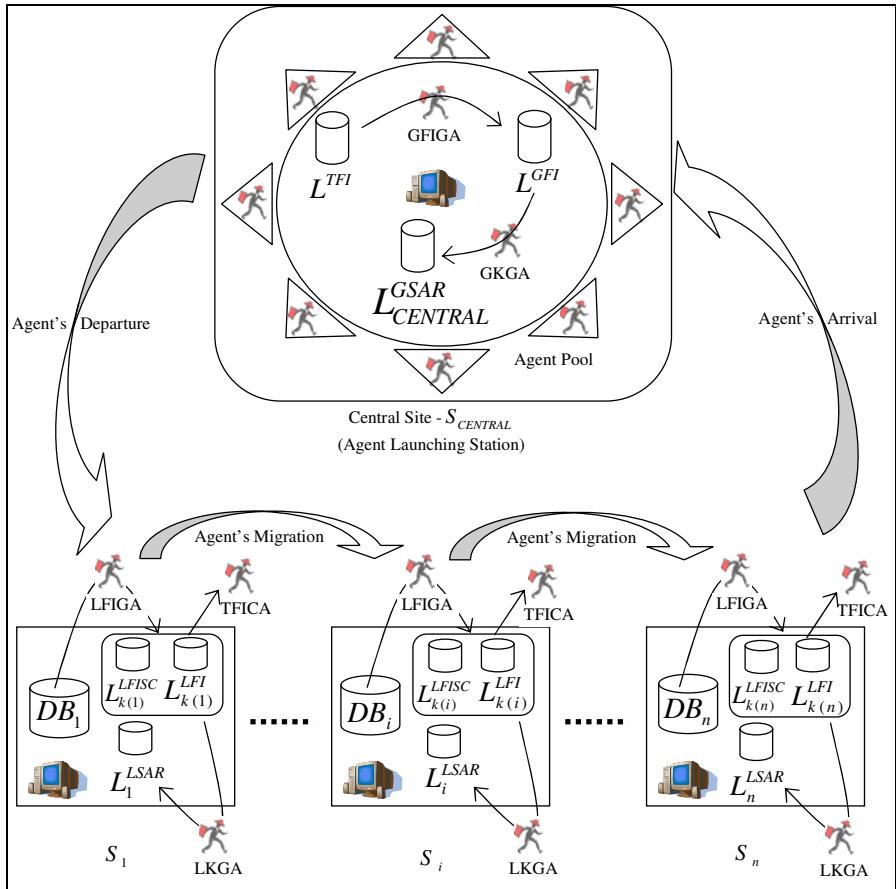


Fig. 2. Architecture of AeMSAR

Detailed relationship among these agents and working behaviour of each agent is given below:

- ***Local Frequent Itemset Generator Agent (LFIGA)*:** It is launched from the $S_{CENTRAL}$ carrying given minimum threshold support (min_sup) and visits n sites $\{S_i, i = 1 \dots n\}$ in its itinerary. It generates and stores the list of local

frequent k-itemset ($L_{k(i)}^{LFI}$) and list of support count ($L_{k(i)}^{LFISC}$) of every items in $L_{k(i)}^{LFI}$ at site S_i by applying the Apriori [1][2][3] algorithm on the local transactional dataset (DB_i) at that site with the constraint of min_sup.

- **Local Knowledge Generator Agent (LKGA):** It is also launched from the $S_{CENTRAL}$ and visits n sites $\{S_i, i = 1 \dots n\}$ in its itinerary. It applies the constraints of given minimum threshold confidence (min_conf) to generate and store the list of locally strong association rules (L_i^{LSAR}) by using $L_{k(i)}^{LFI}$ and $L_{k(i)}^{LFISC}$ generated by LFIGA at site S_i .
- **Total Frequent Itemset Collector Agent (TFICA):** It is also launched from the $S_{CENTRAL}$ and visits n sites $\{S_i, i = 1 \dots n\}$ in its itinerary. While visiting each site it collects lists of local frequent k-itemset ($L_{k(i)}^{LFI}$) generated by LFIGA and carries back the list of total frequent k-itemset $L_k^{TFI} = \bigcup_{i=1}^n L_{k(i)}^{LFI}$ at $S_{CENTRAL}$. $L_{k(i)}^{LFI}$ can be encrypted so that privacy of the local data can be preserved.
- **Global Frequent Itemset Generator Agent (GFIGA):** It is a stationary agent at $S_{CENTRAL}$ mainly used for processing the total frequent k-itemset list L_k^{TFI} generated by TFICA to generate the global frequent itemset list $L_k^{GFI} = \bigcap_{i=1}^n L_{k(i)}^{LFI}$ which is the intersection of all the local frequent k-itemset ($L_{k(i)}^{LFI}$).
- **Global Knowledge Generator Agent (GKGA):** It is also a stationary agent at $S_{CENTRAL}$ mainly used for processing the global frequent itemset list L_k^{GFI} generated by GFIGA to compile the global knowledge, i.e., the list of globally strong association rules $L_{CENTRAL}^{GSAR}$.

7 Conclusion

This paper makes an effort to review the role of Mobile Agents in Distributed Association Rules Mining. An inside view of the existing frameworks in this domain presented and a newly designed and implemented framework called Agent enriched Mining of Strong Association Rules (AeMSAR) from Distributed Data Sources is highlighted. Results are being verified and validated and would be incorporated in forthcoming paper.

References

1. Fayyad, U.M., Piatetsky-Shapiro, G., Smyth, P., Uthurusamy, R.: *Advances in Knowledge Discovery and Data Mining*. AAAI/MIT Press (1996)
2. Han, J., Kamber, M.: *Data Mining: Concepts and Techniques*, 2nd edn. Morgan Kaufmann (2006)
3. Park, B.-H., Kargupta, H.: *Distributed Data Mining: Algorithms, Systems, and Applications*, Department of Computer Science and Electrical Engineering, University of Maryland Baltimore County, 1000 Hilltop Circle Baltimore, MD 21250
4. Tsoumakas, G., Vlahavas, I.: *Distributed Data Mining*, Department of Informatics. Aristotle University of Thessaloniki, Thessaloniki
5. Kulkarni, U.P., Desai, P.D., Ahmed, T., Vadavi, J.V., Yardi, A.R.: Mobile Agent Based Distributed Data Mining. In: Proceedings of International Conference on Computational Intelligence and Multimedia Applications. IEEE Computer Society (2007)
6. Kargupta, H., Chan, P. (eds.): *Advances in Distributed and Parallel Knowledge Discovery*. AAAI/MIT Press (2000)
7. Fu, Y.: *Distributed Data Mining: An Overview*. In: Newsletter of the IEEE Technical Committee on Distributed Processing, pp. 5–9 (Spring 2001)
8. Otey, M.E., Parthasarathy, S., Chao, W., Adriano, V., Meira Jr., W.: Parallel and Distributed Methods for Incremental Frequent Itemset Mining. *IEEE Transactions on Systems, Man, and Cybernetics- Part B: Cybernetics* 34(6) (December 2004)
9. Han, J., Pei, J., Yin, Y.: Mining Frequent Patterns without candidate generation. In: Proc. ACM-SIGMOD, Dallas, TX (May 2000)
10. Agrawal, R., Imielinski, T., Swami, A.: Mining association rules between sets of items in large databases. In: Proc. 1993 ACM-SIGMOD Int. Conf. Management of Data, WA, pp. 207–216 (May 1993)
11. Agrawal, R., Srikant, R.: Fast algorithms for mining association rules. In: Proc. 1994 Int. Conf. Very Large Data Bases, Santiago, Chile, pp. 487–499 (September 1994)
12. Raun, Y.-L., Liu, G., Li, Q.-H.: Parallel Algorithm for Mining Frequent Itemsets. In: Proc. of the Fourth International Conference on Machine Learning and Cybernetics, Guangzhou, August 18-21 (2005)
13. Zaki, M.J.: *Parallel and Distributed Association Mining: A Survey*, Department of Computer Science. Rensselaer Polytechnic Institute, Troy
14. Picco, G.P.: Mobile Agents: An Introduction. *Microprocessors and Microsystems* 25, 65–74 (2001)
15. Klusch, M.: Information Agent technology for the internet: A survey. *Data and Knowledge Engineering, Special Issue on Intelligent Information Integration* 36, 337–372 (2001)
16. Wooldridge, M.J.: Intelligent Agents: The Key Concepts. In: Mařík, V., Štěpánková, O., Krautwurmová, H., Luck, M. (eds.) *ACAI 2001, EASSS 2001, AEMAS 2001, and HolomAS 2001. LNCS (LNAI)*, vol. 2322, pp. 3–43. Springer, Heidelberg (2002)
17. Patel, R.B., Garg, K.: PMADE – A Platform for mobile agent Distribution & Execution. In: The Proceedings of 5th World MultiConference on Systemics, Cybernetics and Informatics (SCI 2001) and 7th International Conference on Information System Analysis and Synthesis (ISAS 2001), Orlando, Florida, USA, July 22-25, vol. IV, pp. 287–293 (2001)
18. Patel, R.B., Garg, K.: A flexible security framework for mobile agents. *Intl. Journal of Control & Intelligent systems'* 33(3), 175–183 (2005)
19. SunMicrosystems, Java Object Serialization Specification (1997),
<http://java.sun.com/j2se/1.3/docs/guide/serialization/spec/serialTOC.doc.html>

20. Yang, G.-P., Zeng, G.-Z.: Mobile Agent Life State Management. In: IMACS Multi-Conference on Computational Engineering in Systems Applications (CESA), Beijing, China, October 4-6 (2006)
21. Verspecht, D.: Thesis, Mobile agents for mobile platforms. Vrije Universiteit Brussel, Faculty of Science, Department of computer science (2001-2002)
22. Klusch, M., Lodi, S., Moro, G.: The Role of Agents in Distributed Data Mining: Issues and Benefits. In: Proceedings of the IEEE/WIC International Conference on Intelligent Agent Technology (IAT 2003). IEEE Computer Society (2003)
23. Stolfo, S., Prodromidis, A.L., Tselepis, S., Lee, W., Fan, D.W., Chan, P.K.: JAM: Java Agents for Meta-Learning over Distributed Databases. In: Proc. 3rd Int'l Conf. Knowledge Discovery and Data Mining (KDD 1997), pp. 74–81. AAAI Press (1997)
24. Chatratchat, J., Darlington, J., Guo, Y., Hedvall, S., Kohler, M., Syed, J.: An Architecture for Distributed Enterprise Data Mining. In: Sloot, P.M.A., Hoekstra, A.G., Bubak, M., Hertzberger, B. (eds.) HPCN-Europe 1999. LNCS, vol. 1593, pp. 573–582. Springer, Heidelberg (1999)
25. Parthasarthy, S., Subramonium, R.: Facilitating Data Mining on a Network of Workstations. In: Kargupta, H., Chan, P. (eds.) Advances in Distributed Data Mining, pp. 229–254. AAAI Press (2001)
26. Kargupta, H., Hamzaoglu, L., Stafford, B., Hanagandi, V., Buescher, K.: PADMA: Parallel data mining agent for scalable text classification. In: Proceedings of Conference on High Performance Computing 1997. pp. 290–295. The Society for Computer Simulation International (1996)
27. Kargupta, H., Park, B., Hershberger, D., Johnson, E.: Collective Data Mining: A new perspective toward Distributed Data Mining. In: Advances in Distributed and Parallel Knowledge Discovery, pp. 131–178. AAAI/MIT Press (2000)
28. Martin, G., Unruh, A., Urban, S.: InfoSleuth: An agent infrastructure for knowledge discovery and event detection (Tech. Rep. No. MCC-INSL-003-99). Microelectronics and Computer Technology Corporation (MCC)
29. Honaver, V., Miller, L., Wong, J.: Distributed Knowledge Networks. In: IEEE Information Technology Conference, Syracuse, NY (1998)
30. Albashiri, K.A., Coenen, F., Sanderson, R., Leng, P.H.: Frequent Set Meta Mining: Towards Multi-Agent Data Mining. In: SGAI Conf., pp. 139–151 (2007)
31. Albashiri, K.A., Coenen, F., Leng, P.: Agent Based Frequent Set Meta Mining: Introducing EMADS. In: Artificial Intelligence in Theory and Practice II, vol. 276, pp. 23–32. Springer, Boston (2008)
32. Albashiri, K.A., Coenen, F., Leng, P.H.: EMADS: An extendible multi-agent data miner. Knowledge Based System 22(7), 523–528 (2009)
33. Albashiri, K.A., Coenen, F.: A Generic and Extendible Multi-Agent Data Mining Framework. In: Corchado, E., Wu, X., Oja, E., Herrero, Á., Baruque, B. (eds.) HAIS 2009. LNCS, vol. 5572, pp. 203–210. Springer, Heidelberg (2009)
34. Albashiri, K.A., Coenen, F.: Agent-Enriched Data Mining Using an Extendable Framework. In: Cao, L., Gorodetsky, V., Liu, J., Weiss, G., Yu, P.S. (eds.) ADMI 2009. LNCS, vol. 5680, pp. 53–68. Springer, Heidelberg (2009)
35. Hu, G., Ding, S.: An Agent-Based Framework for Association Rules Mining of Distributed Data. In: SERA 2009, pp. 13–26 (2009)
36. Hu, G., Ding, S.: Mining of Association Rules from Distributed Data using Mobile Agents. In: ICE-B 2009, pp. 21–26 (2009)

37. Baik, S.W., Bala, J., Cho, J.S.: Agent Based Distributed Data Mining. In: Liew, K.-M., Shen, H., See, S., Cai, W. (eds.) PDCAT 2004. LNCS, vol. 3320, pp. 42–45. Springer, Heidelberg (2004)
38. Giannella, C., Bhargava, R., Kargupta, H.: Multi-agent Systems and Distributed Data Mining. In: Klusch, M., Ossowski, S., Kashyap, V., Unland, R. (eds.) CIA 2004. LNCS (LNAI), vol. 3191, pp. 1–15. Springer, Heidelberg (2004)
39. Aflori, C., Leon, F.: Efficient Distributed Data Mining using Intelligent Agents. In: Proceedings of the 8th International Symposium on Automatic Control and Computer Science, Iasi (2004) ISBN 973-621-086-3
40. Ruan, Y.-L., Liu, G., Li, Q.-H.: Parallel Algorithm for Mining Frequent Item sets. In: Proceeding of the Fourth International Conference on Machine Learning and Cybernetics, August 18-21, pp. 2118–2121. IEEE (2005)
41. Kulkarni, U.P., Tangod, K.K., Mangalwede, S.R., Yardi, A.R.: Exploring the capabilities of Mobile Agents in Distributed Data Mining. In: Proceeding of the 10th International Database Engineering & Applications Symposium- IDEAS 2006. IEEE Computer Society (2006)
42. Kulkarni, U.P., Desai, P.D., Ahmed, T., Vadavi, J.V., Yardi, A.R.: Mobile Agent Based Distributed Data Mining. In: Proceedings of International Conference on Computational Intelligence and Multimedia Applications. IEEE Computer Society (2007)
43. Wang, Y.-L., Li, Z.-Z., Zhu, H.-P.: Mobile-Agent based Distributed and Incremental Techniques for Association Rules. In: Proceedings of the Second International Conference on Machine Learning and Cybernetics, Xi'an, China, November 2-5, pp. 266–271 (2003)
44. Bhamra, G.S., Patel, R.B., Verma, A.K.: TDSGenerator: A Tool for generating synthetic Transactional Datasets for Association Rules Mining. International Journal of Computer Science Issues 8(2), 184–188 (2011) ISSN: 1694-0814
45. Cao, L., Gorodetsky, V., Mitkas, P.: Agent Mining: The Synergy of Agents and Data Mining. IEEE Intelligent Systems 24(3), 64–72 (2009)

Obtaining an Optimal MAS Configuration for Agent-Enhanced Mining Using Constraint Optimization

Chayapol Moemeng, Can Wang, and Longbing Cao

Quantum Computing and Intelligent Systems,
Faculty of Engineering and Information Technology,
University of Technology, Sydney
P.O. Box 123, Broadway, NSW 2007, Australia
{mchayapol,cawang,lbciao}@it.uts.edu.au

Abstract. We investigate an interaction mechanism between agents and data mining, and focus on agent-enhanced mining. Existing data mining tools use workflow to capture user requirements. The workflow enactment can be improved with a suitable underlying execution layer, which is a Multi-Agent System (MAS). From this perspective, we propose a strategy to obtain an optimal MAS configuration from a given workflow when resource access restrictions and communication cost constraints are concerned, which is essentially a constraint optimization problem. In this paper, we show how workflow is modeled in the way that can be optimized, and how the optimized model is used to obtain an optimal MAS configuration. Finally, we demonstrate that our strategy can improve the load balancing and reduce the communication cost during the workflow enactment.

Keywords: Constraint Optimization, Workflow Management System, Agent and Data Mining Interaction.

1 Introduction

A challenge in agent and data mining interaction (ADMI) is the interaction itself: whether agent enhances the data mining process, or data mining is used to improve agent intelligence [3]. Given the current pace of research progression from these two fields, introducing an entirely new mechanism would face the issues in terms of acceptance and adoption by the ADMI community.

At first, let us look at the successful existing data analysis and mining tools (data mining tools, for short). Take SAS Analytics¹, RapidMiner², KNIME³, as examples, they manage complicated data analysis and mining components

¹ SAS website: <http://www.sas.com/>

² Rapid-i website: <http://rapid-i.com/>

³ KNIME website: <http://www.knime.org>

through workflow-style tools. Workflow not only provides a high level of visualization which increases the usability of the system, but also eases system developments in terms of re-usability from the computational resources, such as data analysis and mining components. As a result, users can define arbitrary workflows for their requirements with the support of these application tools. As for today, Workflow Management System (WfMS) has become a de facto standard for data mining tools [9]. Contemporarily, major efforts on improving WfMS performance by using different system architectures and engineering strategies [1,14] have been made.

Besides, in terms of data mining techniques, there exist two major constraints which significantly impact the performance of the data mining process: *large data sets* and *resource access restriction* [8]. In fact, transmitting a large amount of data over the network could degrade such performance due to the enforced resource access restriction, and may lead the system into inconsistency state and various costs [1] which involve message passing for workflow management activities. Consequently, handling the aforementioned constraints together with the performance improvement at the same time is a challenging task.

However, from the standpoint of our interests in ADMI, adjustable anatomy of Agent-based Workflow Management System (A-WfMS) has been investigated to support dynamic constraints analysis recently [2]. Within A-WfMS, essentially, the workflow functions as a process descriptor and a Multi-Agent System (MAS) handles the workflow engine by coordinating the workflow enactment [7,11,5,10].

Our recent work [9] has shown the successful use of workflow system as an interaction platform between agent and data mining. Further, in this paper, we focus on how to obtain the optimal MAS configuration for the agent-enhanced data mining when given an optimized workflow model. In fact, this problem is a class of Constraints Optimization Problem (COP) in which the hard constraint (resource access restriction) must be satisfied and the soft constraint (data transfer minimization) can be optimized.

The main contributions of this paper are in threefold: (i) we explore a type of A-WfMS data mining process explicitly, (ii) we present a strategy to interact between agents and data mining in terms of agent-enhanced mining based on constraint optimization framework, and (iii) we provide the ADMI community with a mechanism that can be quickly and inexpensively accepted and adopted.

The rest of the paper is structured as follows. Section 2 reviews the related work for this paper. Problem statement, including preliminary assumptions, case description, and relevant definitions, is explained in Section 3 in which the POCL plan [13] is used to present our task description. Section 4 specifies the constraint optimization and the MAS configuration schemes individually. Afterwards, we evaluate the strategy and analyse the results in Section 5. Finally, we end the paper in Section 6.

2 Related Work

Integration of agents and WfMS have been introduced in the late 1990s. Pioneer work [7,11] argued that agents are suitable for workflows since the nature of

the requirements could always evolve over time. Therefore, automatic process improvement is desirable, which can then intelligently adapt to the changing environmental conditions. Recent A-WfMSs, JBees [5] and i-Analyst [10], use collaborating software agents as the basis of the systems provides more flexibility than existing WfMSs. JBees uses Coloured Petri nets as process formalism. However, JBees' allocation of process agent (workflow engine) does not concern the cost of communication and resource access restrictions. On the other hand, i-Analyst is specifically designed for data mining process using agent technology for workflow execution layer. It provides a range of capabilities, including workflow construction, workflow management, algorithm development, resource management, workflow enactment, and reporting. The mechanism for MAS configuration of i-Analyst is described in this paper.

In later years, attentions have been moved to the implementation techniques. The increasing growth of the Internet has played a major role in the development of WfMS. Recently and remarkably, web Services is a promising method to provide the computational resources for workflow, and such workflow benefits from the support for distributed process models [12]. Although web services technology is becoming a main stream in workflow integration, Huhns [6] argued that web services alone may not completely fulfil distributed WfMS requirements due to the fact that web services know only about themselves, rather than any meta-level awareness; web services are not designed to utilize or understand ontologies; besides, web services are not capable of autonomous action, intentional communication, or deliberatively cooperative behavior. Accordingly, Buhler and Buhler [2] showed a critical survey of workflow, web services, and agent technologies to make a point that web services and MAS will become imperative parts of the future WfMS development. To our extend, we aim at an agent-enhanced data mining system in which computational resources (operations) are not required to be web services. The location requirement of the web service does not support our approach, because rather than transmitting large data over the network to the designated operation at a service provider, we prefer the operation to be transmitted to the data site instead.

Some works attempting to improve the performance of A-WfMS have become incrementally popular. For instance, Yoo et al. [14] proposed an architecture that involves agents in workflow engine layer, therefore, agents can help distribute workloads of a naming/location server and a workflow engine effectively. Bauer and Dadam [1] proposed a variable server assignment for distributed WfMS which allows dynamic server assignment without expensive run-time analysis. Their approaches use the advanced techniques in system engineering to help minimize the communication load in the distributed system, while our approach takes great advantage of the pre-construct configuration which has its own unique benefit: that is once the agent organization has been acquired and tasks have been allocated to the corresponding agents, they may perform tasks in advance, such as loading independent data and libraries, without having to wait for dependent task completions. In our understanding, improving the performance means minimizing the related costs in A-WfMS.

3 Problem Statement

This section depicts the building blocks of this paper. We propose, firstly, the preconditions of the involved data mining tool. Secondly, the case to be used in the rest of the paper is specified. And finally, the relevant definitions are formalized. In this work, the concerned system is a data mining tool with an embedded WfMS. The WfMS has the information of the size of data sources. The workflow execution layer is deployed on a group of computers (hosts) on a network. Each host has an agent platform to perform as a workflow engine. Lastly, the topology of the network does not change significantly. We have implemented such system in our previous work [10].

3.1 Case Description

For the rest of the paper, we set up a workflow and describe the MAS configuration to be obtained from the above one.

To begin with, Fig. 1 is a workflow that consists of six tasks. All the tasks are connected with each other by one or more directed arcs. Each arc is inscribed with a cause c_i and a cost in a numeric value. A cause c is both an effect of task t_i and a precondition of task t_j [13] which can be written as $t_i \xrightarrow{c} t_j$.

Highlighted tasks t_1 and t_4 are *location-specific tasks*, which must be executed at the specific location only; while other tasks are *non-location-specific*.

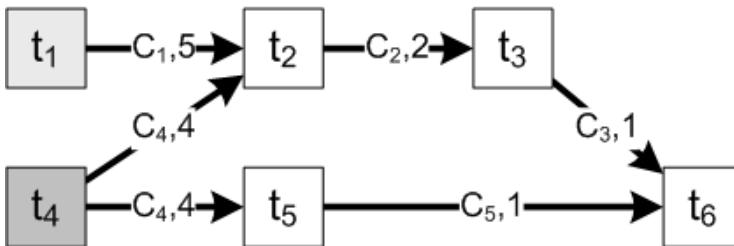


Fig. 1. A Workflow

In addition, MAS configuration is a document that describes the organization of agents in a platform. The description of the configuration is varied by the types of agent platform. Let us use a generic agent platform, WADE, for our common understanding. WADE is an extension of JADE (Java Agent Development Framework)⁴. Listing 1.1 shows a sample of MAS configuration used in WADE. Specifically, an agent *platform* is composed of multiple hosts. Each *host* may contain multiple agent containers, in which each *container* may have multiple agents. *Agents*, in particular, are type specific. In this example, agents are of *Workflow Engine (WE) Agent* type. Note that every element (host, container, and agent) must have a specific name.

⁴ WADE website, <http://jade.tilab.com/wade/index.html>

```

<platform name="Configuration-1">
  <hosts>
    <host name="host1">
      <containers>
        <container name="Execution-Node-1">
          <agents>
            <agent name="performer1" type="WE-Agent" />
            <agent name="performer2" type="WE-Agent" />
          </agents>
        </container>
      </containers>
    </host>
  </hosts>
</platform>

```

Listing 1.1: Sample MAS Configuration

3.2 Definitions

We desire to obtain the MAS configuration that is optimal for a workflow. In order to get such document, we establish the following definitions. In this work, our definition of a multi-agent plan extends that of the Partial-Order Causal-Link (POCL) plan. The POCL plan definition has been well-established in the planning community [13]. Our definition is also inspired by the framework proposed by Cox and Durfee [4].

Definition 1. A *Localized POCL (L-POCL) plan* is a tuple $P = \langle T, \succ_T, \succ_C, L_T, \mathcal{C} \rangle$ where

- T is a set of the tasks,
- \succ_T is the temporal orders on T , where $e \in \succ_T$ is a tuple $\langle t_i, t_j \rangle$ with $t_i, t_j \in T$,
- \succ_C is the causal partial orders on T , where $e \in \succ_C$ is a tuple $\langle t_i, t_j, c \rangle$ with $t_i, t_j \in T$ and c is a cause,
- L_T is a set of the locations and the corresponding tasks on T , where $e \in L_T$ is a tuple $\langle l_i, t_j \rangle$ with location $l_i \in L$ and task $t_j \in T$,
- \mathcal{C} is the cost function mapping $\mathcal{C} : T \times T \rightarrow \{0, \mathfrak{R}\}$, this function $\mathcal{C}(t_i, t_j) = 0$ if $l_{t_i} = l_{t_j}$, otherwise $\mathcal{C}(t_i, t_j) = r$ where $t_i, t_j \in T$, and $r \in \mathfrak{R}$ is a real value related with the size of data to be transmitted.

Location and Cost Function. We have added location set L_T and cost function \mathcal{C} to the original POCL plan. The L-POCL plan itself does not fully represent the workflow, since it does not maintain the information about process and execution conditions. However, it is still adequate to capture necessary information for constraint optimization. Here, location can be in any form that uniquely represents a physical place, such as IP address, host name, etc. A non-location-specific task has \emptyset as its location. In relation to L_T , we introduce two more related sets $L_{|T|}$ and T_L^* .

- $L_{|T|}$: describes the number of tasks at the same location, i.e., $e \in L_{|T|}$ is a tuple $\langle l_i, n \rangle$, where l_i is the location, $t_i \in T$, and n is the number of tasks at location l_i .
- T_L^* : is a set of tasks $t \in T$ whose locations cannot be changed.

In terms of COP, the *hard constraint* must be satisfied strictly; in this scenario, that is the location of each task $t \in T_L^*$ cannot be changed. However, the *soft constraint* is preferred to be optimized. In this case, the concerned global function is the communication cost \mathcal{C} between each task, for the reason that the data transmission between locations incurs communication costs. But when the tasks are at the same location, communication cost is marked to zero.

Consider the workflow as shown in Fig. 1, the L-POCL plan concepts involved are exemplified as follows:

$$\begin{aligned}
T &= \{t_1, t_2, t_3, t_4, t_5, t_6\} \\
\succ_T &= \{\langle t_1, t_2 \rangle, \langle t_1, t_3 \rangle, \langle t_1, t_6 \rangle, \langle t_2, t_3 \rangle, \langle t_2, t_6 \rangle, \langle t_3, t_6 \rangle, \\
&\quad \langle t_4, t_2 \rangle, \langle t_4, t_3 \rangle, \langle t_4, t_5 \rangle, \langle t_4, t_6 \rangle, \langle t_5, t_6 \rangle\} \\
\succ_C &= \{\langle t_1, t_2, c_1 \rangle, \langle t_2, t_3, c_2 \rangle, \langle t_3, t_6, c_3 \rangle, \langle t_4, t_2, c_4 \rangle, \langle t_4, t_5, c_4 \rangle, \langle t_5, t_6, c_5 \rangle\} \\
T_L^* &= \{t_1, t_4\} \\
L_T &= \{(l_1, t_1), (l_2, t_4), (\emptyset, t_2), (\emptyset, t_3), (\emptyset, t_5), (\emptyset, t_6)\} \\
L_{|T|} &= \{(l_1, 1), (l_2, 1), (\emptyset, 4)\} \\
\mathcal{C}(i, j) &= \begin{cases} 5 & \text{if } i = t_1 \text{ and } j = t_2, \\ 2 & \text{if } i = t_2 \text{ and } j = t_3, \\ 1 & \text{if } i = t_3 \text{ and } j = t_4, \\ 4 & \text{if } i = t_4 \text{ and } j = t_2, \\ 4 & \text{if } i = t_4 \text{ and } j = t_5, \\ 1 & \text{if } i = t_5 \text{ and } j = t_6 \end{cases}
\end{aligned}$$

Note that L_T and \mathcal{C} are also illustrated in Fig 2a Initial L_T table.

Multi-Agent L-POCL. We have shown that a workflow can be modeled by a L-POCL plan. However, the MAS configuration also requires information about agents and their task assignments. The task assignment is to be mapped with the corresponding location. Therefore, a multi-agent version of L-POCL is defined as follows:

Definition 2. A *Multi-Agent L-POCL plan (ML-POCL)* is a tuple $P = \langle A, T, \succ_T, \succ_C, L_T, \mathcal{C}, X \rangle$ where

- $\langle T, \succ_T, \succ_C, L_T, \mathcal{C} \rangle$ is the embedded POCL plan,
- A is a set of the agents,
- Execution assignment X is a set of the tuples with the form $\langle t, a \rangle$, representing that the agent $a \in A$ is assigned to execute task $t \in T$.

L_T and X are necessary components to construct a MAS configuration. Consider Listing 1.1, **hosts** are all locations found in L_T ; **agents** are specified in X . L_T and X are mutually linked by tasks. Note that the naming scheme of an agent has no restriction as long as the name is unique.

4 Optimal MAS Configuration

This section describes how to obtain an optimal MAS configuration in terms of constraint optimization. A constraint optimization algorithm, in this case, is to minimize the soft constraint \mathcal{C} . The optimization reflects the change of L_T to an optimal one L_T^* . Then L_T^* is used to guide the construction of the optimal MAS configuration.

4.1 Constraint Optimization

The optimization algorithm migrates workflow tasks based on location. It uses the global cost function \mathcal{C} as a guide by reducing the costly routes by minimizing the number of locations in the workflow.

Algorithm 1. Optimize \mathcal{C}

```

input :  $L_T$ 
output:  $L_T^*$ 
 $L_T^* = L_T;$ 
while there exists non-location-specific task in  $L_T^*$  do
    foreach non-location-specific task  $t \in L_T^*$  do
        let  $T_x$  be a set of location-specific tasks adjacent to  $t$ ;
        if  $T_x = \emptyset$  then continue;
        let  $t_x \in T_x$  be a task that maximizes  $\mathcal{C}(t, t_x)$  or  $\mathcal{C}(t_x, t)$ ;;
        update  $(\emptyset, t) \in L_T^*$  to  $(l_x, t)$  where  $l_x$  is the location of  $t_x$ ;;
        if  $t_x$  maximizes  $\mathcal{C}(t, t_x)$  then
             $\mathcal{C}(t, t_x) \leftarrow 0$ ;
        else
             $\mathcal{C}(t_x, t) \leftarrow 0$ ;
    return  $L_T^*$ ;

```

Given a workflow as shown in Fig. 1, each arc is marked with the estimated size of data (presume in mega-bytes) to be transmitted between relevant tasks. Tasks t_1 and t_4 are at l_1 and l_2 respectively, and the locations of them cannot be changed since $t_1, t_2 \in T_L^*$. The corresponding location set L_T and cost function \mathcal{C} can be presented in the tabular form as shown in Fig. 2a. Each header cell contains the combinations of each task $t_j (j = 1, \dots, 6)$ and its location

\mathcal{C}	l_1, t_1	\emptyset, t_2	\emptyset, t_3	l_2, t_4	\emptyset, t_5	\emptyset, t_6
l_1, t_1	-	5	-	-	-	-
\emptyset, t_2	-	-	2	-	-	-
\emptyset, t_3	-	-	-	-	-	1
l_2, t_4	-	4	-	-	4	-
\emptyset, t_5	-	-	-	-	-	1
\emptyset, t_6	-	-	-	-	-	-

Fig 2a Initial L_T table

\mathcal{C}	l_1, t_1	l_1, t_2	\emptyset, t_3	l_2, t_4	\emptyset, t_5	\emptyset, t_6
l_1, t_1	-	0	-	-	-	-
l_1, t_2	-	-	0	-	-	-
\emptyset, t_3	-	-	-	-	-	1
l_2, t_4	-	4	-	-	0	-
l_2, t_5	-	-	-	-	-	1
\emptyset, t_6	-	-	-	-	-	-

Fig 2b Migrate t_2 to t_1

\mathcal{C}	l_1, t_1	l_1, t_2	l_1, t_3	l_2, t_4	l_2, t_5	\emptyset, t_6
l_1, t_1	-	0	-	-	-	-
l_1, t_2	-	-	0	-	-	-
l_1, t_3	-	-	-	-	-	1
l_2, t_4	-	4	-	-	0	-
l_2, t_5	-	-	-	-	-	1
\emptyset, t_6	-	-	-	-	-	-

Fig 2c Migrate t_5 to t_2 and t_3 to t_1

\mathcal{C}	l_1, t_1	l_1, t_2	l_1, t_3	l_2, t_4	l_2, t_5	l_2, t_6
l_1, t_1	-	0	-	-	-	-
l_1, t_2	-	-	0	-	-	-
l_1, t_3	-	-	-	-	-	1
l_2, t_4	-	4	-	-	0	-
l_2, t_5	-	-	-	-	-	0
l_2, t_6	-	-	-	-	-	-

Fig 2d Optimized L_T^* **Fig. 2.** Cost Function \mathcal{C} Optimization

$l_i (i = 1, 2)$, note that \emptyset means unspecified location. Moreover, ‘-’ denotes the situation of unavailable connection. As $\text{optimize}()$ is applied to the workflow, for instance, we get $\mathcal{C}(t_1, t_2) = 5$, where t_1 is at l_1 and t_2 is non-location-specific. The algorithm gradually evolves to the point L_T^* that every task is assigned with a optimal location. The algorithm performs the migration in two fashions: cost reduction and load balancing.

Cost Reduction Optimization. The algorithm searches for every non-location-specific task that is adjacent to at least one location-specific task. Firstly, (\emptyset, t_2) is found with two adjacent location-specific tasks t_1 and t_4 . As $\mathcal{C}(t_1, t_2) > \mathcal{C}(t_4, t_2)$, t_2 is migrated to the same location of task t_1 , i.e., l_1 . Tasks t_1 and t_2 are now at the same location l_1 , then $\mathcal{C}(t_1, t_2)$ is updated to zero. Subsequently, the algorithm searches again and finds another non-location-specific task t_3 , and then migrates it to the location of task t_2 , i.e., l_1 . The same principle goes on for non-location-specific task t_5 , it is migrated to the location of t_4 , i.e., l_2 .

Load Balancing Optimization. As the cost reduction optimization goes on, the last non-location-specific task t_6 has two choices to migrate, t_3 ’s location l_1 and t_5 ’s location l_2 , and both the values of $\mathcal{C}(t_3, t_6)$ and $\mathcal{C}(t_5, t_6)$ equal to 1. Now, let us consider the number of tasks at each location, indicated as $L_{|T|} = \{(l_1, 3), (l_2, 2), (\emptyset, 1)\}$. A simple load balancing advises to migrate task t_6 to location l_2 . However, it also depends on other factors such as computing power which we do not cover in this paper, we will consider it for our future work. The final L_T^* is shown in Fig. 2d.

4.2 Obtaining MAS Configuration

This section discusses the way to obtain the optimal MAS configuration. The basic configuration rule is to assign one task to one agent, thus $|A| = |T|$ regardless of locations. But agents are related to the structure of L_T^* which has been attained in the previous steps. The execution assignment X of ML-POCL plan is also needed to obtain the desired configuration. Instead of starting out with one agent for one task, one agent is assigned to one location, thus initially $|A| = |L|$ regardless of tasks. Tasks that can be executed without having to wait for other tasks should be allocated with an agent. At each location $l \in L_T^*$, determine the agent allocation according to the workflow constructs (illustrated in Fig. 3):

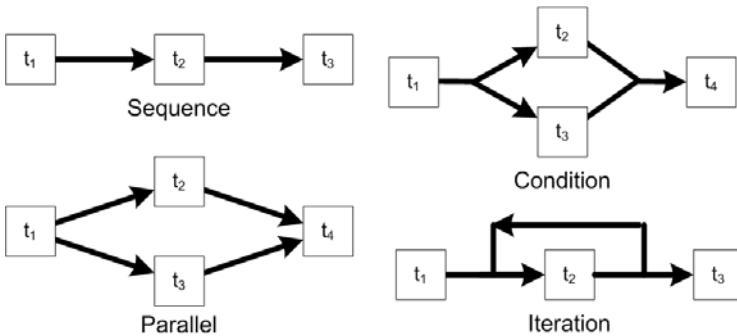


Fig. 3. Workflow Constructs

- **Sequence:** If task t_i has a successive task t_j defined in \succ_T , an agent a is assigned to t_i and t_j , s.t. $(t_i, a), (t_j, a)$.
- **Parallel:** If task t_i and t_j are not linked with temporal ordering as defined in \succ_T , two agents a_1 and a_2 are assigned to tasks t_i and t_j respectively, s.t. $(t_i, a_1), (t_j, a_2)$. For example in Fig. 1, if tasks t_2 and t_5 were at the same location, they are parallel.
- **Condition:** If task t_i is to choose either t_j or t_k exclusively, which means only one successive task will be executed, therefore an agent a is assigned to all t_i, t_j, t_k , s.t. $(t_i, a), (t_j, a), (t_k, a)$.
- **Iteration:** Iterative task is a sequential form of condition and loop. Let task t_i be the entrance to the iteration, T_i is the set of tasks to be repeated in a loop, t_j is the decision-making task, and t_k is the successive task of t_j , then an agent a is assigned to all tasks $t \in \{t_i, t_j, t_k\} \cup T_i$.

With respect to iteration, we cannot precisely calculate the number of iterations which will directly affect the total costs. But this is a factor that will influence every scheme of configuration proportionally.

5 Evaluation

In this section we demonstrate that the overall communication cost can be significantly reduced by taking advantage of the optimal MAS configuration strategy. For this purpose, we have simulated the execution of the randomly generated workflow in a distributed system composed of 5 hosts (locations). We randomly generate 20 workflows consisting of 10-20 tasks with random structures. In a workflow, 20-30% of all tasks are location specific. And each causal link is assigned with a random cost value between 1-5 (assumed mega-bytes). This simulation generates three MAS configurations for each workflow:

1. **Centralized configuration (C)**: All tasks run at one host.
2. **Equally distributed configuration (E)**: Equally assign a host with a number of tasks.
3. **Optimal configuration (O)**: The method we have proposed.

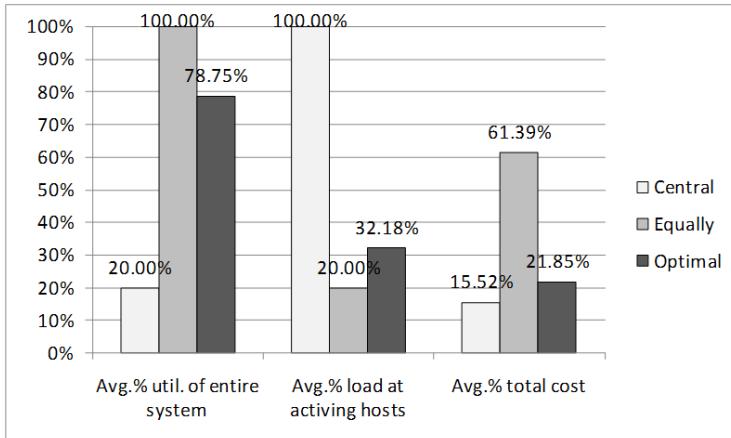


Fig. 4. Results of the Simulation

Based on the 20 workflows and 5 hosts, we calculate three evaluation values (the results are shown in Fig. 4):

1. **The average percentage of utilization of the entire system**: indicates the percentage of hosts being used to run the tasks. **C** uses only one host; **E** distributes all tasks to all hosts, so all hosts are utilized; and **O** uses only the hosts that have location-specific tasks, s.t., the number of active hosts is equal to $|L_T|$.
2. **The average percentage of load at the active hosts**: indicates the percentage of the number of tasks assigned to the active hosts. **C** runs all tasks at the only active host, s.t. 100%; **E** distributes all tasks to all hosts,

- s.t. the number of tasks per host is $|T|/|\text{host}|$; and **O** uses only the hosts that have location-specific tasks, therefore, it is slightly higher than that of **E**, $|T|/|L_T|$.
3. **The average percentage of total cost:** indicates the amount of communication cost compared to the maximum cost in each workflow of the three configurations. In this simulation, we add one extra unit of cost for each causal link although all communications happened at the same host in order to demonstrate the activity that happens during the workflow enactment, otherwise **C** will always have zero cost. **C** communicates at a single host internally, then the cost is minimum; **E** communicates through out the system, so the cost is the highest among all; and **O** communicates only between selected hosts, therefore, the cost is between **E** and **C**.

The results of the simulation show that the proposed optimal MAS configuration both reduces the communication cost significantly and maintains a high level of system utilization. Its load balancing is better than that of centralized configuration, and obviously it cannot be better than that of equally distributed configuration as it does not use all of the hosts. With the proposed approach, we can achieve an optimal configuration that satisfies the hard constraints (location specific tasks) and optimizes the soft constraints (communication cost).

6 Conclusion and Future Work

We have presented a strategy to obtain an optimal MAS configuration for agent-enhanced data mining. The method utilizes the existing component in modern data mining tools, i.e., the workflow. The workflow is modeled with our proposed ML-POCL plan. The plan is then optimized and used to obtain an optimal MAS configuration. The result shows that the attained MAS configuration optimally reduces the communication cost and maintains a high level of system utilization.

However, there are a few issues yet to be improved. The constraint optimization process assumes all costs (\mathcal{C}) between tasks to be pre-defined. But in a more complex situation, the configuration should deal with cost function \mathcal{C} dynamically since the environment may change and affect it. Another issue is that most WfMSs use data pipe-lining to pass data from one task to another, while global resource sharing scheme allows data production from one activity to be published to the naming directory service for later reference and re-use. This will help boost re-usability of the resource.

References

1. Bauer, T., Dadam, P.: Efficient Distributed Workflow Management Based on Variable Server Assignments. In: Wangler, B., Bergman, L.D. (eds.) CAiSE 2000. LNCS, vol. 1789, pp. 94–109. Springer, Heidelberg (2000)
2. Buhler, P.A., Vidal, J.M.: Towards Adaptive Workflow Enactment Using Multiagent Systems. Information Technology and Management 6(1), 61–87 (2005)

3. Cao, L., Gorodetsky, V., Mitkas, P.: Agent mining: The synergy of agents and data mining. *IEEE Intelligent Systems* 24(3), 64–72 (2009)
4. Cox, J., Durfee, E.: An efficient algorithm for multiagent plan coordination. In: *Proceedings of the Fourth International Joint Conference on Autonomous Agents and Multiagent Systems*, pp. 828–835. ACM (2005)
5. Ehrler, L., Fleurke, M., Purvis, M., Savarimuthu, B.: Agent-based workflow management systems (WfMSs). *Information Systems and E-Business Management* 4(1), 5–23 (2006)
6. Huhns, M.N.: Agents as Web services. *IEEE Internet Computing* 6(4), 93–95 (2002)
7. Judge, D.W., Odgers, B.R., Shepherdson, J.W., Cui, Z.: Agent-enhanced Workflow. *BT Technology Journal* 16(3), 79–85 (1998)
8. Klusch, M., Lodi, S., Gianluca, M.: The role of agents in distributed data mining: issues and benefits. *IEEE Comput. Soc.* (2003)
9. Moemeng, C., Zhu, X., Cao, L.: Integrating Workflow into Agent-Based Distributed Data Mining Systems. In: Cao, L., Bazzan, A.L.C., Gorodetsky, V., Mitkas, P.A., Weiss, G., Yu, P.S. (eds.) *ADMI 2010. LNCS*, vol. 5980, pp. 4–15. Springer, Heidelberg (2010)
10. Moemeng, C., Zhu, X., Cao, L., Jiahang, C.: i-Analyst: An Agent-Based Distributed Data Mining Platform. *IEEE* (December 2010)
11. Odgers, B.R., Shepherdson, J.W., Thompson, S.G.: Distributed Workflow Co-ordination by Proactive Software Agents. In: *Intelligent Workflow and Process Management. The New Frontier for AI in Business IJCAI 1999 Workshop* (1999)
12. Savarimuthu, B.T., Purvis, M., Purvis, M., Cranefield, S.: Agent-based integration of Web Services with Workflow Management Systems. In: *AAMAS 2005: Proceedings of the Fourth International Joint Conference on Autonomous Agents and Multiagent Systems*, pp. 1345–1346. ACM, New York (2005)
13. Weld, D.S.: An Introduction to Least Commitment Planning. *AI Magazine* 15(4), 27–61 (1994)
14. Yoo, J.-J., Suh, Y.-H., Lee, D.-I., Jung, S.-W., Jang, C.-S., Kim, J.-B.: Casting Mobile Agents to Workflow Systems: On Performance and Scalability Issues. In: Mayr, H.C., Lazanský, J., Quirchmayr, G., Vogel, P. (eds.) *DEXA 2001. LNCS*, vol. 2113, pp. 254–263. Springer, Heidelberg (2001)

Towards a Numerical, Agent-Based, Behaviour Analysis: The Case of Tourism

Sébastien Corniglion and Nadine Tournois

Université de Nice Sophia Antipolis
Institut d'Administration des Entreprises
Laboratoire CRIFP EA 1195
Pôle Universitaire St Jean d'Angely
24, Avenue des Diables Bleus
06357 Nice Cedex 4 France
{sebastien.corniglion,nadine.tournois}@unice.fr

Abstract. We discuss who should be in charge of providing data relevant to marketing segmentation for the tourism industry. We review the most commonly found consumer behavioural models and discuss the difficulties of their integration within an information system. They will be oppose to a novel approach in marketing segmentation, based on out-goings analysis. We use agent-modelling techniques, based on cellular automaton rules and stochastic processes to implement our model and generate sales data. We then present our algorithm to identify similarly behaved tourists, showing that the commonly used “*nationality*” variable for segments discrimination is not efficient. We conclude with some test runs results discussion and possible further research tracks.

Keywords: Simulation, Stochastic processes, Cellular automata, Tourism, Business, Public Policy Issues, Management techniques, Marketing, Market segmentation, Customer behaviour model.

1 Introduction

This paper presents the outcome of a multi-agents simulation, designed to produce data relevant to the behaviour of tourists. The aim of this generator is two-fold: to produce artificial sales data from the tourists’ trail of transactions throughout their stay, for which no actual data is easily available, and, to emphasise the existence of homogeneous groups of “*tourists’ behaviour*”, in a market segmentation sense. Discounting the fact that the analysis of tourism activity is still rooted in splitting tourists into well-known variables, such as nationality or revenues, we formed the hypothesis that, although homogeneous behaviour groups may exist, the individuals composing them are not necessarily homogeneous. Hence, traditional models might not be leveraging enough decision-making abilities for devising tourism policies. After discussing who should carry the duty of knowledge gathering with regard to the necessary variables to be

fed into a marketing process, we will review the most common behaviour models found in the literature, discuss their suitability for the tourism industry and underline a strong paradox, as they cannot be operationalised within an information system. We will address this by presenting the simulation techniques used and our behaviour model, providing details on the algorithmic core of the generator, and particularly how it decides whether a tourist will visit a shop and proceed to buy. We will follow with our discovery approach for the similar “behaviour groups”, describe the preliminary results obtained and conclude with further possible research tracks.

2 Who Does?

Should the tourism industry follow the prominent definition given by Philip Kotler in his millennium edition of *Marketing Management*, it would be likely to raise more interrogations than actual actionable answers. “*The marketing process consists of analysing market opportunities, researching and selecting target markets, designing marketing strategies, planning marketing programs, and organising, implementing and controlling the marketing effort*” (p.86). Further, Kotler adds “*(and to) transform marketing strategy into marketing programs, marketing manager must make basic decisions on marketing expenditure, marketing mix and marketing allocation*” (p.87) [32]. Why do these clear definitions cause an issue for a touristic destination? Certainly because they have been conceived for addressing to self-contained organisations, whereas the nature of a destination is to be composed by a myriad of different actors, of all economical and financial sizes, only joined together by a common goal: increasing their revenues [45]. Hence, when Kotler proposes to analyse, select, design, plan, etc., the first question one may ask is “**who does ?**” A brief look at tourism stakeholders shows:

- Privately owned organisations, i.e. *restaurants, hotels, ...*
- Public sector of various levels, i.e. *local councils, regional assemblies, ministry(ies)*
- Public-Private joint-ventures, i.e. *some museums, historical venues, ...*

Such a diverse structure naturally brings the question of governance, as discussed in [25]. The nature of capitalist, “*free-market*” economies tends to self-justify that a (*highly*) competitive market such as tourism should regulate itself, under fundamentals such as the *supply and demand law* [55,17] and notion of *perfect competition* [46]. However, the presence of the public sector in the market induces a breach into this perfect competition ideal. Interventions of the *public hand*, nemesis of Smith’s “*Invisible Hand*”, i.e. funding and/or subsidising a museum, distort the competition for privately-owned entities. Subsequently, the neo-classic *free-market* theory would promote abolishing such an unfair competition. However, the financial and following economical events between the 2008-2011 period have favoured a return in grace of the *public hand*. Consequently, some

counter-arguments from the academic world have been greatly rewarded for refuting the classic theory, notably using the concept of “*information asymmetry*”, which formalises the studies of decisions in transactions where one party has more or better information than the other [5,57,58]. When observing the tourism ecosystem, this “*asymmetry*” seems to apply to the tourists, away from home and their comfort zone, sometimes from their own country, language and culture. They could form the “*ignorant*” party, who lacks information when agreeing to the terms of a transaction, hence perfectly fitting the definition of the “*adverse selection*”. However, when dealing with marketing a touristic area, we believe the productive actors to be the “*ignorant*” party, as nobody, from independent shops to the whole value chain, owns a global view on the tourists purchasing habits, making market segmentation and consequently, target marketing very difficult, if not *impossible*. This situation reminds of the definition of *Constrained Pareto Optimality* [59,28], where a potential planner may not be able to improve a market outcome, even in a situation of inefficiency. Nevertheless, this article aims to demonstrate that if an *economically benevolent party* possesses this *global view* by developing an information system based on novel theories of customers’ behaviour, delivering almost real-time business intelligence, and provides it for free to all the *interested stakeholders*, a *Pareto improvement*, making at least one individual better off, without making any other worse off, could be achieved for the value chain. Being economically benevolent is probably not quite reasonable to assume, however the ***public sector***, whom would also potentially gain from an increased touristic activity, seems in our eyes, the best actor to perform the duty. Hence, we suggest that the sought *Pareto improvement* could be achieved *if and only if* the ***public hand does act*** in providing necessary knowledge to all the market’s stakeholders for a more efficient marketing policy. This claim could be counter-argued by a simple management thought experiment: should one drain customers, *i.e.* revenues, away from business *B*1 to *B*2, the former would naturally become worse off as the latter becomes better off. However, [49] has shown that thanks to the mutually dependence of stakeholders in a tourism *cluster*, “*good performance by one [cluster’s member] can boost the success of the other*”, leaving the door open for the sought *Pareto improvement*.

3 Modelling Consumer Behaviour

3.1 Of “*Paradigms*” and “*Shifts*”

As any model should be built upon certain paradigm(s), we felt necessary to discuss the usage of the term. In spite of the word “*paradigm*” to have been commonly used for a long time, [33] has widespread its acceptance and usage in the marketing literature thanks to his “*Structure of Scientific Revolutions*”. His definitions of “*paradigm*” usually refer to three possibilities:

- as a complete view of reality or way of seeing,
- as relating to the social organisation of science in terms of different schools of thought,
- as relating to the specific use of instruments in the process of scientific puzzle solving.

The first one is the most commonly encountered amongst scholars, and we will adhere as well, for the sake of consistency, as this definition is shared between marketing and information science. According to Kuhn, science advances through *revolutions*, during which paradigm shifts can take place. He proposes that one paradigm prevails in a scientific discipline during a certain period, notably by providing ontological frameworks and philosophical foundations to scientists working towards this paradigm. As time goes, anomalies may arise for which the established paradigm fails to provide appropriate answers. A new one may hence emerge, challenging the old paradigm, or including it in a wider frame, which results in a “*paradigm shift*” [7]. To summarise, a “*paradigm shift*” relates to the *change of our axioms about the world*. In the following, we will examine some of these important *shifts*.

3.2 Main Models for Customers Behaviour

In modern management literature, *i.e.* post 90’s, two¹ models are often put forward, discussed and opposed:

- a. *Relationship Quality* Model², or *RQ*, by [51]
- b. *Theory of Planned Behavior*, or *TPB*, by [3]

The former emphasise the notion of *quality in the relationships* of the stakeholders, and particularly the notion of *loyalty*, in order to explain purchase intention and behaviour, when the latter distinguishes a *subjective norm*, defined as the attitude towards the behaviour along with the impact of relevant reference people, and the *perceived behavioral control* of a customer over the behaviour under study, which results in the formation of a behavioural intention, and eventually forms behaviour, as shown in figure 1. It could be questionable to immediately discard [51] model, as it advocates for a strongly entangled triad of customers, employees and investors loyalty, whereas we discussed the particular composition of the tourism industry, with its many actors seen in section 2, for which such control seems incompatible, or at least *difficult to set*. Moreover, the definition of customers’ loyalty as in “*a deep commitment of repurchase and favor without hesitation*” [47], is equally difficult to observe and analyse. However, it is also arguable for this commitment be analysed under larger umbrellas, such as the destination globally and/or its types of businesses (*i.e.* restaurants, hotels, *etc.*), which take part in the necessary relationships to discover for creating relevant holiday packages.

¹ The TPB model supplanted the “*Theory of Reasoned Action*” [23], from the same authors, hence will not be discussed here.

² Also known as the *Satisfaction-Profit Chain* model.

Relationship-oriented Approaches

Relationship Marketing The advent of the so-called “*relationship marketing*” (*RM*), first coined by [8] and then advocated by many scholars, was a major shift in marketing paradigm. Its original definition “*attracting, maintaining and in -multi-service organizations- enhancing customer relationships*” was rapidly widened from this service-customer relationship into ten different forms of relational exchanges [44], involving “*involve suppliers, lateral organizations, customers, or one's own employees or business units*”. Let us start with a rather concise definition of marketing: “*an exchange process where value is given and received between two or more parties*” [19], who argues that along the continuum of such relations, one will find “*transactional*” and “*collaborative*” exchanges. The former refers to “*one-shot*” types of exchanges, and forms a “*distinct beginning, short duration, and sharp ending by performance*” [22]. In this context, buyers and sellers take part in the process as a “*zero-sum game*”, in which “*the positive outcomes to one party are directly and equally matched by negative outcomes to the other as a result of their joint choice from interaction*” [50]. Anchoring into this “*game*” relates to the “*traditional*” *transactional marketing*, which [53] describe as based on two axioms:

- competition and self-interest are the drivers of value creation,
- independence of choice among marketing actors creates a more efficient system for creating and distributing marketing value.

The first one depicts that self-interests of different parties can be optimised and constrained by competition, conflicts, and mutual compromise in transactions, when the second implies that marketing actors should avoid obligation from each other to preserve freedom and flexibility in operations. In contrast, relationship marketers believe that value creation is eased by commitment and mutual trust, and suggest that cooperation provide better performance and competitive advantage [44], while they counter the second axiom by arguing that such independence and freedom would be achieved at higher transaction cost. Hence, **interdependencies** (as opposed to *independence*) will give rise to improved quality and reduced costs. A certain blur in the respective roles of the different marketing stakeholders results from this “*interdependency*” as buyers and sellers become partners, but also in the temporal and spatial boundaries between producers and consumers, as they become “*co-marketers*” [53]. In contrast, *collaborative exchanges* are characterised by “*very close information, social, and process linkages, and mutual commitments made in expectation of long-run benefits*” [19], where the process turns to a “*positive-sum game*”, and a “*win-win situation*” is likely to be achieved by cooperation and trust. With relationship marketing, valourising relationships becomes an equally, if not more important objective than instant profit generation. Some researchers, such as [65], have even suggested to extend this relationship continuum further, where both buyers and sellers become (*conceptually*) internalised in one fully integrated hierarchical organisation. We have seen that the traditional, transaction-based marketing view, is rooted in the principles of “*outcomes of exchange*” and “*value distribution*”, whereas

RM paradigm insists on “*value creation*” and the “*process of relationship engagement*” [53]. The latter is argued to be superior to the former because in a post-industrial economy, committed, long-term relationships can bring financial and competitive advantage [19]. However, RM is not free of criticisms. It has been noted that *buying* customers’ loyalty schemes, usually under the form of financial incentives, does not necessarily tie customers emotionally to the firm, but rather to the actual monetized advantages. Should a competitor offer the same or slightly better, the so-called relation could easily break [48]. In the service sector, where RM would seem to be the most adequate, the widespread use of information technologies by marketers, for process “*industrialisation*”, has sometimes been proven disastrous in dissolving long-standing relationships. The best example is certainly the large depersonalisation of relations by the Western banks in the 2000’s. Although IT decision and recommendation systems have done wonders to target relevant customers, the disappearance of the well-known figure of the bank manager has triggered millions of customers frustration and complaints, forcing the banks to reconsider what relationship marketing really meant. Finally, [52] have also shown that the canons of RM, and particularly the constant increase in revenues and lower retention costs of long-lasting customers, are not so obvious when the relation is non-contractual, which happens to be the case of a large part of exchanges. Nevertheless, it is important to underline that one does not need to believe RM as an advocate of sole long-lasting exchange. As we consider the customer’s relation as a continuum, its time duration could be accepted as “*short*”, hence rejoining Day’s view in traditional transactional exchanges not being necessarily “*wrong*”, but rather a *special case* of various relationship orientations. In reality, the *relationship model* is **richer** than the transactional marketing, as it can express the very same, and more.

The Network Paradigm. As *constructivism* goes, it has been argued by [44] than *Relationship Marketing* is a subset of a larger model: the *Network Paradigm*, where the ubiquitous competition amongst *business units* is claimed to be replaced by competition between *networks of firms*, and [2] see marketing as the *integrator and coordinator* of networked organisations. The foundation of this model goes back to the traditional microeconomic paradigm, before RM, where *transactions* are the *bond connecting* one firm with *its consumers and other companies*. In an era of industrial economy, large hierarchical, integrated firms relied on their marketing department to fulfil the [31] “analysis”, “planning”, “organizing”, and “control” functions to minimise transaction cost and maximise economic efficiency, hence the marketing goal to find buyers for sellers, and enable a set of independent transactions. In a postindustrial environment characterised by [1] with *high diversity, knowledge richness and turbulence*, “maximizing organizational learning and adaptive flexibility rather than economizing transaction costs becomes the critical organizing imperative” [2]. The previous organisation structure is then found to be inefficient with the three new elements quoted above. [65] refers to networks as “*the complex, multifaceted organization structures that result from multiple strategic alliances, usually combined with*

other forms of organization including divisions, subsidiaries, and value-added resellers". The network paradigm preaches for companies to *commit* resources on their *core business*, while *outsourcing* the rest of their organisation to specialised *partners*. Within network and quasi-network organisations (*i.e. partnerships and alliances*), the role of marketing changes. At the corporate level, marketers need to endorse a new role in designing and negotiating the firm's relationship with vendors and technology partners, while the business level sees an additional responsibility of deciding whether to outsource, partner, or internals perform certain marketing functions. Finally, the functional level encompass a new task in becoming a pro-customer champion and maintaining long-term relationships with customers and organisations. With these, [65] injects RM and transactional concepts into the paradigm. On a rather academical aspect, it is interesting to notice that network marketing theorists have overturned the traditional role assigned to marketing and marketers. When early marketers typically claimed a *neutral role* between manufacturers and customers (as middle-men), they were stereotyped to ally with producers from the consumer's perspective. As a result, [31] used to acknowledged that "selling" rather than "buying" activity was closer to the core meaning of marketing. As firms started to house marketing staff, a transition was made "*from seller of a firm's outputs to key player in shaping a firm's products, technologies, marketing policies, and strategic direction*" [2], because technological developments and the "network economy" give marketers an opportunity to reposition to get closer to their customers. This new shift was devised as "*being an agent of the seller to being an agent of the buyer, from being a marketer of goods and services to being a customer consultant and manager of his or her saleable consumption assets*". The most prevalent example would be the changing roles of intermediaries when looking at electronic marketplaces [26]: information gathering costs are declining, which in turns lowers information asymmetry, resulting in a far better *costs transparency*. They now face options of *pure disintermediation, re-intermediation* (*i.e.* repositioning) or *cybermediation* (*i.e.* embracing IT). Interestingly, the latter twos are supporting the network paradigm claims, as intermediaries are switching their role from solely serving companies (*selling products to consumers on behalf of the producers*), to serving customers (*seeking the best option for customers from a number of offerings*).

The Service-Dominant Paradigm. The reader may have noticed the previously discussed paradigms being strongly "*products oriented*". It is rather odd in an era when the world's GDP is composed by 63.2% of services³ [13], that all the main marketing models revolve around the idea of products and production. As [54] pointed out early, "*the classical marketing mix, the seminal literature, and the language of marketing all derived from the manufacture of physical goods*". Nevertheless, as the reality shifts, it was only natural than marketing should follow. Specifically, this new logic focuses on *intangible* rather than *tangible* resources, co-creation of value rather than embedded value, and relationships rather than transactions [61]. According to them, a shift of the dominant logic is the result

³ 2010 estimate.

of a changed understanding of resources and value. In their taxonomy, resources can be divided into two categories: *operand* and *operant* resources. The former refers to “resources on which an operation or action is performed to produce an effect”, while the latter are “resources that produce effects”. Simply put, operand resources are physical, while operant resources are typically “human”, “organisational”, “informational”, and “relational” [29]. In a “tangible economy” where physical materials are the source of competitive advantage, goods are the primary unit of exchange and the ultimate “gold-standard” of value. Operand resources (*e.g.* land, minerals, or even customers) are the synonym of *wealth* and possessing them represents *economic success*. Hence the producer determines the value of tangible good, as it is embedded. The exchange between different parties focuses on the transaction of operand resources, and goods are valuable only in terms of the “value-in-exchange” [61]. Finally, this goods-centered view sets the standard, terminology and performance evaluation system for marketing, as services, from a manufacturing perspective suffer from inferior characteristics such as intangibility, inseparability, heterogeneity, and perishability [37,62]. It was very common to hear that marketers had to “*industrialise*” their offerings [48]. Nevertheless, one can only observe knowledge and information, the *operant* resources, steadily replacing physical goods and capital, the *operand* resources, as the major impersonation of value⁴, upsetting the validity of the original goods-centered logic. In the S-D view, although marketing is still about *facilitating exchanges*, the ultimate object of exchanges has switched from valuable *goods* to *services*. The art becomes a continuous set of “social and economic processes”, which generally begins “with an interactive definition of the customers’ problem”. The role of tangible goods is merely reduced to a vehicle to satisfy customers’ needs and desires, while services bundled with goods become what *differentiate* a product offering from others⁵, as competitive value propositions rely on an organisation’s operant resources, and its *network* interactions grow with increase of the said operant resources. [61,39] have proposed a set of foundational premises of this logic, namely:

- All economies are service economies.
- Indirect exchange masks the fundamental unit of exchange.
- The application of specialised skills and knowledge is the fundamental unit of exchange.
- Knowledge is the fundamental source of competitive advantage.
- Goods are distribution mechanisms for service provision.
- Organisations exist to integrate and transform micro-specialised competences into complex services that are demanded in the marketplace.
- A service-centered view is customer oriented and relational.
- The customer is always a co-creator of value.

S-D logic promotes organisations to be *customer-centric*, *market driven*, and *learning-oriented*. It insists the value of a product is “value in use” defined by

⁴ The shift is also transposable in the accounting domain, *e.g.* the *IFRS* quest for “*fair value*”.

⁵ This is even more true when the business domain is *regulated* by a given authority.

customers, as opposed to embedded “exchange value” decided by sellers, hence the marketers’ responsibility to enable the maximisation of product customisation, notably with customers’ involvement in the production process. The new logic relies heavily on the importance of operant resources, and stresses the coordination and integration of multiple functions in an organisation to operate in a service-centered model [20]. It is also good to notice, according to [20], that the focus on an operant-resource-based, service-centered view, does not complete refutation of the goods-centered logic, as both are likely to coexist for a long time. As with transactional and RM, most organisations will need to reorient, readjust, or *subordinate* the goods-centered logic.

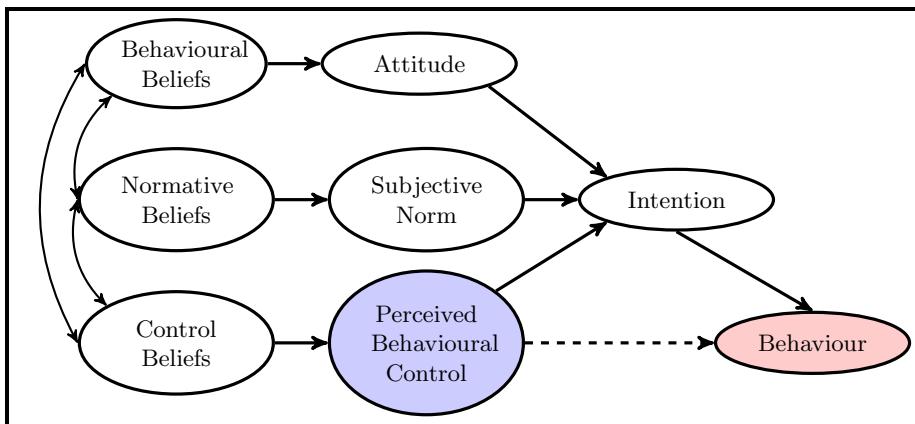


Fig. 1. Concepts & Transitions *Theory of Planned Behavior* model

Theory of Planned Behavior. An alternative approach to predicting intentions and behaviour that is widely used in consumer behaviour research is the “*Theory of Planned Behavior*” [3]). It postulates three conceptually independent determinants of intention: *attitude* towards the behaviour, *subjective norm*, and *perceived behavioral control*. The relative importance of each antecedent may fluctuate change with regard to behaviours and situations. TPB also suggests that intentions are the immediate antecedent of behaviour and they fully mediate the impact of attitude towards the behaviour, as well as subjective norm on behaviour. However, they partially mediate the impact of perceived behavioural control. Most empirical applications of the TPB try to explain or predict newly introduced behaviour, which is fundamentally different from the customer-firm relationship context, as the study lies in the impact of intentions to *shift from a habit to a newly introduced behaviour*. Nevertheless, some authors have assessed how TPB could be used to assess how attitudes, subjective norm, perceived behavioural control and intentions predict the extent to which *existing behaviour* will be *repeated or reinforced in the future*. In the case of our field of studies, this would be a more relevant application than the “*new behaviour*” one. [24]

suggests that attitudes that were formed on the basis of past behaviour may be more stable predictors of subsequent behaviour than attitudes that are not based on behavioural experience, when [15] has shown that TPB modelling can provide effective predictors in the family restaurant business, with the same interest in repeat patronage. In order to operationalise the model, TPB requires the following three conditions to hold in order to be used for a study [3]:

- i. the measures (*i.e. the survey's questions*) of perceived behavioural control and intention must be strictly semantically identical,
- ii. *intentions* and *Perceived Behavioural Control* must remain stable in time between their evaluation and observation of the behaviour,
- iii. *confidence* in measurement *accuracy* of PBC.

If the first requirement seems obvious and easy to meet, the other two need a discussion. In the field of tourism, it is rather easy to imagine a couple, with no children, in the median line of their country's household revenues, who have either saved or using a credit card to fund their "*dream*" yearly holidays. Let us fix an arbitrary *-nonetheless realistic-* length of stay of seven days. The said couple may have started with strong intentions to treat themselves to a few top-range restaurants dinners. Their PBC is also high, as they have a *known amount* of money available to spend. However, after the first dinner, and for whichever reason, the experience did not live up to their expectations: they could have felt uncomfortable, judged the quality/price ratio was not good enough, etc. In this case, intentions and PBC will immediately diverge, hence falsifying Ajzen's second requirement. He was prudent enough to warn that "*intervening events may produce changes in intentions or in perceptions of behavioral control, with the effect that the original measures of these variables no longer permit accurate prediction of behavior*". As for the third, the same explanations apply, showing the same difficulty of use for our field of research. It is important to note that this divergence in intention and PBC is supported by [56], who refers to the [42] hierarchy of needs, classifying travelling and tourism in its highest "*self-actualization*" stage, which is the most difficult to achieve, and would rather be satisfied by occasional "*peak experiences*". [4] himself has review the predictive power of his model compared to domain-specifics constructs (*e.g.* health or voting behaviour), which did not perform essentially better, or even sometimes worse, than the more general TPB. Interestingly, [21] compares the predictive power of the TPB with the RQ model, assuming possible comparison as they are both "*cognitive psychological frameworks that assume a path from attitudinal antecedents over intentions to behaviour*". They conclude a throughout review with important findings:

- *RQ* and *TPB* are **interchangeable** models to *uncover the dynamics of the customer-firm relationship*.
- *TPB* constructs outperform *RQ* in *predicting intentions*, hence its predictive power seems to be higher than the *RQ*
- The three TPB constructs have to be taken into account, as they all contribute to the explanatory power of the model. More importantly, leaving any construct out of the model would lead to incomplete conclusions.

If [21] seem to favour the TPB over RQ, they also make a point in mentioning that a TPB study would only be valid when measurements of cognitive and actual behaviour are close in time, which rejoins Ajzen's second requirement. The scenario discussed above or the idea of measuring intentions of tourists even before they would have booked prevent us to meet this condition. For all these reasons, we will avoid using Ajzen's model in our field of studies.

3.3 Lifecycle of the “Tourism Product” and IS-fitness of Behaviour Models

In order to analytically connect customers to their behaviour, a clear understanding of the addressed market, with its products and services, should be established. Swarbrooke [60] describes it as “*complex and multi-layered*”, as it is not only constituted by tangible items (*food, accommodation*) and intangible items (*easiness of road access, cleanliness of the place*), but also varies in timespan and monetary value. Customers (the tourists) are away from home, hence their behaviour may be influenced and changed from their usual lifestyle [27,40,16]. They are looking for an overall *experience*, which includes the pre-trip phase (*anticipation*), the *global* consumption during the stay, and concludes with the *memory* of the stay, after returning home. The actual production process of the “tourism product” involves many different missions: the shop-owner will aim to make his window as attractive as possible, whereas the local authority will make sure the extra traffic generated will be bearable for both the tourists and the locals. Moreover, the consumer-tourist also influences the production process, as an individual’s behaviour may influence the experience of fellow tourists. This is particularly true in the case of a “complainer” who may trigger a complaining atmosphere around him. Nonetheless, all these entities share a same goal in increasing touristic revenues, as discussed by Moutinho [45]. Also, and like many others, a new challenge is facing the tourism industry: the Internet. Bonn, Furr & Susskind [10] and Luo, Feng, & Cai [38] have shown that tourists who searched the Internet to select their destination, but also to gather information before departure, had a tendency to spend more during their stay, compared to other sources of information. The most successful online community for tourism, namely TripAdvisor®, has been shown by Wang & Fesenmaier [64] to be a powerful platform for interaction between peers by notably propagating satisfaction and impacting the revenues of the commented locations. All these elements underline the pace needed for the industry to adapt its offers in order to be as close as possible to its customers’ needs, recall our concerns on who provides the needed information and call for a market segmentation, defined by Venugopal & Baets [63] as “*a process of dividing a market into distinct groups of tourists who might require separate experience or marketing service mixes*”. Target marketing, described by Kotler [32], then follows, aiming to develop products and marketing programmes tailored for the found segments. However, the situation is paradoxical: on one hand, marketers and the nature of the touristic market suggest a “*need for speed*”, and on the other hand, the only analytical tools provided *by and to* the formers, being RM variations (*including transactional*) or TPB models have

been built to study and explain behaviour *a posteriori*, once surveys and questionnaires, generally *in situ*, using a typical Likert scale structure [36]. They are not readily usable in an knowledge discovery IS system, as the variables they describe cannot be assimilated to “computer science” variables. Actually, they are a guidance to the researcher in tailor-making questionnaires, hence their openness to interpretations do not provide (*and do not claim to*) a stable data model, fit for being instantiated in an information system’s database. In order to build the latter, the standard information system lifecycle applies [12], and requires to understand the business and the product’s lifecycle to be computerised. This is why we believe the standard process of selecting a model, organising an empirical study and analysing the results, which will mostly concentrate on explaining the inner causes for tourists’ behaviour, in order to eventually transform them into actual marketing actions, is both structurally unsuited and too slow for the tourism industry: most of the time, a well-conducted study will last longer than a whole touristic season. Hence, we propose to concentrate first on constructing a clear picture of the tourists’ buying habits by analysis of their most revealing data: their outgoings. We believe the ability to know the habits segments is more important for immediate marketing action than explaining what caused these habits. This idea was approached by Bloom in [9], where he presented a novel segmenting approach, opposing linear and non-linear analyses techniques. He remarked that most behaviour segmenting analyses are performed using linear methods, where a given segment is characterised by a function affecting different weights to each entry feature (*i.e. variable*). In spite of the actual tourist behaviour being indeed expressed by these features, they are linked by non-linear relationships, as they interact between each other, *i.e.* length of stay multiplied by the money spent *on a daily basis*. The latter approach allowed to increase by almost twice the segments’ coefficient of determination (R^2), using *data mining* techniques. It led us to believe that using classification techniques from the computer science field of *data mining* would allow to describe homogeneous groups of similar buying habits. This type of data analysis is much quicker to set up and run, and would provide profiling elements to tourism policy-makers, who could then adapt their offers to the market needs. Using data mining techniques for the tourism industry has been thoroughly reviewed by Law et al. [35] but was called for further work, mining this field being described at “*infancy stage*”. Out of 174 papers selected from a 28-year span, only 14 were using mining techniques, all solely based on forecasting tourists arrival by nationality, which is not quite the same as identifying segments of similar behaviour, unless taking the hypothesis that the single variable of “*nationality*” is a determinant to create clusters of behaviour. This idea is indeed commonly found through many studies on tourism, being public or private sector-funded. However, we desired to challenge this widely-accepted postulate by choosing, as discussed above, the tourists’ outgoings as main criterion of discrimination. Hence, before running our framework with actual sales data, which would require partnerships with many of the industry stakeholders and involve legal aspects for individuals’ privacy, we have proceeded in generating artificial sales data, using agent-based

computer simulation. Investigating for new management science theories using computer-based simulations have become increasingly popular in the last five years, especially thanks to the formalising work of Davis, Eisenhardt and Bingham [18], who propose a framework for developing such tools, detailed in the next section, and on which we based our simulator.

4 Simulation, Simulators and Models

4.1 Numerical Simulations Applied to Management Science

The common lack of empirical data, evoked by Zott [67], is one major argument in favour of creating relevant, however artificial data. Nevertheless, numerical simulations in the management science fields had long been thought to be either too simplistic as they would remove important correlations or too basic as they would only replicate a well-known phenomenon, as discussed by Chattoe [14]. More recently, new agent-based techniques [34] have arisen, where a software could make an *agent* (actor) evolve independently from all the others in either controlled, semi-controlled or random fashion. With respect to these advances, Davis et al. [18] discussed above, have provided an extensive review on computer-based simulation methods used for developing and evaluating new management science theories. Their paper distinguishes five types of simulation, four being targeted for a specific research question:

- *System dynamics*, which focuses on a whole system behaviour with complex timing and causality and looks for what causes system instability.
- *NK fitness landscape*, which focuses on the effectiveness and pace of a system to reach an optimal point and looks for the conditions of a high-performing strategy.
- *Genetic algorithms*, which focuses on adapting a population of agents towards the best type of agent and looks to explain the factors of this best agent.
- *Cellular automaton*, which focuses on creating ruled micro-interactions for observing macro-consequences and looks to explain how, when and how fast a pattern emerges.
- *Stochastic processes*, which focuses on random variations of the input variables and observes how this randomness impacts the agents population.

The stochastic processes have the particularity of not being as specific, but in our opinion, they come as a great help to computerise the most challenging aspect of simulating human behaviour: randomness, or at least “controlled randomness”. A stochastic process can be left totally random, or be contrived to a

probability distribution. In our eyes, using the properties of cellular automata, but loosening their micro-interactions rules via a contrived stochastic process, create a decent proposal to depict human behaviour. In consequence, we chose to use the NetLogo environment⁶, which allows implementing cellular automata-type rules at “agent” and “world” level as well as providing a random engine for stochastic processes. The latter is qualified as a “*pseudo-random*” generator, for a computer being a deterministic machine, which is not qualified by definition to provide real randomness. If the discussion of the quality of a pseudo-random algorithm is out of the scope of this article, it is to note that NetLogo provides two types of these:

- i. a general *random* functionality, based on the *Mersenne-Twister* algorithm[43], well accepted as a high-quality random generator, notably passing the DieHard tests[41],
- ii. other random functionalities, contrived to probabilistic distributions, namely *gamma*, *normal*, *exponential* and *Poisson*.

The lack of real-world randomness involved by pseudo-random generators can in fact be turned into a precious feature for running repeatable trials: these algorithms have to be “*seeded*” with a value before they start generating numbers and from this “*seed*” depends the order of the generated numbers. Although seeded by default with the computer clock value, the seed value can be forced into a *hard-coded* one, insuring a simulator to obtain the exact same order of numbers generated for every run. Although we did not use this feature for our work, adding a simple line of code would allow reviewers to repeat the stochastic aspects of our simulator in the exact same conditions.

4.2 Agents Model

Our work deals with two types of agents: the tourists and the “shops”, where a shop is to be understood as *any place where a tourist can spend money*. We defined tourist and shop-specific variables, for which each instantiated individual would carry its specific values:

- a. *Tourist*
 - i. *Age*
 - ii. *Country of origin*
 - iii. *Type* (*i.e. single, couple or family*)
 - iv. *Revenue available*
 - v. *Length of stay*
 - vi. *List of visited shops*
 - vii. *Total money spent*
 - viii. *List of affinity factors to types of shops*
 - ix. *List of money spent per types of shops*

⁶ <http://ccl.northwestern.edu/netlogo/>

b. ***Shop***

- i. *Type of shop*
- ii. *Total number of visits*
- iii. *List of sensibility factors to countries*
- iv. *List of sensibility factors to age groups*
- v. *List of sensibility factors to types of tourists*
- vi. *Average amount of money spent per visit*

Stochastic Aspects. The simulation distributes randomly initial locations, nationalities, age and duration of stay for the tourists and for the shops' locations.

Cellular Automaton Aspects. For the sake of clarity of results, we have limited the types of shops to seven, namely: *hotel, restaurant, bar, museum, take-away, souvenirs, art gallery*, and same for the tourist's nationalities, limited to eight: *GB, USA, Italy, Spain, Russia, Netherlands, Benelux and France*. Revenues are ruled by indicating an average daily expenditure per nationality. This element could also be left to random, but we believed that using easily obtainable expert insight would make the generated data closer to reality. With regards to the shops, the simulator will apply two rules. First, the sum of hotels, bars and restaurants must hold between $\frac{3}{5}$ and $\frac{4}{5}$ of the total numbers of shops. This assumption has been verified in almost all touristic venues we have been observing from our local area, which is one of the most active in Europe. Then, each type of shop (bar, restaurant, ...) is initialised to an average amount of money spend per tourists' visits, which could also be gained from expert knowledge. However, this variable is adjusted through the simulation process, when a tourist buys in the shop. The different variables of "affinity factors" designate dictionaries of real numbers in $[0; 1]$ which can be assimilated to a probability expressing the chance for an instance to be attracted to a target. For example, the list of affinity factors of shops in a tourist-agent, using types of shops as keys, shows the probability of a given tourist to be attracted to a particular type of shop. Finally, all the factors are initialised to zero and will evolve every time a tourist will visit and buy in a given shop (*same applies for the shops' sensibilities*⁷).

5 Simulation's Decision Core

With the agent-model established, we propose the decision core, which focuses in making tourists visit and buy in a shop, to be designed with the same duality between cellular automaton and stochastic processes. The latter allows the unpredictability of human behaviour, while the former forces the decision core to observe some commonly-observed behaviour generalities.

⁷ If the meaning of the tourists' affinity factors are self-explanatory, please note that in terms of management science, the shops' sensibility factors could be assimilated to the notion of *reputation* amongst their respective categories.

5.1 Decision: Stochastic Aspects

Tourists. The tourists are “free” of movements around the streets, which are modelled to be pedestrian-only. When reaching a crossing, a tourist has exactly $\frac{1}{4}$ chances to head toward one of the four cardinal points. If a tourist-agent arrives in front of a shop, the core will observe its affinity factors and apply (*table 1*):

- a. *all affinity factors are zero-valued (notably at the start of the simulation):* the core refers to an interactively-set global constant $VB1$ in $[0; 1]$ expressing the probability of not visiting a shop when the tourist-agent has no affinity,
- b. *affinity factors are different from zero:* the core refers to an interactively-set global constant $VB2$ in $[0; 1]$ expressing the probability of not visiting a shop when the tourist-agent has an affinity for the shop’s type.

Hence, the “*dice is rolled*” each time a tourist-agent passes by a shop, no matter its affinities, giving it one out of $VB1$ or $VB2$ chances for visiting. The corresponding pseudo-code is described in algorithm 1. A subsequent purchase would also *not* necessarily affect the tourist-agent’s affinity for the type of shop (*it might not have appreciated the shop’s experience*), and the same method of global constant applies in the core for deciding whether it will increase the relevant affinity factor by the money spent⁸. Finally, when a tourist-agent leaves a shop, its new heading direction will also be left to an *equidistributed* random choice.

Shops

- a. Once a purchased is settled, the shop feature *Average amount of money spent per visit* will not automatically be affected. An interactively set constant FB (*see table 1*), allows to provide the core with the probability of updating this feature. This allows representation of *influential* individuals, *i.e.* those who would diffuse their experience to the community (TripAdvisor[®], …).

5.2 Decision: Cellular-Automaton Aspects

The rules implemented for the tourist and shop-agents contrive their stochastic movements and buying behaviour, in order to fit a real-world generalities.

Tourists

- a. **α condition:** a tourist can only settle one hotel payment during its whole stay. If an hotel has not been chosen by the decision core during the tourist-agent existence, the core will force the presentation of *all the hotels* to the agent through the decision algorithm before its “departure”⁹,

⁸ Increasing affinity factors is performed using a increasing and continuous minimisation function, imaging its results in $[0; 1]$, namely $f(x) = \frac{x}{x+10}$.

⁹ Please note that under this rule, a tourist may not be paying an hotel, which we thought to be representative of people visiting a city without necessarily staying there.

Algorithm 1. Visit (*aTourist*, *aSetOfShops*)

Ensure: *aTourist* is in reaching distance of *aSetOfShops*

Ensure: $(\alpha, \beta, \tau_1, \tau_2)$ conditions

for all shops in *aSetOfShops* **do**

- Add* all shop's sensitivities to *aTourist* characteristics
- Gather* all *aTourist*'s affinities for current shop's type

end for

if (All shop's sensitivities = 0 and *aTourist*'s attractions = 0) **then**

- Attrib Random* *theChosenShop* from *aSetOfShops*
- Roll dice* for not visiting *theChosenShop* *using* VB1

else

- Attrib* shop with max sensitivities to *aTourist* in *sShop*
- if** *aTourism* has any non-zero affinity factor **then**

 - Attrib* shop with max attractiveness for *aTourist* in *tShop*

- end if**
- if** *sShop* != *tShop* **then**

 - Attrib* *theChosenShop* *with*

Ensure: *Roll dice* $\frac{1}{4}$ chances for choosing *sShop*

end if

Roll dice for not visiting *theChosenShop* *using* VB2

end if

if *aTourists visits* *theChosenShop* **then**

Ensure: λ condition

- if** *aTourists buys in* *theChosenShop* **then**

Ensure: σ and μ conditions

- end if**

end if

- b. **β condition:** with regard to the previous rule, if a tourist-agent has access to multiple shops at once *and* has already paid its hotel bill, the latter is removed from the list of possible attractive shops,
- c. **λ condition:** a tourist may enter a shop but not necessarily buy anything, as the possible purchase is randomly generated in the interval $[0; Average\ amount\ of\ money\ spent\ per\ visit]$ of the chosen shop. This rule allows the natural fact of visiting a shop but not being *able* to buy,
- d. **μ condition:** the possible purchase value is generated randomly in the previously mentioned interval, but contrived to the *exponential distribution*.

Shops

- a. **σ condition:** once a purchase is settled, the three sensitivity factors of a shop-agent are raised by the minimised value of the purchase value (*see §5.1*), but affected by weights as follows: 0.4 for sensitivity to *age*, 0.4 for *country* and 0.2 for *tourist type*,
- b. **τ_1 condition:** visits in shop-agents other than *restaurants and bars* may only happen between 9am and 7pm,

- c. **τ_2 condition:** visits in shop-agents instances of *restaurants or bars* may happen between the intervals 8am to midnight and midnight to 3am.

Table 1. Simulation global constants

Constant	Description
D_MAvgAbsPurch	Maximum deviation allowed around the mean average absolute deviations of purchases vectors
D_MAvgAbsFact	Maximum deviation allowed around the mean average absolute deviations of affinities vectors
D_PearsPurch	Minimum value allowed for the Pearson product-moment correlation coefficients of purchases vectors
D_PearsFact	Minimum value allowed for the Pearson product-moment correlation coefficients of affinities vectors
VB1	Probability of not visiting a shop for a tourist with zero valued affinity factors
VB2	Probability of not visiting a shop for a tourist with non-zero valued affinity factors
LB	Probability of not liking (<i>i.e. not increasing affinity</i>) a shop for a tourist after a purchase
FB	Probability of not providing community feedback (<i>i.e. not increasing shop's sensitivities factors</i>) of shop by a tourist
VS	Number of vertical streets
HS	Number of horizontal streets
STN	Number of “single” tourists
CTN	Number of “couple” tourists
FTN	Number of “family” tourists

6 An Approach for Identifying Tourists Groups of Similar Behaviour

As we discussed in section 2, our two main concerns with traditional customers' behaviour models were:

- a. their inability to be quickly integrable in an information system,
- b. their inner motives, which are designed to explain why the customer behaved the way he/she did, which we understand to be of utmost importance, but only after identifying groups of similar behaviour.

Moreover, we evoked the recurrent habit of designing tourism studies by starting to split the tourists using their nationalities. On the contrary, we define a ***homogeneous behaviour group*** as a set of people, with no regard for their age, nationality, family profile or revenues, who spend their holiday money in the same types of shops, and for each type, ***within analyst-chosen levels of monetary value***. Using this definition, our simulation computes groups of similar behaviour by calculating the mean average absolute deviation of tourists' affinity factors for types of shops as well as their actual purchase values, and, the ***Pearson product-moment correlation coefficients*** between the two sets of factors vectors. Between a tourist and its potential similar peers, the following must hold (see table 1):

- a. the ***mean average absolute deviations*** of the *affinity factors* and *purchases values* lists must be of the same dimensions, ± interactively set global constants, *D_MAvgAbsFact* and *D_MAvgAbsPurch* respectively,
- b. ***Pearson product-moment correlation coefficients*** of the *affinity factors* and *purchases values* vectors must be of the same dimensions, ± interactively set global constants, *D_PearsFact* and *D_PearsPurch* respectively.

The *Pearson coefficients* are only used as a tool for ensuring that the values in the vectors conveying the affinity factors and purchases are ordered in the same way, as the sole usage of the mean average absolute deviation could not provide this guarantee when producing equivalent values. Currently, the simulation has been implemented to establish the behaviour groups at the end of days 3, 5, 7, 10 and 14. The process of linking a tourist to its similarly behaved peers is detailed by pseudo-code in algorithm 2. As we will detail with the results in the next section, the interesting aspect of this model is to allow some elasticity on the behaviour similarity definition, by varying the deltas of the four compared variables. In fact, two combinations could be of potential interest in market segmentation:

- a. a low admitted delta for both of the two mean average absolute deviations (affinities and purchases) would lead to close financial dimension of purchases, as well as similar attractions for the very same types of shops¹⁰,
- b. a low admitted delta for the affinities mean average absolute deviations, but higher for the purchases, would lead to similar attractions for the very same types of shops, but in a looser financial dimension¹⁰.

7 Results Obtained

The simulation has been implemented using *NetLogo 4.1.1* and offers a graphical interface for running trials. It allows watching tourists moving hour per hour until they 'depart' (see figure 2), entice them to buy with respect to the algorithm seen in section 5 and records every purchase made. On certain

¹⁰ Please note the *Pearson coefficients* have to be set in the exact same high values for these hypotheses to hold.

Algorithm 2. Link Similar Peers

Ensure: *aTourist* has non all-zeros affinity factors

```

 $myAvgAbsDevFact \leftarrow MeanAvgAbsDev(aTourist.Affinity_Factors)$ 
 $myPersonFact \leftarrow PersonCorr(aTourist.Affinity_Factors)$ 
 $myAvgAbsDevPurc \leftarrow MeanAvgAbsDev(aTourist.Purchases)$ 
 $myPersonPurc \leftarrow PersonCorr(aTourist.Purchases)$ 
 $otherTourists \leftarrow all\ others\ aTourist\ and\ not\ have\ zero\ aff\ factors$ 
for all currTourist in otherTourists do
    currAvgAbsDevFact  $\leftarrow$ 
        MAvgAbsDv(currTourist.Affinity_Factors)
    currPersonFact  $\leftarrow$  PersonCorr(currTourist.Affinity_Factors)
    currAvgAbsDevPurc  $\leftarrow$ 
        MAvgAbsDv(currTourist.Purchases)
    currPersonPurc  $\leftarrow$  PersonCorr(currTourist.Purchases)
    if  $abs(myAvgAbsDevFact - currAvgAbsDevFact) \leq D\_MAvgAbsFact$ 
        and  $abs(myAvgAbsDevPurc - currAvgAbsDevPurc) \leq D\_MAvgAbsPurch$ 
        and  $abs(myPersonFact - currPersonFact) \leq D\_PearsFact$ 
        and  $abs(myPersonPurc - currPersonPurc) \leq D\_PearsPurch$  then
            Link aTourist with currTourist
        end if
    end for

```

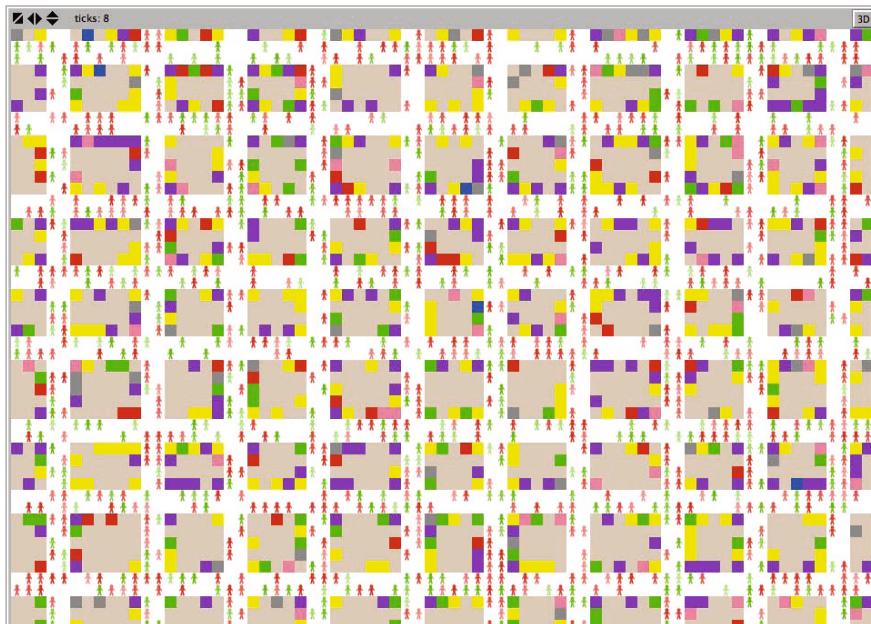


Fig. 2. Simulated tourists, shops and city

simulated days, the program will compute the behaviour groups, with regard to the model defined in *section 6*, and the user can graphically see links appearing between tourists (*see figure 3*). As the simulator decision-core is influenced by

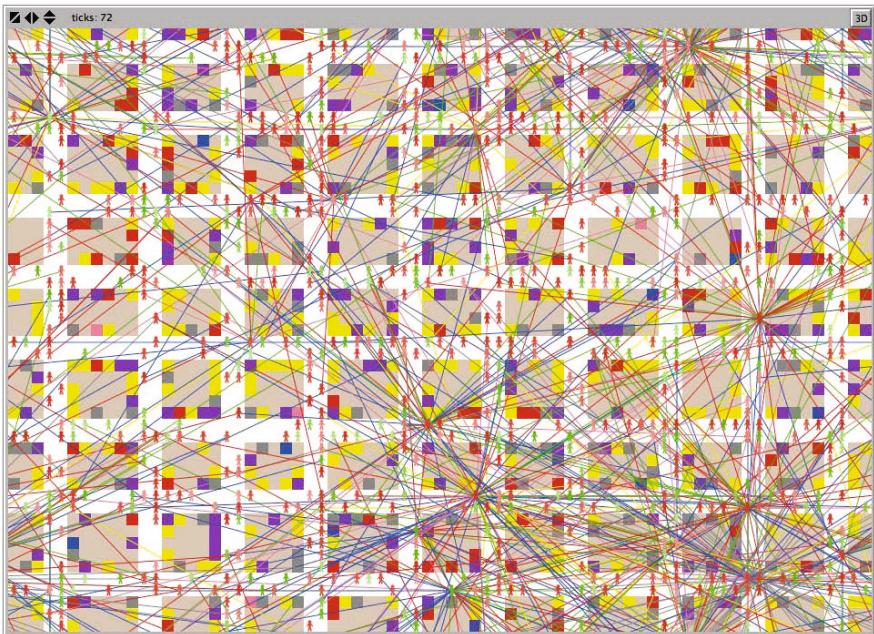


Fig. 3. Graphical identification of tourists with similar behaviour

the global constants (*see sections 5, 6 and table 1*), we used two sets of constants for running our trials as described in table 2. These two runs have respectively produced 7,867 and 9,152 lines of data conveying tourists' purchases, tourists' and shops' variables evolution, all globally spanning through respectively fifteen and seventeen simulated days. We discussed in the previous section that changing the acceptable delta of the affinities mean average absolute deviations between tourists would show similar buying behaviour segments, but with different financial dimension. However, we wanted to make sure of the intra-heterogeneousness of the linked tourists, with respect to their inner attributes (*nationality, type, length of stay, ...*). Hence, for preserving stability in comparing the results, we only modified the global constant $D_MAvgAbsPurch$ between the two test runs. For the same reason, the distribution of the tourists' nationalities has been contrived to the same proportions for the two runs, as shown in table 3. Tables 4 and 5 confirm our hypothesis held no matter the financial dimension chosen. In fact, setting a larger delta for $D_MAvgAbsPurch$ allows an even better stability of the ratio between the total number of tourists linked for a similar behaviour

Table 2. Global constants values for test runs

Constant	Test Run 1	Test Run 2
<i>D_MAvgAbsPurch</i>	0.05	0.7
<i>D_MAvgAbsFact</i>	0.05	0.05
<i>D_PearsPurch</i>	0.85	0.85
<i>D_PearsFact</i>	0.85	0.85
<i>VB1</i>	0.6	0.6
<i>VB2</i>	0.3	0.3
<i>LB</i>	0.4	0.4
<i>FB</i>	0.9	0.9
<i>VS</i>	10	10
<i>HS</i>	8	8
<i>STN</i>	300	300
<i>CTN</i>	400	400
<i>FTN</i>	350	350

and the subset formed by the ones of different nationalities. We have also observed the actual linked individuals, as shown in tables 6 and 7. If their numbers vary greatly with regard to the chosen financial dimension, it is remarkable that the most important period for observing the largest numbers of group leaders spans between three and five days regardless. When marketing a destination, it would imply that the available offer must “hit” the tourists as early as this time interval, to make sure of influencing the group leaders. One could wonder whether the group leaders were the ones who did affect the shops’ sensitivities factors, hitting the probability set by the constant **FB** (*see table 1*). However, we were not yet able to formally verify this hypothesis, as at the time of the test runs, the simulator was not designed to record whether a tourist was changing a shop’s factors. We can however conclude with these emerging patterns:

- a. the heterogeneous intra-group is very high with regard to usual variables, and particularly the “*nationality*” **confirming our primary hypothesis**,
- b. the number of groups is large in the first three days of simulation time, and then suffers a drop of almost a half each time the simulation hits the market traditional lengths of stay (5, 7 and 10 days). Past the 14-day milestone, the drop is above ten times from the 10-day one,
- c. a phenomenon of group leaders is clearly observable at any time, as very few tourists get highly connected to a large number of individuals.

Table 3. Tourists' nationalities distributions

Nationality	Test Run 1	Test Run 2
<i>Benelux</i>	105	105
<i>France</i>	209	209
<i>Italy</i>	158	158
<i>Netherlands</i>	105	105
<i>Russia</i>	53	53
<i>Spain</i>	105	105
<i>United Kingdom</i>	210	210
<i>United States</i>	105	105

Table 4. Analysis of similarly behaved tourists for Test Run 1

Day	A. Behaviour	B. With tourists of	C. With tourists of	Ratio
	links	s. nationalities	diff. nationalities	$\frac{C}{A}$
3	910	132	778	0.85
5	498	99	399	0.80
7	331	65	266	0.80
10	120	35	85	0.71
14	7	1	6	0.86

Table 5. Analysis of similarly behaved tourists for Test Run 2

Day	A. Behaviour	B. With tourists of	C. With tourists of	Ratio
	links	s. nationalities	diff. nationalities	$\frac{C}{A}$
3	9,241	1,351	7,890	0.85
5	5,798	841	4,957	0.85
7	3,863	607	3,256	0.84
10	1,232	216	1,016	0.82
14	85	12	73	0.86

Table 6. Analysis of behaviour-connected tourists for Test Run 1

Day	A. One link	B. Between 2 and 4 links	C. Between 5 and 9 links	D. Above 10 links
3	32%	36%	18%	14%
5	41%	39%	16%	4%
7	50%	37%	13%	0%
10	57%	38%	5%	0%
14	100%	0%	0%	0%

Table 7. Analysis of behaviour-connected tourists for Test Run 2

Day	A. One link	B. Between 2 and 4 links	C. Between 5 and 9 links	D. Above 10 links
3	10%	20%	15%	55%
5	15%	21%	17%	47%
7	18%	23%	18%	41%
10	22%	31%	19%	28%
14	51%	41%	8%	0%

8 Conclusion

This article presents a software designed to artificially mimic the spending behaviour of tourists in a virtual city. It also describes a different behavioural model from the ones traditionally found in the literature, as it is targeted for decision and policy-making rather than sociometric analysis. Our main hypothesis was that usual market segmentation performed for the tourism industry, chiefly using the tourists' nationalities as discriminant, was not accurate. This has been proven thanks to both including stochastic processes in the multi-agent simulation and our expenditure-based segment model. The elements uncovered in the results exploration also support our introductory claim that if a benevolent party had access to the discussed data and would provide it for free to the economically active stakeholders, the leveraged knowledge would help increasing their marketing segmentation, enhancing the general destination's offer, hence contributing to a global *Pareto improvement*. However, this tool is only a first step toward more research investigations. On an marketing research aspect, it would still be desirable to know whether:

- a. statistical criteria other than the mean average absolute deviation and the Pearson product-moment correlation coefficients should be used to qualify a behaviour group,

- b. the same preoccupation may apply for explaining the “group-leaders” phenomenon,
- c. widening the simulation to include tourists with existing *i.e. non-zero* affinity factors **and** larger geographical space than a single city would trigger them to travel to find their likings,
- d. on the contrary, the same tourists with limited abilities to travel would see their existing behaviour ”change” to *accept* the local offer.

Answering these questions would lead, in our eyes using the presented behaviour model as a base for explaining the resulting segmentation with the traditional, qualitative models we reviewed in section 2. In computer science, finding patterns of knowledge in large data volumes can be achieved using data mining techniques. Mining tourism data still being a relatively new task, as mentioned in [35], the common issue of “feature selection”, *i.e. selecting relevant variables from the whole available set in order to optimise the results of the mining algorithm*, will certainly be encountered. Another problem lies in the integration of domain expertise in the knowledge discovery task, as discussed in [11]. In future work, we will present a step-by-step framework for the market segmentation of the tourism industry based on:

1. a swarm intelligence algorithm such as the *harmony search* [66] or the more recent *charged system search* [30], used in conjunction with domain knowledge features such as econometrics, for automating the feature selection process (*inspired from* [6]),
2. a clustering algorithm for discovering homogeneous market segment, *i.e. homogeneous behaviour groups*,
3. a supervised classification algorithm to explain the formation of such groups **and** their leaders.

References

1. Achrol, R.: Evolution of the marketing organization: new forms for turbulent environments. *The Journal of Marketing*, 77–93 (1991)
2. Achrol, R., Kotler, P.: Marketing in the network economy. *The Journal of Marketing*, 146–163 (1999)
3. Ajzen, I.: The Theory of Planned Behavior. *Organizational Behavior and Human Decision Processes* 50(2), 179–211 (1991)
4. Ajzen, I.: Perceived behavioral control, self-efficacy, locus of control, and the theory of planned behavior. *Journal of Applied Social Psychology* 32(4), 665–683 (2002)
5. Akerlof, G.A.: The market for “lemons”: Quality uncertainty and the market mechanism. *The Quarterly Journal of Economics* 84(3), 488–500 (1970)
6. Alexandre, E., Cuadra, L., Gil-Pita, R.: Sound Classification in Hearing Aids by the Harmony Search Algorithm. In: Geem, Z.W. (ed.) *Music-Inspired Harmony Search Algorithm*. SCI, vol. 191, pp. 173–188. Springer, Heidelberg (2009)
7. Arndt, J.: On making marketing science more scientific: role of orientations, paradigms, metaphors, and puzzle solving. *The Journal of Marketing* 49(3), 11–23 (1985)

8. Berry, L., et al.: Relationship marketing. Emerging Perspectives on Services Marketing 66(3), 33–47 (1983)
9. Bloom, J.: Tourist market segmentation with linear and non-linear techniques. Tourism Management 25, 723–733 (2005)
10. Bonn, M., Furr, H., Susskind, A.: Using the internet as a pleasure travel planning tool: An examination of the sociodemographic and behavioral characteristics among internet users and non-users. Journal of Hospitality & Tourism Research 22, 303–317 (1998)
11. Brisson, L., Collard, M.: How to Semantically Enhance a Data Mining Process? In: Filipe, J., Cordeiro, J. (eds.) Enterprise Information Systems. LNBP, vol. 19, pp. 103–116. Springer, Heidelberg (2009)
12. Cavarero, J.L.: Lapage: un modèle et un outil d'aide à la conception de systèmes d'information. Ph.D. thesis, Université de Nice-Sophia Antipolis (1979)
13. Central Intelligence Agency: The world fact (August 2011),
<https://www.cia.gov/library/publications/the-world-factbook/geos/xx.html>
14. Chattoe-Brown, E.: Just how (un)realistic are evolutionary algorithms as representations of social processes? Journal of Artificial Societies and Social Simulation 1 (1998)
15. Chiou, J.: Antecedents and moderators of behavioral intention: differences between us and taiwanese students. Genetic, Social, and General Psychology Monographs 126(1), 105–204 (2000)
16. Cooper, C., Fletcher, J., Gilbert, D., Wanhill, S.: Tourism: Principles and practice. Pitman Publishing (1993)
17. Cournot, A.: Recherches sur les principes mathématiques de la théorie des richesses (1838)
18. Davis, J., Eisenhardt, K., Bingham, C.B.: Developing theory through simulation methods. Academy of Management Review 32(2), 480–499 (2007)
19. Day, G.: Managing market relationships. Journal of the Academy of Marketing Science 28(1), 24 (2000)
20. Day, G., Deighton, J., Narayandas, D., Gummesson, E., Hunt, S., Prahalad, C., Rust, R., Shugan, S.: Invited commentaries on “evolving to a new dominant logic for marketing”. Journal of Marketing 68(1), 18–27 (2004)
21. De Cannière, M.H., De Pelsmacker, P., Geuens, M.: Relationship quality and the theory of planned behavior models of behavioral intentions and purchase behavior. Journal of Business Research 62(1), 82–92 (2009),
<http://www.sciencedirect.com/science/article/B6V7S-4RS9SP4-1/2/e34703c37f6fdb5db53f1cc0526658a3>
22. Dwyer, F., Schurr, P., Oh, S.: Developing buyer-seller relationships. The Journal of Marketing 51(2), 11–27 (1987)
23. Fishbein, M., Ajzen, I.: Belief, attitude, intention, and behavior: an introduction to theory and research. Addison-Wesley Pub. Co., Reading (1975)
24. Foxall, G.: Understanding consumer choice. Palgrave Macmillan (2005)
25. Gerbaux, F., Marcelpoil, E.: Governance of mountain resorts in france: the nature of the public-private partnership. Revue de Géographie Alpine 94(1), 20–31 (2006),
http://www.persee.fr/web/revues/home/prescript/article/riga_0035-1121_2006_num_94_1_2381
26. Giaglis, G., Klein, S., O’Keefe, R.: The role of intermediaries in electronic marketplaces: developing a contingency model. Information Systems Journal 12(3), 231–246 (2002)

27. Goodall: How tourists choose their holidays: An analytical framework. In: *Marketing in the Tourism Industry: The Promotion of Destination Regions*. Groom Helm (1988)
28. Hammond, J.: Four characterizations of constrained pareto efficiency in continuum economies with widespread externalities. *Japanese Economic Review* 46(2), 103–124 (1995)
29. Hunt, S., Day, G., Deighton, J., Narayandas, D., Gummesson, E., Prahalad, C., Rust, R., Shugan, S.: Invited commentaries on “evolving to a new dominant logic for marketing”. *Journal of Marketing* 68(1), 18–27 (2004)
30. Kaveh, A., Talatahari, S.: A novel heuristic optimization method: charged system search. *Acta Mechanica* 213, 267–289 (2010),
<http://dx.doi.org/10.1007/s00707-009-0270-4>,
doi:10.1007/s00707-009-0270-4
31. Kotler, P.: A generic concept of marketing. *The Journal of Marketing*, 46–54 (1972)
32. Kotler, P.: *Marketing Management*, 10th edn. Prentice-Hall Inc. (2000)
33. Kuhn, T.: *The structure of scientific revolutions*. University of Chicago Press, Chicago (1962)
34. Cao, L., Gorodetsky, V., Mitkas, P.: Agent mining: The synergy of agents and data mining. *IEEE Intelligent Systems* 24(3), 64–72 (2009)
35. Law, R., Mok, H.M.K., Goh, C.: Data Mining in Tourism Demand Analysis: A Retrospective Analysis. In: Alhajj, R., Gao, H., Li, X., Li, J., Zaïane, O.R. (eds.) *ADMA 2007. LNCS (LNAI)*, vol. 4632, pp. 508–515. Springer, Heidelberg (2007)
36. Likert, R.: A technique for the measurement of attitudes. *Archives of Psychology* 22(140), 1–55 (1932)
37. Lovelock, C., Gummesson, E.: Whither services marketing? *Journal of Service Research* 7(1), 20 (2004)
38. Luo, M., Feng, R., Cai, L.A.: Information search behavior and tourist characteristics: The internet vis-à-vis other information sources. *Journal of Travel & Tourism Marketing* 17, 15–25 (2004)
39. Lusch, R., Vargo, S.: Service-dominant logic: reactions, reflections and refinements. *Marketing Theory* 6(3), 281 (2006)
40. Mansfeld, Y.: From motivation to actual travel. *Annals of Tourism Research* 19, 399–419 (1992)
41. Marsaglia, G.: DIEHARD: a battery of tests of randomness (1996),
<http://stat.fsu.edu/~geo/diehard.html>
42. Maslow, A.H.: *Motivation and personality*. Harper & Row, London (1987)
43. Matsumoto, M., Nishimura, T.: Mersenne twister: a 623-dimensionally equidistributed uniform pseudo-random number generator. *ACM Trans. Model. Comput. Simul.* 8, 3–30 (1998), <http://doi.acm.org/10.1145/272991.272995>
44. Morgan, R., Hunt, S.: The commitment-trust theory of relationship marketing. *The Journal of Marketing* 58(3), 20–38 (1994)
45. Moutinho, L.: Trends in tourism. In: *Strategic Management in Tourism*. CABI Publishing (2000)
46. Nicholson, W.: *Microeconomic Theory: Basic Principles and Extensions*, 9th edn. South-Western College Pub. (April 2005)
47. Oliver, R.: Whence consumer loyalty? *The Journal of Marketing* 63, 33–44 (1999)
48. Palmer, A.: Relationship marketing: a universal paradigm or management fad? *The Learning Organization* 3(3), 18–25 (1996)
49. Porter, M.: *The competitive advantage of nations*. MacMillan, London (1991)
50. Rahim, M.: *Managing conflict in organizations*. Transaction Pub. (2010)

51. Reichheld, F., Teal, T.: The Loyalty Effect, the Hidden Force Behind Growth, Profits and Lasting Value. Harvard Business School Press, Cambridge (1996)
52. Reinartz, W., Kumar, V.: On the profitability of long-life customers in a noncontractual setting: An empirical investigation and implications for marketing. *The Journal of Marketing*, 17–35 (2000)
53. Sheth, J., Parvatiyar, A.: The evolution of relationship marketing. *International Business Review* 4(4), 397–418 (1995)
54. Shostack, G.: Breaking free from product marketing. *The Journal of Marketing*, 73–80 (1977)
55. Smith, A.: An Inquiry into the Nature and Causes of the Wealth of Nations (1776)
56. Solomon, M.: Consumer Behaviour. Prentice-Hall (1996)
57. Spence, A.M.: Market signaling: informational transfer in hiring and related screening processes. Harvard University Press, Cambridge (1974)
58. Stiglitz, J., Greenwald, B.: Externalities in economies with imperfect information and incomplete markets. *The Quarterly Journal of Economics* 101(2), 229–264 (1986)
59. Stiglitz, J.E.: Pareto efficient and optimal taxation and the new new welfare economics. National Bureau of Economic Research Working Paper Series No. 2189 (December 1988), <http://www.nber.org/papers/w2189>
60. Swarbrooke, J., Horner, S.: Consumer behaviour in tourism. Butterworth-Heinemann, Oxford (1999)
61. Vargo, S., Lusch, R.: Evolving to a new dominant logic for marketing. *Journal of Marketing*, 1–17 (2004)
62. Vargo, S., Lusch, R.: The four service marketing myths. *Journal of Service Research* 6(4), 324 (2004)
63. Venugopal, V., Baets, W.: Neural networks and statistical techniques in marketing research: A conceptual comparison. *Marketing Intelligence and Planning* 12, 30–38 (1994)
64. Wang, Y., Fesenmaier, D.R.: Towards understanding members' general participation in and active contribution to an online travel community. *Tourism Management* 25, 709–722 (2004)
65. Webster, F.: The changing role of marketing in the corporation. *The Journal of Marketing*, 1–17 (1992)
66. Zong, W.G.: Music-Inspired Harmony Search Algorithm Theory and Applications. SCI. Springer, Heidelberg (2009)
67. Zott, C.: Dynamic capabilities and the emergence of intraindustry differential firm performance: insights from a simulation study. *Strategic Management Journal* 24(2), 97–125 (2003)

A Comparative Study of a Financial Agent Based Simulator Across Learning Scenarios

Filippo Neri

Dept. of Computer Science, University of Naples, 80100 Naples, Italy
filipponeri@yahoo.com

Abstract. Integrating agent based modeling with learning results in a promising methodology to model the behavior of financial markets. We discuss here how partial and full knowledge learning setups can be combined with agent based modeling to approximate the behavior of financial time series. Partial knowledge learners operate with limited knowledge of the domain, usually only the initial conditions are used. While full knowledge learners use any domain data any time it is made available to adjust their predictions.

We report in this paper an experimental study of our learning system L-FABS, introduced in previous works, in order to show how it can discover models for approximating time series working in partial knowledge and full knowledge learning scenarios.

Keywords: Agent based modeling, simulated annealing, financial markets, prediction of the SP500 and DJIA time series.

1 Introduction

Financial markets can be viewed as complex systems whose basic entities and interactions can be easily described. However no theory or method is able to explain or predict with certainty what will happen during their future evolution. The state-of-the-art literature shows that agent based modeling (ABM) is a promising methodology for simulating several types of domains, when interpreted as complex systems, like, for instance, consumer markets, economies or societies [9,16,2,4,5,10,25,6]. Thus ABM could be a promising candidate for investigating the behavior of financial markets as well. Current research in ABM and in modeling financial markets usually focuses on: evaluating or learning trading strategies for financial instruments (commodities, bonds, shares, etc.) [13,23,8]; or developing artificial markets, whose internal dynamics can be controlled, in order to study the formation of notable phenomena (price bubble, for instance) [24,1,12]; or, finally, modeling the time series of the values/prices for financial assets [19,20,15]. The cited papers have been selected with the only intent to provide examples of the listed approaches and without claim to be an exhaustive list. For researchers active in the ABM community, computational simulation takes the form of agent based simulators where hypothesis about the decision making process of the individual and about the relationships occurring

among them could be tested [3,9,17,25,12]. One of the main difficulties encountered by these works stays in tuning the model so that the simulated behavior approximate the observed one. With this respect machine learning algorithms, and simulated annealing [14] in the case of our research, can provide a useful tool to learn the main parameters governing the simulation process [19,20].

In this paper, we report our empirically investigation of the Learning Financial Agent Based Simulator L-FABS, that has been introduced in [19,20], showing that it can approximate the time series of the SP500 and DJIA indexes under two experimental setups. The novelties of this paper are:

- 1) the definition of the partial knowledge learning scenario and the full knowledge one, we will refer to them as PK and FK, respectively, thereafter for conciseness;
- 2) the description of how L-FABS can operates under the PK and FK learning frameworks;
- 3) the study of the behavior of L-FABS over different time series to evaluate its performances under several experimental settings.

The rest of the paper is organized as follows: in Section 2, we describe the main economic hypothesis that L-FABS is based on. In Section 3 we report the architecture of the main component of our system: the Financial Agent Simulator FAS, essentially this is our system without the learning capability. Following, in Section 4 and Section 5, we discuss how to measure the approximation error between two time series and how Simulated Annealing is used to add a learning capability to FAS in order to create the complete system L-FBAS. In Section 6, we discuss the differences between PK and FK learning and, in Section 7, our experimental results are reported.

2 Economic Hypothesis Made in Designing L-FABS

The abstraction of financial markets that we used to design L-FABS [19,20] takes into account that the behavior of the investors is driven by two main factors:
a) the propensity to take some investment risks today, by buying a financial asset, in exchange for a future uncertain reward, when selling the asset; and
b) the common consensus (the market sentiment or just sentiment) about the future behavior of the market itself. If the people believe that the economic outlook will be negative, each individual will tend to sell some of her/his assets.

3 The Financial Agent Simulator

Our system L-FABS is composed of two sub-systems: the Financial Agent Simulator, FAS for short, and the learning component, Simulated Annealing in our case. As L-FABS has been introduced in our previous works [19,20], we just recall here the main elements of its architecture that are useful for understanding the reported study. In this section we describe the FAS sub-system that is responsible for the simulation of the set of interacting investors thus realizing the market simulator. During each investment round, an investor in FAS, will behave accordingly to this stochastic procedure:

- buy some assets with probability $P(X < \text{BuyThreshold})$,
- sell some of its assets with probability $P(\text{SellThreshold} < X)$, and
- do nothing with probability $P(\text{BuyThreshold} < X < \text{SellThreshold})$.

The probabilities are dependent on both the current estimation of the outlook (Sentiment) for the economy, which is the same for all the agents, and his *risk/reward propensity rate*, which varies according to the type of investor (bank, hedge funds, central banks, and so on).

The simulation of the whole financial market can be obtained by creating several agents, defined as above, each one with its own status in terms of hold assets, invested assets and risk/reward propensity, and then performing a sequence of investment rounds where each agent decides if buying, selling, or holding taking into account the current Sentiment value. At the end of each round, it is possible to measure the percentage of invested assets, which act as a proxy to estimate one data point of the target time series. If the simulation is repeated for a number of rounds, FAS will output a estimate for the full time series. The template for FAS is shown in Table 1. FAS takes as input the

Table 1. The FAS algorithm template

```

FAS(RiskRewardRates)
round=1
maxrounds=DaysToSimulate()
numDays=numDaysF()
numAgents=NumOfAgents()
initialise(numAgents,RiskRewardRates)
while round < maxrounds do
    Sentiment = Sentiment(round,numDays)
    TotMarketAssets=0, TotInvAssets=0
    for i=1 to numAgents do
        FinancialAgent(i, Sentiment)
        TotMarketAssets += TotalAssetsF(i)
        TotInvAssets += InvestedAssetsF(i)
    endfor
    EstValue = TotInvAssets / TotMarketAssets
    round = round + 1
    Append(PredictedData, EstValue)
endwhile
```

vector of risk/rewards propensity rates for each agent. During each round, the risk/rewards propensity rates are used in combination with the current value of the economic sentiment by each agent to take an investment decision. We also note that our approach does not impose any restriction on how the market mood is determined. Let us define the MAVG function as:

$$\text{MAVG(index,t,n)} = \sum_{k=0}^{n-1} \text{index}(t-k)/n$$

According to our choice for the Sentiment function [19,20], the outlook of the real market is considered bullish if the moving average (MAVG) of the *predicted* time series is lower than the moving average of the *real* time series. If this is the case, the Sentiment value is set to an high value so that a bullish mood is communicated to the agents. The opposite happens if the predictions of the system have been higher than the real data. For short, in the following, S_n will identify the Sentiment function called as $\text{Sentiment}(\text{currentTime}, n)$. Finally, we recall that in our study we have defined four types or classes of agents to capture the richness in investment decisions and in size of financial transactions that occur in real financial markets: individual investors (and the likes), banks (and the likes), hedge funds (and the likes), and central banks (and the likes). They differ in term of the size of the assets they can invest in financial markets and their numerical presence. Please refer to [19,20] for details.

As the reader may have certainly noted, we did not specify the risk/reward propensity rates to be given as input to FAS. The reason is that those values are learned by L-FABS from the time series under study.

4 Measuring the Approximation Error between Two Time Series

In order to use a learning algorithm to find a model for a time series, we need to select some error function to be minimized during learning that is able to evaluate how well two time series (the estimated and the target one) are similar. The Mean Squared Error and the Mean Average Percentage Error fulfill our requirements and are commonly used in Statistics when two data samplings have to be compared. They are defined as:

$$MSE(X, Y) = \frac{1}{N} \sum_{i=1}^N (x_i - y_i)^2;$$

$$MAPE(X, Y) = \frac{1}{N} \sum_{i=1}^N \left| \frac{x_i - y_i}{x_i} \right|$$

Given two time series X and Y, the lower the values for MSE and MAPE, the closer the two are. MSE provide an absolute measure for the error while MAPE returns the error measured in percentage terms. The squared root of MSE is known as the Standard Error. The MSE or Standard Error are useful error measures when comparing the approximation error between several estimated time series for the same target one: as the time series have the same data range, a measure of error in absolute terms makes sense. Instead the MAPE is more suitable when comparing the approximation error for time series having different data ranges. In this case, an error expressed as a percentage is more suitable in evaluating their degree of similarity.

5 L-FBAS: Combining Simulated Annealing and FAS

In this section, we describe how a learning capability can be added to FAS so that it may find the best model for a given time series. The resulting system will be called L-FBAS. Learning in L-FBAS consists of finding the vector of risk/reward propensity rates that approximates a given time series with a minimum error. This learning setting allows for combining FAS, the simulation engine, with any of the many machine learning algorithms able to find a vector of values that minimizes a given error function. Examples of suitable machine learning algorithms include genetic algorithms [11,18], decision trees [21], neural networks [22], simulated annealing [14] and many others. In our study, we decided to use simulated annealing because evolutionary algorithms proved to be robust and well performing across several domains [11,14,18] and an individual oriented method is less computationally expensive than a population oriented one.

Simulated annealing [14] is a searching algorithm that iteratively searches for the minimum of a given function E (the Energy or Error function) by generating at each step random candidate points for the minimum in the proximity of the current candidate minimum point which represents the "center" of the exploration. At the end of each step, Simulated Annealing may move the center of its exploration at a newly generated point with a given probability. Running L-FBAS consists of running Simulated Annealing to search the vector space of the risk/reward propensity rates in order to find one that minimizes the error function. At each round of annealing, the candidate vector, s , is used as input to FAS. FAS then returns an estimated time series to approximate the given $targetTimeSeries$. The approximation error measured by MAPE (MSE) is used to decide if the candidate vector is an improved solution. The template for L-FBAS is shown in Table 2.

Table 2. The L-FBAS algorithm using Simulated Annealing for learning

```
L-FBAS(targetTimeSeries)
  s = randomstate() // s is a vector of risk/reward propensity rates
  e = MAPE(FAS(s), targetTimeSeries)
  T = 1; α = 0.95
  k = 0; kmax = 100; emax = 0
  while(k < kmax)and(e > emax)do
    sn = neighbour(s)
    en = MAPE(FAS(sn), targetTimeSeries)
    if(en < e)then sbest = sn; ebest = en
    endif
    if(random(1) < P(en, e, T))then s = sn; e = en
    endif
    k = k + 1; T = T × α
  endwhile
  return(beststate)
```

The *neighbor(s)* function returns a state in proximity of its argument. We implemented it as a function that for each value of the vector s randomly adds or decreases it by $0.1 \times \text{random}(1)$. $P(e_n, e, T)$ is returns 1, if the approximation error of the new state, e_n , is lower than the approximation error of the current state, e ; else it returns $\exp(\frac{e - e_n}{T})$. This means that even if the new state has an higher approximation error than the current state, there is a non zero probability that the L-FABS will move its exploration center to it. The parameters α and T are set to 0.95 and to 1 according to [14].

6 Partial Knowledge Learning vs Full Knowledge Learning

Before describing the experimental study, we want to explain the difference and between learning scenarios where partial or complete knowledge of the data in the time series is available because this information will help understanding the selected experimental setups. The difference between partial knowledge learning and full knowledge can also be related to learning when no domain knowledge is available, apart from the data, or when a domain theory, maybe even a causal model of the domain is available.

In our study, we consider the effect of having access or not having access to the real values of a time series from time 0 up to the current time t when forecasting its future values from time $t+1$ and ahead. In general, when making a prediction, all learning systems try to exploit every bit of information in order to reduce their forecasting error. This is a common and correct approach when employing a machine learning system. However, it can also be noted that, when modeling a time series, the use of all the information up to the current time t is generally a necessity to obtain an acceptable forecasting error as the majority of the learning systems base their predictions on the identification of patterns in the recent part of the target time series [7]. In summary, pattern learning methods operates under a full knowledge learning scenarios.

We will call *full knowledge* scenario a learning set up where all the target time series data points up to the current time are available to the learner, while we identify with *partial knowledge* scenario the learning set up where only the initial data point of the target time series is available to the learner. Most machine learning systems operate through pattern matching and are then able to work only in a full knowledge learning scenario. Instead L-FABS bases its predictions on the simulation of the investment process that underlies the market. L-FABS can either take advantage of all the data in the time series or it can just use the initial conditions of the market, data at time zero. So L-FABS can operates both in a full knowledge or partial knowledge learning scenario.

The above considerations then open up the possibility to experiment with two learning scenarios with when studying L-FABS:

FullKnowledge(FK) - all the information in the time series, up to the current time, is accessible during learning. This implies that, at the beginning of each simulation round and if needed, the correct value of the time series at time t-1 is set as the value of the estimated time series at time t-1.

PartialKnowledge(PK) - only the starting point of the time series, t=0, is given to L-FABS in order to initialize the simulation, but all the rest of the real time series values are not known to the L-FABS. In this case, the simulation model in L-FABS will move from one predicted value for the time series to the next without using the correct value of the time series at time t-1 in order to estimate a value for time t.

Also as learning scenario Partial Knowledge exploits less information than the Full Knowledge one, it corresponds to a more difficult learning task. We are considering the Partial Knowledge scenario because this learning setting can provide insights about the "robustness" of the model acquired by L-FABS. By robustness we mean the feature of the model to have intrinsically captured the law of evolution in time of the time series which can be measured by the maintenance of accuracy in forecasts as the system move far from the last known point of origin.

7 Empirical Analysis

For the empirical evaluation of L-FABS, we selected some financial time series to work with. As usual with learning systems, we will train L-FABS on a part of the dataset, the learning set, and then we will use the remaining part of the dataset as test set to assess the performances of the learned model. The selected datasets are:

Dataset 1 - learning set: SP500 from 3 Jan 1994 to 17 Dec 2003 and test set: SP500 from 18 Dec 2003 to 23 Oct 2006.

Dataset 2 - learning set: DJIA from 3 Jan 1994 to 17 Dec 2003 and test set: DJIA from 18 Dec 2003 to 23 Oct 2006.

All the datasets contain the daily closing values of the indexes and have been freely acquired from the finance section of yahoo.com.

In the performed experiments, the Sentiment value will be calculated using *Sentiment(round,1)*, case identified with S1, or calling *Sentiment(round,5)*, case identified with S5. We recall that in case S1, only the previous day value of the real index is used, while in case S5, the average of the latest previous five days is used, as explained earlier. In term of information available to S-FABS, we will explore both the *FullKnowledge* learning scenario, the information about the correct value at time t is used when learning or making a prediction for t+1, and the *LimitedKnowledge* setting, only information about the the first point in the time series is used to learn or make a prediction.

Finally, we will evaluate S-FABS when estimating the next value of the time series (the value of the next trading day) and the seven days ahead value of the time series.

The following reported results are averaged over 10 runs of the same experimental setting and the shown forecast errors are measured on test sets. In Table 3, the performances of L-FABS are shown when run on the Datasets 1 and 2.

Another finding from the experiments is that the error rates obtained in the learning scenarios *PartialKnowledge* and *FullKnowledge* on Datasets 1 and 2 tend to be very close. A possible explanation for these observations is that, during the period of time represented in Dataset 1 and 2, the SP500 and DJIA time series behaved in trend with the latest days moving average (S1 or S5) so the Sentiment value contains enough information for L-FABS to make good forecasts. Thus the superiority of the learning scenario *FullKnowledge* over the *PartialKnowledge* one.

With a special attention to the latest two empirical findings, we would like to suggest the hypothesis that the ability of L-FABS to learn a deep model of the time series under a partial or full knowledge scenario might be useful in dealing with financial time series having high volatility.

Table 3. Experimental findings on Datasets 1 and 2 relative to different periods of the SP500 and DJIA time series

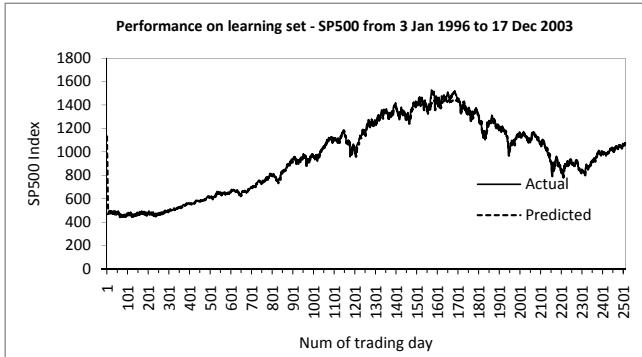
Experimental results on Dataset 1

Knowledge	Day to predict	Sentiment	MAPE	StdErr
PK	1	S1	0.70	14.65
PK	1	S5	1.06	22.26
PK	7	S1	1.46	25.75
PK	7	S5	1.65	33.01
FK	1	S1	0.76	16.55
FK	1	S5	0.70	14.31
FK	7	S1	1.46	25.16
FK	7	S5	1.42	24.29

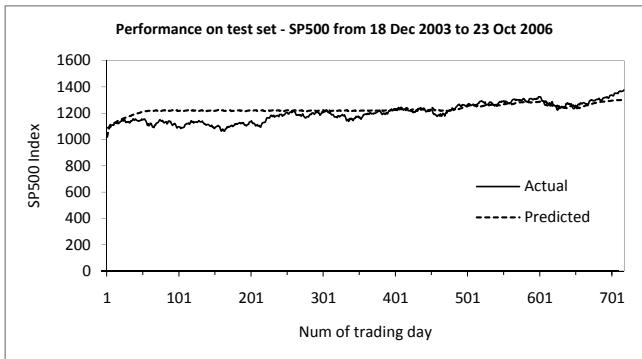
Experimental results on Dataset 2

Knowledge	Day to predict	Sentiment	MAPE	StdErr
PK	1	S1	0.70	114.1
PK	1	S5	1.16	203.54
PK	7	S1	1.38	214.34
PK	7	S5	1.72	290.51
FK	1	S1	0.76	136.52
FK	1	S5	0.74	131.73
FK	7	S1	1.48	215.26
FK	7	S5	1.50	221.83

For the sake of completeness, we also show the graphs of two time series as predicted by L-FABS in fig. 1 and fig. 2, which are exemplars of the results



(a) SP500 time series during learning phase on Dataset 1 with parameters: PartialKnowledge, S1, Day to predict:1.



(b) SP500 time series when testing L-FABS on Dataset 1 with parameters: PartialKnowledge, S1, Day to predict: 1.

Fig. 1. Comparison of the actual and predicted time series obtained by L-FABS under the experimental settings appearing in the captions of the graphs

obtained over several runs of L-FABS. The predicted time series are compared with the actual ones. As it can be seen from the graphs, the solid line (actual) and the dotted line (predicted) are very close confirming the error figures that have been reported in the tables.

The columns in the table stand for: "Knowledge" identifies the learning setting as with *FullKnowledge* or *LimitedKnowledge*, "Day to predict" indicates the number of days ahead for which a prediction of the time series is made, "Sentiment" indicates if the Sentiment index is calculated with modality S1 or S5, and, finally, the measured errors on the test set are reported in terms of the MAPE and Standard Error. The results in Table 3 show lower errors both in terms of the MAPE and of the Standard Error when the *FullKnowledge* learning setting is selected instead of the *PartialKnowledge* one. This empirical finding suggests that using all the information available in the time series

up to the current time, before making a prediction, produces better forecasts. However the performances obtained under the *PartialKnowledge* learning set up are quite good considering that the system would only know about the starting point of the time series. This finding suggests that the model learned by L-FABS has captured the intrinsic dynamics of the target time series. Moreover, from Table 3, it appears that the predictions of the next day values are more accurate than the predictions made for the seven days ahead values. This result confirms the intuitive experience that the farther a prediction is moved into the future, the less accurate it will be.

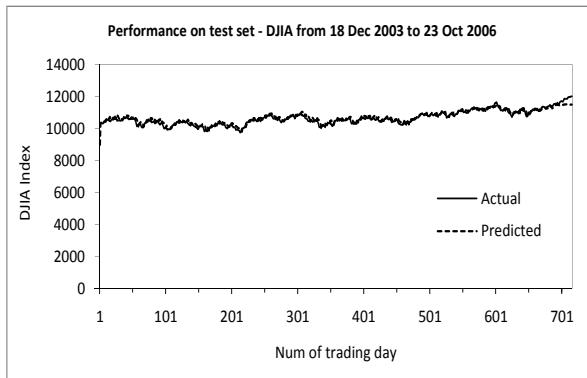


Fig. 2. The actual and predicted DJIA time series when testing L-FABS on Dataset 2 with settings: Full Knowledge, S1, Day to predict: 1

8 Conclusions

We have reported a study about learning agent based simulations to model financial time series by investigating the behavior of our system L-FABS when operating under a partial knowledge learning scenario and a full knowledge learning one. In L-FABS, a set of agents simulated the interacting individual investors that can be recognized in financial markets, while learning is used to discover the simulation parameters that best suit a target financial time series. The main results reported in this work are:

- 1) the description of a partial knowledge learning scenario and of the full knowledge one;
- 2) the description of how the architecture of L-FABS allows the systems to work under the PK and FK learning frameworks;
- 3) the analysis of the behavior of L-FABS over different time series under several experimental settings.

Finally, we believe that our findings support the working hypothesis that agent based simulation together with learning algorithms could result in a methodology fit to model financial time series.

References

1. Arthur, W.B., Holland, J.H., LeBaron, B., Palmer, R., Taylorm, P.: Asset pricing under endogenous expectation in an artificial stock market. In: *The Economy as an Evolving Complex System II*. pp. 15–44. Santa Fe Institute Studies in the Sciences of Complexity Lecture Notes (1997)
2. Axelrod, R.: Agent-based modeling as a bridge between disciplines. In: Judd, K.L., Tesfatsion, L. (eds.) *Handbook of Computational Economics. Agent-Based Computational Economics*, Handbooks in Economics Series, vol. 2, pp. 1562–1583. North-Holland, Amsterdam (2006)
3. Bonabeau, E.: Agent-based modeling: Methods and techniques for simulating human systems. *Proceedings of the National Academy of Sciences* 99(3), 7280–7287 (2002)
4. Bruun, C.: The economy as an agent-based whole simulating schumpeterian dynamics. *Industry and Innovation* 10 (2003)
5. Canzian, G.: Three Essays in Agent-Based Macroeconomics. PhD thesis, University of Trento (2009)
6. Cao, L., Feng, Y., Zhong, J. (eds.): *ADMA 2010, Part I. LNCS*, vol. 6440. Springer, Heidelberg (2010)
7. Creamer, G., Freund, Y.: Automated trading with boosting and expert weighting. *Quantitative Finance* 4(10), 401–420 (2010)
8. Dempster, M.A.H., Payne, T.W., Romahi, Y., Thompson, G.W.P.: Computational learning techniques for intraday fx trading using popular technical indicators. *IEEE Transactions on Neural Networks* 12(4), 744–754 (2001)
9. Epstein, J.M., Axtell, R.: Growing artificial societies: social science from the bottom up. The Brookings Institution, Washington, DC, USA (1996)
10. Fagiolo, G., Dosi, G.: Exploitation, exploration and innovation in a model of endogenous growth with locally interacting agents. Lem papers series, Laboratory of Economics and Management (LEM), Sant'Anna School of Advanced Studies, Pisa, Italy (2002)
11. Goldberg, D.: *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley, Reading (1989)
12. Hoffmann, A.O.I., Delre, S.A., von Eije, J.H., Jager, W.: Artificial multi-agent stock markets: Simple strategies, complex outcomes. In: *Advances in Artificial Economics. LNEMS*, vol. 584, pp. 167–176. Springer, Heidelberg (2006)
13. Kendall, G., Su, Y.: A multi-agent based simulated stock market - testing on different types of stocks. In: *Congress on Evolutionary Computation, CEC 2003*, pp. 2298–2305 (2003)
14. Kirkpatrick, S., Gelatt, C.D., Vecchi, M.P.: Optimization by simulated annealing. *Science* 220, 671–680 (1983)
15. Kitov, I.: Predicting conocophillips and exxon mobil stock price. *Journal of Applied Research in Finance* 2, 129–134 (2009)
16. Cao, L., Gorodetsky, V., Mitkas, P.: Agent mining: The synergy of agents and data mining. *IEEE Intelligent Systems* 24(3), 64–72 (2009)
17. Lebaron, B.: Agent based computational finance: Suggested readings and early research. *Journal of Economic Dynamics and Control* 24, 679–702 (1998)
18. Neri, F.: Traffic packet based intrusion detection: decision trees and generic based learning evaluation. *WSEAS Transaction on Computers* 4(9), 1017–1024 (2005)
19. Neri, F.: How to investigate the decision making behavior of investors in financial markets by means of software agents. In: *12th WSEAS International Conference on Automatic Control, Modelling and Simulation, Catania, Italy*, pp. 118–124. Euromedia Press (2010)

20. Neri, F.: Learning and Predicting Financial Time Series by Combining Natural Computation and Agent Simulation. In: Di Chio, C., Brabazon, A., Di Caro, G.A., Drechsler, R., Farooq, M., Grahl, J., Greenfield, G., Prins, C., Romero, J., Squillero, G., Tarantino, E., Tettamanzi, A.G.B., Urquhart, N., Uyar, A.S. (eds.) *EvoApplications 2011, Part II*. LNCS, vol. 6625, pp. 111–119. Springer, Heidelberg (2011)
21. Quinlan, J.R.: *C4.5: Programs for Machine Learning*. Morgan Kaufmann, California (1993)
22. Rumelhart, D.E., Hinton, G.E., Williams, R.J.: Learning internal representations by error propagation. In: *Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Foundations*, vol. 1, pp. 318–362. MIT Press, Cambridge (1986)
23. Schulenburg, S., Ross, P.: An Adaptive Agent Based Economic Model. In: Lanzi, P.L., Stolzmann, W., Wilson, S.W. (eds.) *IWLCS 1999. LNCS (LNAI)*, vol. 1813, pp. 263–282. Springer, Heidelberg (2000)
24. Takahashi, H., Terano, T.: Analyzing the Influence of Overconfident Investors on Financial Markets Through Agent-Based Model. In: Yin, H., Tino, P., Corchado, E., Byrne, W., Yao, X. (eds.) *IDEAL 2007. LNCS*, vol. 4881, pp. 1042–1052. Springer, Heidelberg (2007)
25. Tesfatsion, L.: Agent-based computational economics: Growing economies from the bottom up. *Artif. Life* 8(1), 55–82 (2002)

Agent-Based Cluster Analysis of Tropical Cyclone Tracks in the Western North Pacific

Wei Zhang¹ and Yuanfei Wang²

¹ Department of Geography and Resource Management,

The Chinese University of Hong Kong,

Shatin, Hong Kong, China

zhangwei@cuhk.edu.hk

² Key lab of Geo-information Science,

Ministry of Education, East China Normal University,

Shanghai, China

yfwang@geo.ecnu.edu.cn

Abstract. The clustering model integrating Finite Mixture Model (FMM) and classical Expectation-Maximum (EM) algorithm has been applied to tropical cyclone (TC) tracks during the last decade. However, the efficiency of classical EM algorithm is insufficiently good and the robustness of the model is not verified. Besides, it is inconvenient for users to manually choose the parameters for the cluster analysis. In order to improve the efficiency of classical EM algorithm, the “Lazy- $\Psi\alpha_2$ ” EM is proposed by integrating Lazy EM algorithm and $\Psi\alpha_2$ algorithm. Sensitivity analysis is conducted to ensure the insensitivity of the clustering model to the amount of data set. The cluster analysis is implemented on an agent-based framework by which the tool can automatically choose the parameters by evaluating the clustering performance. TC tracks in western North Pacific from 1949 to 2006 are classified into 12 clusters by the probabilistic clustering model that is solved by “Lazy- $\Psi\alpha_2$ ” EM algorithm. The log-likelihood is taken as the performance indicator. Elaborate comparisons are made between the present cluster analysis and other cluster analyses related to TC tracks.

Keywords: Cluster analysis, Agent-based, Tropical Cyclone, Finite Mixture Model, EM algorithm.

1 Introduction

Tropical Cyclones (TCs) impose considerably destructive impacts on coastal societies each year around the world, particularly in the western North Pacific (WNP) and Atlantic basins. Developing the capabilities of forecasting the formation and movement of TCs and mining valuable information and pattern from current and historical TC archives is taken as one top research priority by both academics and governments of all levels [1, 25, 40]. TC movement and intensity change are two critical problems in TC-related studies of great scientific and operational concerns [40, 42].

Numerous researches have been conducted on the prediction of TC track and intensity and on forecasting TC frequency in a particular year, as well as on TC analog forecast. With the advancements of statistics and numerical weather prediction models, the TC track and intensity forecasting are increasingly accurate [11, 12, 16, 24]. The characteristics of TC tracks need further understanding so as to isolate potentially predictable aspects of TC movement or landfall [9]. Cluster analysis, as a typical data mining algorithm, has long been used to discover potential and hidden patterns or groups from historical TC tracks [9, 10, 28]. The quantitative characteristics of clusters in TC tracks can naturally provide valuable references for TC track prediction including landfall, recurvature and translational velocity. Therefore, clustering of TC tracks is of great concern to public and scientific communities due to its capabilities in unraveling the hidden and useful knowledge, information, and patterns from the historical TC database [9, 10, 26].

In case of the application of cluster analysis to TC trajectories, the K-means method [32] has been used to analyze TCs in WNP [18] and North Atlantic [17]. In these studies, the grouping was implemented according to the positions of maximum and final hurricane intensity (i.e. the last position at which a TC reached hurricane intensity). Three clusters were unraveled from the trajectories. The K-means approach has also been used to cluster North Atlantic extra-tropical cyclone trajectories, where 6-hourly latitude-longitude positions over 3 days were converted into 24-dimensional vectors suitable for clustering [4].

The K-means is a straightforward and widely-used partitioning method that seeks to assign each track to one of K groups such that the total variance among the groups is minimized. However, K-means cannot accommodate tracks of different lengths, and it is a serious shortcoming for TCs. Harr and Elsberry [27] used fuzzy cluster analysis and empirical orthogonal functions to describe the spatial patterns associated with different kinds of TC characteristics.

Continuous attention has been paid to overcome the shortcoming of K-means in grouping TCs with different lengths [7, 8, 9, 10, 21, 22, 23]. Gaffney [23] employed polynomial and spline regression models to simulate hurricane tracks in Atlantic basin. Finite Mixture Model (FMM) [35] is integrated to build the clustering model, and eventually Expectation-Maximization (EM) algorithm [15] is used to solve the model and estimate the parameters for the FMM. However, the parameters of cluster analysis including the number of clusters and number of validation times are required to be chosen manually by the users. The automatic selection processes on the clustering framework are of great importance for the users.

An agent is a computer system capable of flexible autonomous action in a dynamic, unpredictable and open environment. Agents offer an abstraction tool, or metaphor, for the design and construction of complex systems with multiple distinct and independent components [14, 31]. Multi-agent systems are computational systems in which several artificial “agents”, which are programs, interact or work together over a communication network to perform some set of tasks jointly or to satisfy some set of goals [20, 29, 43]. Multi-agent systems have been widely applied to a variety of areas [44] such as land use and land cover change simulation [37], traffic and transportation [2, 6, 41], and ecological system analysis [5]. Few investigations are conducted on integration of multi-agent framework with TC analysis. Therefore, the FMM-based clustering algorithm, implemented on an agent-based framework, can facilitate the

clustering process of TC tracks and provide helpful knowledge and references for the operational forecasters.

This paper is organized as follows. Section 2 gives an overview of cluster analysis model for TC tracks. Section 3 specifies the “Lazy- $\Psi\alpha 2$ ” algorithm, the sensitivity analysis for the TC tracks clustering model as well as the cluster analysis of TC tracks in WNP from 1949 to 2006. Concluding remarks are outlined in section 4.

2 Methodology

2.1 Clustering Model

The curve clustering algorithm is based on the algorithms that are used in [9, 23, 24]. This algorithm is derived from FMM [19, 33, 35], which is capable of modeling probability densities, especially non-Gaussian density, using a small set of basic probability component densities [9]. This algorithm employs mixed polynomial regression models (i.e., curves) to fit the geographical “shape” of the TC tracks and model a TC’s longitudinal and latitudinal positions versus time [24]. Through FMM, non-Gaussian density functions can be expressed as a mixture of uni-modal component probability distribution functions. The model is fitted to the data by maximizing the likelihood of the parameters, given the historical TC datasets.

At the beginning of building the clustering model, we use polynomial regression models to simulate the TC tracks.

$$y_i = X_i \beta + \varepsilon_i, \quad \varepsilon_i \sim N(0, \sigma^2 I), \quad (1)$$

where y_i is the curve of one TC track. Each curve has length n_i that means the number of observations, with measurements observed at the points in x_i . The $n_i \times p$ regression matrix is the Vandermonde matrix at x_i , and β is the p -vector of regression coefficients. The p -th order Vandermonde matrix evaluated at x_i is equal to

$$X_i = \begin{pmatrix} 1 & \cdots & x_{i1}^p \\ \vdots & \ddots & \vdots \\ 1 & \cdots & x_{in}^p \end{pmatrix}$$

In order to accommodate the spatial alignment in the regression model, the space-alignment, is integrated into the model. The space alignment is implemented by adding spatial transformation - d_i into the regression model. This d_i allows for the entire curve to be translated as a unit.

After d_i is integrated, the model is depicted as

$$y_i = X_i \beta + d_i + \varepsilon_i, \quad \varepsilon_i \sim N(0, \sigma^2 I), \quad d_i \sim N(0, v^2) \quad (2)$$

where y_i gives the i -th curve (i.e., TC track) of length n_i , and $N(0, \sigma^2 I)$ gives the Gaussian noise model, The $n_i \times p$ regression matrix is the Vandermonde matrix at x_i , and β is the p -vector of regression coefficients, v^2 is the error term of the transformation in the regression model.

“k” is utilized to express the conditional PDF $p_k(y_i | d_i)$ as

$$p_k(y_i | d_i) = N(X_i \beta + d_i, \sigma_k^2 I) \quad (3)$$

The conditional mixture density function is written as:

$$p_k(y_i | d_i) = \sum_k \alpha_k p_k(y_i | d_i) \quad (4)$$

As a result, the joint probability and marginal density are utilized to calculate the mixture density function of y_i as

$$p(y_i) = \sum_k \alpha_k p_k(y_i) \quad (5)$$

$$\begin{aligned} p_k(y_i) &= \int p_k(y_i | d_i) p_k(d_i) dd_i \\ &= \int p_k(y_i | d_i) p_k(d_i) dd_i \\ &= \int N(y_i | X_i \beta_k + d_i, \sigma_k^2 I) N(d_i | 0, v_k^2) dd_i \\ &= N(y_i | X_i \beta_k, v_k^2 I + \sigma_k^2 I) \end{aligned} \quad (6)$$

The FMM-based clustering model is solved via EM algorithm. The estimators of parameters $\{\beta, \sigma^2, v^2\}$ are:

$$\begin{aligned} v_k^2 &= 1/n \sum_i \hat{g}(d_i | k), \\ \sigma_k^2 &= 1/N \sum_i \hat{f}(d_{ik}), \\ \hat{\beta}_k &= [\sum_i X_i' X_i]^{-1} \sum_i X_i' (y_i - \hat{d}_{ik}) \end{aligned} \quad (7)$$

Other than using polynomial regression model to simulate the TC tracks and integrating spatial alignment, there also exist other ways of building clustering model such as by spline regression model [23]. Given the clustering algorithm proposed by Gaffney [23, 24], several questions can be proposed. How to automatically determine the parameters for cluster analysis such as k - the number of clusters and other parameters? Are the clustering results stable or robust? How about the efficiency of the clustering model?

The following methods provide ways of solving these problems; for example, the multi-agent framework is thus used to automatically select the proper parameters; sensitivity analysis will be employed to test the robustness of the clustering model; we propose a new algorithm for accelerating the classical EM algorithm.

2.2 Agent-Based Cluster Analysis Framework

Cluster analysis has a wide range of members (e.g., partitioning clustering, hierarchical clustering, density-based and probability-based clustering). Users are required to determine the number of clusters and other parameters prior to cluster analysis. With regard to the FMM-based clustering algorithm, we need to select a wide range of parameters such as number of clusters and number of polynomial order. After comparing the cluster analysis results by the indicators with respect to the selected parameters, the users manually determine the proper parameters for the cluster analysis. This process is considerably inconvenient for users, especially for potentially large clusters existing in the database. Therefore, intelligent agent that substitutes the analyzer to complete the tasks and that undertakes automatic clustering is required. Therefore, the agent-based cluster analysis in this study comprises a clustering agent and a parameter-chosen agent. Using the parameters determined by parameter-chosen agent, the clustering agent implements the cluster analysis and generates results including different clusters and measurements for clustering performance (see Fig. 1).

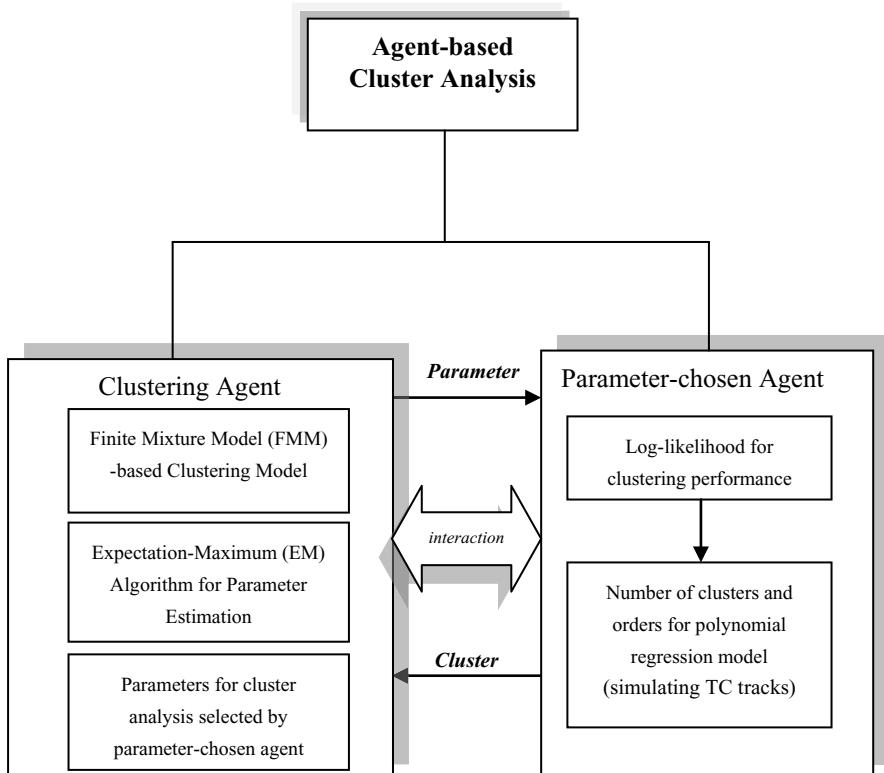


Fig. 1. System Architecture of Agent-based Cluster Analysis of Tropical Cyclone Tracks

The parameter-chosen agent evaluates the clustering performance generated by clustering agent and decides whether the results need improvements or not. If the evaluation for the clustering reveals poor performance, the parameter-chosen agent requires the clustering agent to perform a better cluster analysis. Performance evaluation of the clustering results is undertaken by considering the log-likelihood for the EM algorithm, which is based on the maximum likelihood framework. Meanwhile, the parameter-chosen agent records the parameters for the clustering models corresponding to the clustering results. After the initial parameters are set, the algorithm can cease under the certain stop criterion. It is observed in Fig. 1 that the clustering agent and parameter-chosen agent interact with each other for cluster analysis.

2.3 Improvements of EM Algorithm

In solution of the clustering model built by Gaffney, the ML estimates for Θ do not, in general, yield closed-form solutions since the estimates rely nonlinearly on each other [23, 24]. However the development of a general, iterative ML procedure, called EM algorithm, provides an efficient framework for parameter estimation in the mixture context [15, 34]. EM is an approximate root-finding procedure that is used to seek the root of the likelihood equation—it iteratively searches for a set of parameters Θ that maximize the probability of the observed data, here the historical TC tracks.

In general, EM algorithm is comprised of two steps - E-step to calculate the expectation and M-step to maximize the log likelihood. The traditional EM algorithm is complained by users for the low efficiency, especially encountering large data amount. Some accelerating EM algorithms, for example, Lazy EM, Sparse EM, Incremental EM, and SPIEM appeared in the last decades.

The techniques for the acceleration of EM can be largely divided into two groups: the succession of expectation step E and maximization step M, and those considering globally the two steps as defining a mapping:

$$\begin{aligned} F : \mathbb{R}^p &\rightarrow \mathbb{R}^p \\ \theta &\mapsto F(\theta) \end{aligned} \tag{8}$$

where θ denotes the parameter over which the likelihood has to be maximized [3]. The PX-EM [30] which modifies the set Θ of possible values of θ , and the SEM [13], as well as Lazy EM, Sparse EM and Incremental EM belong to the first group whereas fixed point numerical methods belong to the second one.

2.3.1 First EM Accelerating Group

There is a famous EM acceleration algorithm coined “Lazy EM” [36]. In fitting a mixture model to a data set via EM algorithm, the posterior probability of i th component membership of the mixed model, $\tau_i(y_j; \Theta^{(k)})$, for some y_j (object), are often observed to be close to one for some i ; that is,

$$\max \tau_i(y_j; \Theta^{(k)}) \geq C \tag{9}$$

For some y_j after the k^{th} iteration, where C is some threshold specified to be very close to one, say $C > 0.95$. If the above formula continues to hold beyond the k^{th} iteration, there would be no need to update these posterior probabilities when performing the E-step on subsequent EM iteration. Suppose that the n sample observations y_1, \dots, y_n are divided into blocks as before with the Lazy EM algorithm, but where now there are two blocks and the y_i belonging to the first block satisfy formula with the remaining observations put in the second block. The Lazy EM algorithm would be calculated in the same manner as the classical EM algorithm, except that only the posterior probabilities of component membership of those observations y_j in the second block would be updated on the E-step.

The empirical justification for the Lazy EM algorithm is that if an observation y_j has a very high posterior probability of belonging to one of the components of the mixed model, it is expected that it will not change its probabilistic component membership in subsequent iterations; and if it does, the change will occur gradually rather than suddenly. The idea therefore is to run the Lazy EM algorithm for a number of iterations k_1 between full E-step. Initially, the EM algorithm would be run for k_0 iterations before switching to the Lazy EM algorithm, which is then run for k_1 iterations between full EM steps.

The Lazy EM algorithm is theoretically justified by noticing that the formation of [36] is applicable for an arbitrary and not necessary an exhaustive set of data blocks, as long as the data are regularly. The Lazy EM algorithm is, therefore, guaranteed to converge at a local maximum (or saddle point).

2.3.2 The Second EM Accelerating Group

When it comes to the second EM Accelerating group, Berlinet and Roland [3] proposed a new algorithm to accelerate the EM algorithm via sequence transformation in Hilbert space.

To complete the acceleration, it is just like to solve fixed point problem: find $x \in H$ such that $x = F(x)$ where F is a mapping from H to H . A well-known method to solve this kind of problem is Picard's iterative procedure defined by a starting point x_0 and the iteration $x_{n+1} = F(x_n)$. In order to practically deal with the slow convergence, the number of evaluations of F will be significantly reduced with the new sequence.

$$\psi_{\alpha\beta}(x, y, z) = x + (\alpha + \beta)(y - x) + \alpha\beta(z - 2y + x) \quad (10)$$

where α and β can be considered as free parameters. The choice of α determines a point a_n on the straight line joining x_n and x_{n+1} and a second one b_n on the straight line joining x_{n+1} and x_{n+2} . Then the choice of β determines a third point on the straight line defined by a_n and b_n . This third point is used in the acceleration algorithm. Different β leads to different accelerating algorithms. $r\text{-}\Psi\alpha\alpha$ and $\Psi\alpha 2$ are proposed to get remarkable stability by all numerical tests [3].

2.3.3 The New EM Accelerating Algorithm

The algorithms of two EM acceleration groups have been discussed. That is, the $\Psi\alpha 2$ and $r\text{-}\Psi\alpha\alpha$ are stably accelerating EM algorithms in the first group while Lazy EM is

a stable algorithm of the second group. Through integration of the two algorithms, the higher acceleration effect is expected.

```

 $\theta_0 = x_n$ 
 $\theta_1 = F(\theta_0)$ 
 $\theta_2 = F(\theta_1)$ 
 $Delta1 = \theta_1 - \theta_0$ 
 $Delta2 = \theta_2 - \theta_1 - Delta1$ 
 $\alpha = -\langle Delta1, Delta2 \rangle_H / \|Delta2\|_H^2$ 
 $\beta = 2$ 
 $n = n + 1$ 
 $x_n = \theta_0 + (\alpha + \beta)Delta1 + \alpha\beta Delta2$ 
if  $n > ite \max$  or  $|L(x_{n+1}) - L(x_n)| < \varepsilon$ , stop

```

Above is the Pseudo-code for $\Psi\alpha2$, if LEM is integrated in the second and third line to calculate $\theta1$ and $\theta2$, it should be $\theta1 = \text{FLazy}(\theta0)$, $\theta2 = \text{FLazy}(\theta1)$. The other steps keep the same. The new algorithm is entitled “Lazy- $\Psi\alpha2$ ” EM algorithm.

For testing the acceleration effect, the TC tracks from 1949 to 2006 in WNP are clustered by the probabilistic clustering model proposed by Gaffney [24], which is solved by classical EM, Lazy EM, $\Psi\alpha2$ EM, Lazy- $\Psi\alpha2$ EM algorithms respectively to compare the efficiency and effect of each EM algorithm. We run the model for ten times by each algorithm and the average CPU time is taken to measure the efficiency.

From table 1, the Lazy- $\Psi\alpha2$ EM is the most efficient algorithm among the four EM algorithms. Meanwhile, the clustering effect which is measured by log likelihood is almost the same as classical EM algorithm. It is noted that the efficiency of Lazy- $\Psi\alpha2$ is about five times faster than classical EM algorithm.

Table 1. Average calculating time of different EM algorithms

EM algorithms	CPU Time(S)	Iteration	Log likelihood (10^5)
EM	173.17190	50.0	-2.26525
Lazy EM	75.56564	42.8	-2.26212
$\Psi\alpha2$	55.53645	16.8	-2.26905
Lazy- $\Psi\alpha2$	39.01565	15.8	-2.27180

2.4 Sensitivity Analysis (SA) for Cluster Analysis

In order to test the robustness of the clustering model, SA is conducted after cluster analysis. The significances of SA are three-fold, i.e., to test the stability and robustness of the clustering model, to apply the clustering model to another TC tracks archives, and to better control the model.

SA is used to determine how “sensitive” a model is to changes in the value of the parameters of the model and to changes in the structure of the model. SA is also used to discuss the relationship between “input” and “output” of a model [38, 39]. The conceptions of “input” and “output” have gradually gone so far to examine the parameters. In this paper, we will focus on discussing the relationship between the “input” and “output” of the clustering model.

In TC clustering model, “input” is TC tracks whereas “output” is the clustering results; for example, the number of the clusters and the mean curve of each cluster. The TC data used in this paper are based on the TC archives collected by Shanghai Typhoon Institute available at 6-hour sampling frequency over the time interval 1949–2006. The TC tracks in WNP are defined to be within the rectangular region ($0\text{--}60^{\circ}\text{N}$ and $100\text{--}180^{\circ}\text{E}$) during at least part of lifespan. The TC tracks are subdivided into different parts according to time periods, such as 1949–1958, 1949–1968, 1949–1978, 1949–1988, 1949–1998 and 1949–2006. The number of TCs in each time span is shown in Table 2.

Table 2. Number of TCs in each time period

<i>Year Span</i>	<i>Number of TC</i>
1949– 1958	354
1949–1968	768
1949–1978	1147
1949–1988	1476
1949–1998	1798
1949–2006	1995

In order to be consistent with the results in section 3, we fix the number of clusters to be 12 for each time period during sensitivity analysis. The same clusters of six time periods are plotted in Fig. 2.

The first, ninth, tenth and twelfth clusters of six time period are mostly alike whereas the other clusters of six time spans are similar to each other except some abnormal situations. Thereafter, based on the results derived from the sensitivity analysis, the clustering model for TC tracks are not sensitive to the amount of TC tracks.

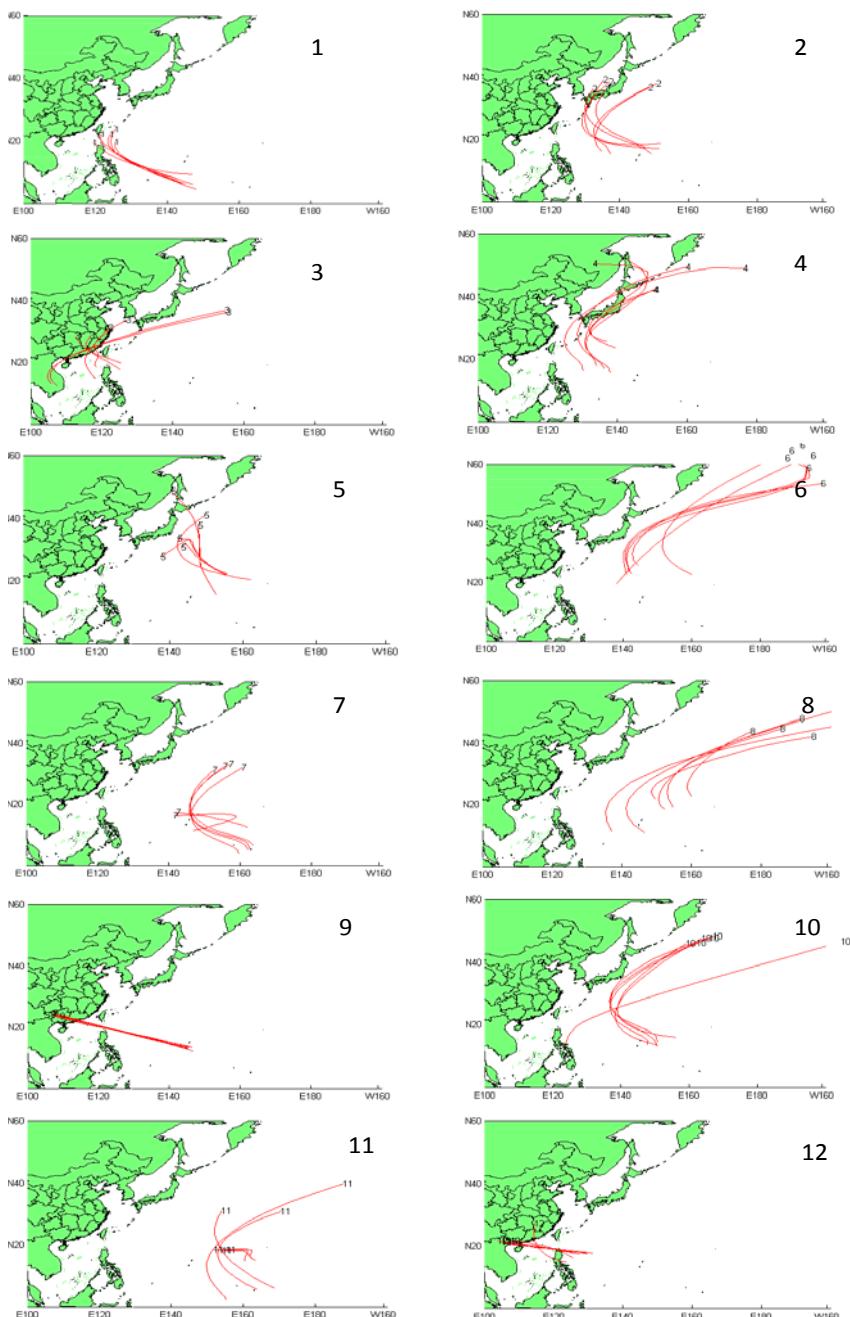


Fig. 2. Sensitivity Analyses for Clustering Model by various TC Tracks archives (The numbers from 1 to 12 indicate the ordinal number of the cluster)

3 Cluster Analysis of TC Tracks

3.1 Clustering Results

The attention is focused on the clustering of TC tracks in WNP. The TC dataset is collected and made available from Shanghai Typhoon Institute. The clustering model is elaborated in subsection 2.1, and the model is solved by Lazy- $\Psi\alpha$ 2 EM algorithm.

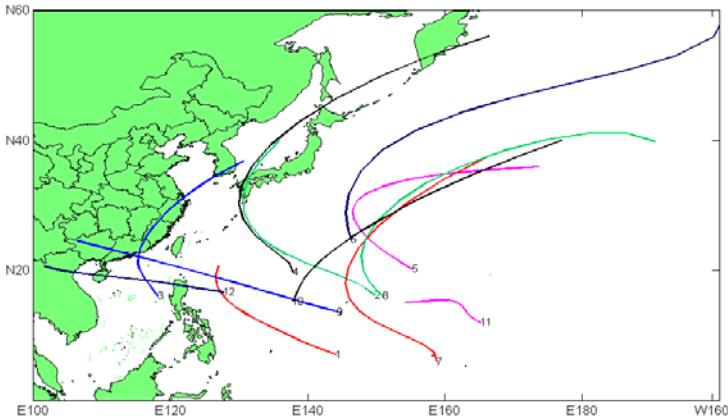


Fig. 3. The twelve mean regression tracks of 12 clusters

The first step of FMM-based cluster analysis is to choose the number of clusters and orders for polynomial regression model based on the TC data set. Using the agent-based cluster analysis, the tool automatically determines the proper parameters and unravels the clusters from the historical database. As illustrated in Fig. 3, the resultant 12 curves indicate the 12 clusters unraveled by the agent-based cluster analysis. Each curve indicates the mean regression track of corresponding cluster.

Fig. 4 illustrates the 12 clusters and TC tracks belonging to each cluster. It can be observed that the first, second, third, fourth, ninth and twelfth clusters tend to affect coastal areas in western North Pacific. The other clusters draw little impact on these areas. The number of clusters is selected to be twelve by the agent-based cluster analysis with respect to the TC tracks from 1949 to 2006. We verify the results by manually determining the number of clusters. There exist a variety of tools and methods such as Akaike's Information Criterion, Bayesian Information Criterion, Efron Information Criterion, Likelihood Ratio Test Statistic and Cross-Validation-Based Information Criterion. As the number of clusters increases from one to twelve, the log likelihood goes up steadily; the rising speed is especially dramatic from one to four (see Fig. 5). However, it drops when the number is bigger than twelve. Twelve is the potential number of clusters for the cluster analysis chosen by the algorithm based on the TC tracks from 1949 through 2006 in western North Pacific.

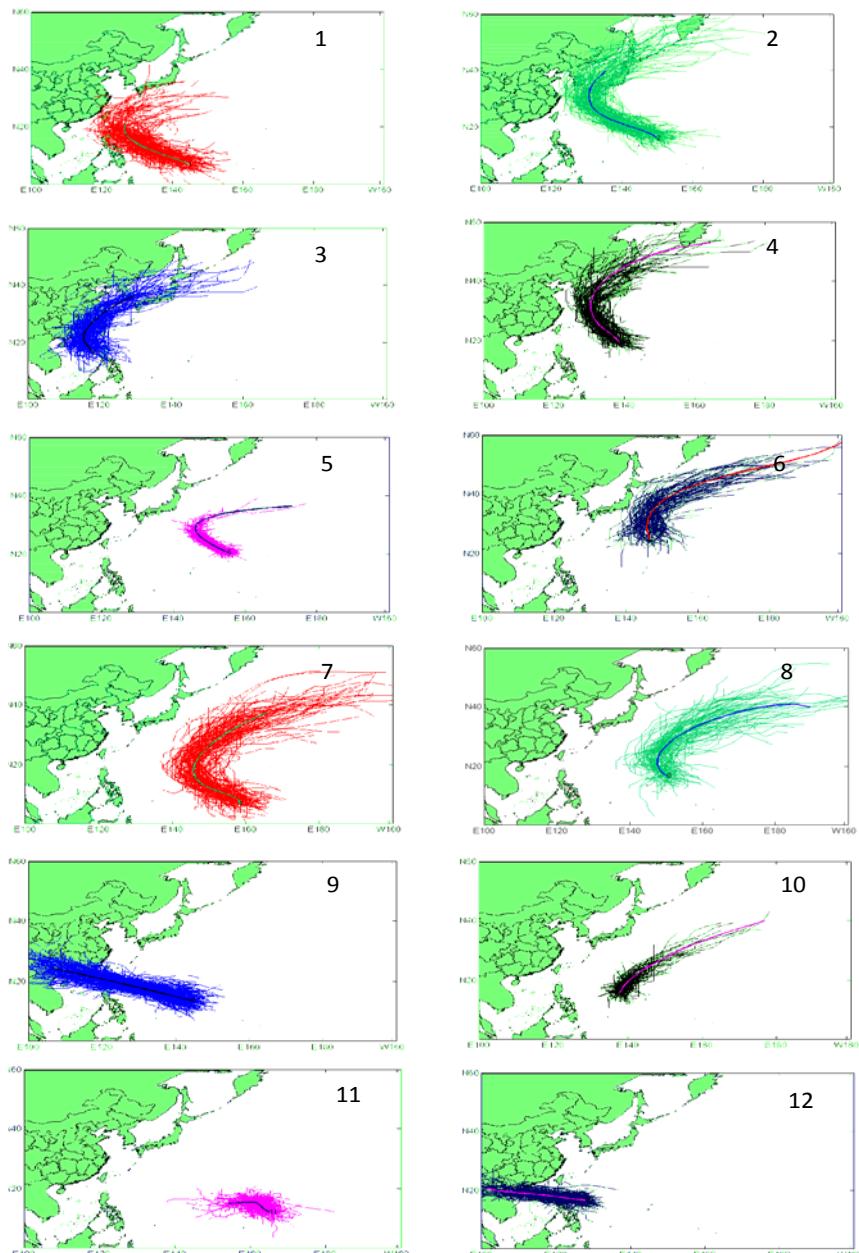


Fig. 4. The mean regression track and TC tracks belonging to each cluster. The numbers in the top right corner of each plot signify the ordinal number of the cluster; for example, “1” indicates the cluster 1 and other are defined likewise.

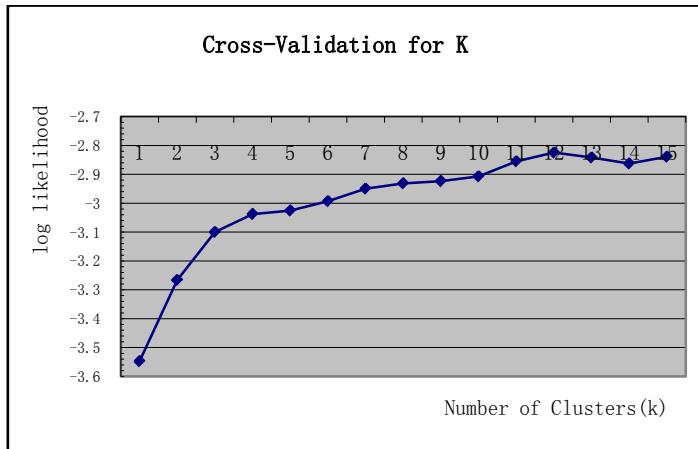


Fig. 5. Cross validation to determine the proper number of clusters

3.2 Compared with the Clustering Results of Other Researches

Gaffney and Camargo have conducted various researches on cluster analysis of TC tracks by FMM and classical EM algorithm as discussed in section 2 [9, 10, 23, 24]. We compare the results from the perspectives of algorithm, efficiency, the rank of the considered TCs, the time interval of TCs, the months of the TCs, the number of TC tracks and the number of clusters. The comparing results are described in Table 3.

Table 3. The comparisons between this study and other similar studies

Research	Algorithm	Efficiency	Intensity Level	Time Period	Month	NOT*	NOC*
Present Study	Lazy- $\Psi\alpha_2$ EM	High	all	1949-2006	All	1995	12
Gaffney	Classical EM	Low	Tropical Storm or above	1950-2001	Jul to Dec	1198	10
Camargo	Classical EM	Low	Tropical Storm or above	1950-2002	All	1393	7

* “NOC” and “NOT” mean “Number of Cluster” and “Number of TCs” respectively.

As described in Table 3, the differences are specified in each aspect. The tracks we used are comprised of all intensity levels of TCs from 1949 to 2006. Therefore, the number of clusters is larger than the cluster number of [9, 10, 23, 24].

4 Conclusion

The EM algorithm is complained about its low efficiency at times. In order to improve the efficiency, we proposed a new algorithm “Lazy- $\Psi\alpha2$ ” EM to accelerate the classical EM algorithm without sacrificing the clustering effect. The clustering model based on FMM for clustering TC tracks has proven to be stable and robust according to sensitivity analysis. The agent-based cluster analysis framework includes interactive clustering agent and parameter-chosen agent. The clustering agent implements the cluster analysis and generates results including different clusters and measurements for clustering performance, whereas the parameter-chosen agent aims at selecting the proper parameters for cluster analysis using the results generated by clustering agent. Using the agent-based cluster analysis framework, the TC tracks from 1949 to 2006 in WNP are classified into twelve clusters, which is solved by “Lazy- $\Psi\alpha2$ ” EM algorithm. Of the twelve clusters, each cluster has special characteristics. The cluster analysis of TC tracks is mainly on spatial aspect, and more attention will be paid to temporal aspect, using time series to mine the TC tracks in further investigation.

Acknowledgements. The project was supported by the research grant (08ZR1406600) of Shanghai Natural Science Foundation. The authors would like to thank Prof. Chaoyi Li at East China Normal University for his valuable suggestions throughout the research.

Reference

1. Anthes, R.: Tropical Cyclones: Their Evolution, Structure and Effects. Meteor. Monogr. (41), 0-933876 (1982)
2. Bazzan, A.L.C., Wahle, J., Klügl, F.: Agents in Traffic Modelling - From Reactive to Social Behaviour. In: Burgard, W., Christaller, T., Cremers, A.B. (eds.) KI 1999. LNCS (LNAI), vol. 1701, pp. 303–306. Springer, Heidelberg (1999)
3. Berlinet, A., Roland, C.: Acceleration schemes with application to the EM algorithm. Computational Statistics & Data Analysis 51, 3689–3702 (2007)
4. Blender, R., Fraedrich, K., Lunkeit, F.: Identification of cyclone-track regimes in the North Atlantic. Quarterly Journal of the Royal Meteorological Society 123, 727–741 (1997)
5. Bousquet, F., Le Page, C.: Multi-agent simulations and ecosystem management: a review. Ecological Modelling 176, 313–332 (2004)
6. Burmeister, B., Haddadi, A., Matylis, G.: Application of multi-agent systems in traffic and transportation. In: IEE Proceedings Software Engineering, vol. 144, pp. 51–60 (1997)
7. Camargo, S., Robertson, A., Gaffney, S., Smyth, P.: Cluster analysis of western North Pacific tropical cyclone tracks, pp. 250–251 (2004)
8. Camargo, S.J., Robertson, A.W., Barnston, A.G., Ghil, M.: Clustering of eastern North Pacific tropical cyclone tracks: ENSO and MJO effects. Geochemistry Geophysics Geosystems 9 (2008)
9. Camargo, S.J., Robertson, A.W., Gaffney, S.J., Smyth, P., Ghil, M.: Cluster analysis of typhoon tracks. Part I: General properties. Journal of Climate 20, 3635–3653 (2007)
10. Camargo, S.J., Robertson, A.W., Gaffney, S.J., Smyth, P., Ghil, M.: Cluster analysis of typhoon tracks. Part II: Large-scale circulation and ENSO. Journal of Climate 20, 3654–3676 (2007)

11. Carr III, L., Elsberry, R.: Dynamical tropical cyclone track forecast errors. Part II: Midlatitude circulation influences (2000)
12. Carr III, L., Elsberry, R.: Dynamical tropical cyclone track forecast errors. Part I: tropical region error sources. *Weather and Forecasting* 15, 641–661 (2000)
13. Celeux, G., Diebolt, J.: The SEM algorithm: a probabilistic teacher algorithm derived from the EM algorithm for the mixture problem. *Computational Statistics Quarterly* 2, 73–82 (1985)
14. d’Inverno, M., Luck, M.: *Understanding agent systems*. Springer, Heidelberg (2004)
15. Dempster, A., Laird, N., Rubin, D.: Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)* 39, 1–38 (1977)
16. Elsberry, R., Carr III, L.: Consensus of dynamical tropical cyclone track forecasts-Errors versus spread. *Monthly Weather Review* 128, 4131–4138 (2000)
17. Elsner, J.B.: Tracking hurricanes. *Bulletin of the American Meteorological Society* 84, 353–356 (2003)
18. Elsner, J.B., Liu, K.B.: Examining the ENSO-typhoon hypothesis. *Climate Research* 25, 43–54 (2003)
19. Everitt, B., Hand, D.: *Finite mixture distributions*. Chapman & Hall (1981)
20. Ferber, J.: *Multi-agent systems: an introduction to distributed artificial intelligence*, vol. 222. Addison-Wesley, New York (1999)
21. Gaffney, S., Smyth, P.: Trajectory clustering with mixtures of regression models. In: *Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM (1999)
22. Gaffney, S., Robertson, A., Smyth, P., Camargo, S., Ghil, M.: Probabilistic clustering of extratropical cyclones using regression mixture models. *Climate Dynamics* 29, 423–440 (2007)
23. Gaffney, S.J.: Probabilistic curve-aligned clustering and prediction with regression mixture models, p. 281. University of California, Irvine (2004)
24. Goerss, J.: Prediction of consensus tropical cyclone track forecast error. *Monthly Weather Review* 135, 1985–1993 (2007)
25. Gray, W., Landsea, C., Mielke Jr., P., Berry, K.: Predicting Atlantic basin seasonal tropical cyclone activity by 1 June. *Weather and Forecasting* 9, 103–115 (1994)
26. Han, J., Kamber, M.: *Data mining: concepts and techniques*. Morgan Kaufmann (2006)
27. Harr, P.A., Elsberry, R.L.: Large-Scale Circulation Variability over the Tropical Western North Pacific. Part I: Spatial Patterns and Tropical Cyclone Characteristics. *Monthly Weather Review* 123, 1225–1246 (1995)
28. Lee, J., Han, J., Whang, K.: Trajectory clustering: a partition-and-group framework, pp. 593–604. ACM (2007)
29. Lesser, V.: *Multi-agent systems*, pp. 1194–1196. John Wiley and Sons Ltd. (2003)
30. Liu, C., Rubin, D., Wu, Y.: Parameter expansion to accelerate EM: the PX-EM algorithm. *Biometrika* 85, 755 (1998)
31. Luck, M., McBurney, P., Preist, C.: *Agent Technology: Enabling Next Generation Computing (A Roadmap for Agent Based Computing)* (2003)
32. MacQueen, J.: Some methods for classification and analysis of multivariate observations, California, USA, p. 14 (1967)
33. McLachlan, G., Basford, K.: *Mixture Models: Inference and Applications to Clustering*. Marcel Dekker, New York (1988)
34. McLachlan, G., Krishnan, T.: *The EM Algorithm and Extensions*. Wiley, New York (1997)

35. McLachlan, G., Peel, D.: Finite mixture models. Wiley-Interscience (2000)
36. Neal, R., Hinton, G.: A view of the EM algorithm that justifies incremental, sparse, and other variants. Learning in Graphical Models 89, 355–368 (1998)
37. Parker, D., Manson, S., Janssen, M., Hoffmann, M., Deadman, P.: Multi-Agent Systems for the Simulation of Land-Use and Land-Cover Change: A Review. Annals of the Association of American Geographers 93, 314–337 (2003)
38. Saltelli, A., Chan, K., Scott, E.: Sensitivity analysis, vol. 475. Wiley, New York (2000)
39. Saltelli, A., Ratto, M., Andres, T., Corporation, E.: Global sensitivity analysis: the primer. Wiley Online Library (2008)
40. Simpson, A., Riehl, H.: The hurricane and its impact. Louisiana State University Press (1981)
41. Wahle, J., Bazzan, A., Klügl, F., Schreckenberg, M.: Anticipatory traffic forecast using multi-agent techniques, pp. 87–92 (2000)
42. Wang, Y., Wu, C.C.: Current understanding of tropical cyclone structure and intensity changes - a review. Meteorology and Atmospheric Physics 87, 257–278 (2004)
43. Weiss, G.: Multiagent systems: a modern approach to distributed artificial intelligence. The MIT press (2000)
44. Cao, L., Gorodetsky, V., Mitkas, P.: Agent Mining: The Synergy of Agents and Data Mining. IEEE Intelligent Systems 24(3), 64–72 (2009)

Exploiting Domain Knowledge in Making Delegation Decisions

Chukwuemeka David Emele¹, Timothy J. Norman¹,
Murat Sensoy¹, and Simon Parsons²

¹ University of Aberdeen, Aberdeen, AB24 3UE, UK

² Brooklyn College, City University of New York, 11210 NY, USA

{c.emele,t.j.norman,m.sensoy}@abdn.ac.uk,

parsons@sci.brooklyn.cuny.edu

Abstract. In multi-agent systems, agents often depend on others to act on their behalf. However, delegation decisions are complicated in norm-governed environments, where agents' activities are regulated by policies. Especially when such policies are not public, learning these policies become critical to estimate the outcome of delegation decisions. In this paper, we propose the use of domain knowledge in aiding the learning of policies. Our approach combines ontological reasoning, machine learning and argumentation in a novel way for identifying, learning, and modeling policies. Using our approach, software agents can autonomously reason about the policies that others are operating with, and make informed decisions about to whom to delegate a task. In a set of experiments, we demonstrate the utility of this novel combination of techniques through empirical evaluation. Our evaluation shows that more accurate models of others' policies can be developed more rapidly using various forms of domain knowledge.

Keywords: Argumentation, Machine learning, Decision Making, Policies/Norms, Ontologies.

1 Introduction

In many settings, agents (whether human or artificial) engage in problem solving activities, which often require them to share resources, act on each others' behalf, communicate and coordinate individual acts, and so on. Such problem-solving activities may fail to achieve desired goals if the plan is not properly resourced and tasks delegated to appropriately competent agents. Irrespective of the field of endeavour, the overall success of problem-solving activities depends on a number of factors; one of which is the selection of appropriate candidates to delegate tasks to (or share resources with). However, successful delegation decisions depend on various factors. In norm-governed environments, one of the factors for making successful delegation decisions is the accuracy of the prediction about the policy restrictions that others operate with.

In multi-agent environments, agents may operate under policies, and some policies may prohibit an agent from providing a resource to another under certain circumstances. Such policies might regulate what resources may be released to a partner from some other organisation, under what conditions they may be used, and what information regarding their use is necessary to make a decision. In addition, policies may govern actions that can be performed either to pursue individual goals or on behalf of another. In Emele *et al.* [2], we show that intelligent agents can determine what policies others are operating within by mining the data [1] gathered from past encounters with that agent (or similar agents) as they collaborate to solve problems. This prior research uses a novel combination of argumentation-derived evidence (ADE) and machine learning in building stable models of others' policies. Here we explore the question that given agents may have access to some background (or ontological) domain knowledge, how can we exploit such knowledge to improve models of others' policies? To do this, we propose the use of ontological reasoning, argumentation and machine learning to aid in making effective predictions about who to approach if some other collaborator is to be delegated to perform a task on the behalf of another.

The rest of this paper is organised as follows. Section 2 discusses delegation in norm-governed environments. Section 3 presents our approach for learning policies. Section 4 reports the results of our evaluations. Section 5 summarises our findings, discusses related work and outlines future directions.

2 Delegating in Norm-Governed Environments

Task delegation, in general, is concerned with identifying a suitable candidate (or candidates) to transfer authority to act on one's behalf [5]. In other words, we are concerned with finding candidate agents that will achieve a given goal on our behalf. Implicitly, this model is based on experience gathered from past encounters. In a norm-governed multi-agent system where each agent is regulated by a set of rules, referred to as *policies* (or norms), the policies could determine what actions an agent is (or is not) allowed to perform. Delegation, in this case, is not just a matter of finding agents that possess the appropriate expertise (or resource); if an agent has the required expertise but is operating under a policy that prohibits the performance of that action, it may not take on the delegated task. Nevertheless, delegation in such settings entails finding agents who possess the required expertise (or resources), and whose policies permit the performance of the required action (or provide the required resource). If an agent is allowed to perform an action (according to its policy) then we assume it will be willing to perform it when requested, provided it has the necessary resources and/or expertise, and that doing so does not yield a negative utility. In our framework, an agent that has been assigned a task is solely responsible for that task. However, an agent can delegate an aspect of a task to another. For example, agent x is responsible for performing task T_x but could delegate the provision of some resource R_r required to fulfill T_x to another agent y_1 . Provided agent y_1 has resource R_r , and does not have any policy that forbids the provision of R_r then we assume y_1 will make R_r available to x .

From the above example, we see that agent x needs to find effective ways of delegating the provision of resource R_r . In order to delegate a task successfully, we need to find out the agent whose policy constraints will most likely, according to a chosen metric, permit it to execute the delegated task. In our framework, whenever there is a task to be delegated, policy predictions are generated alongside the confidence of those predictions from the policy models that have been learned over time. Confidence values of favourable policy predictions are easily compared to determine which candidate to delegate the task to. In our case, confidence values range from 0 to 1, with 0 being *no confidence* in the prediction, and 1 being *complete confidence*.

The delegating agent explores the candidate space to identify suitable candidates to whom it can delegate a task. In these terms, our delegation problem is concerned with finding potential candidates whose policies permit to perform the delegated task, and thereafter, selecting the most promising candidate from the pool of eligible candidates. Borrowing ideas from economics, we assume that some payment will be made to an agent for performing a delegated task (e.g. payment for the provision of a service).

Example 1. Consider a situation where an agent x is collaborating with a number of agents, y_1, y_2, y_3 , and y_4 , to solve an emergency response problem. Let us assume that agent x does not have a helicopter in its resource pool, and that each of agents y_1, y_2, y_3 , and y_4 can provide helicopters, jeeps, vans, bikes, fire extinguishers, and unmanned aerial vehicles (UAVs).

Agent x in Example 1 has to decide which of the potential providers, y_1, y_2, y_3 , and y_4 to approach to provide the *helicopter*. Let us assume that the four providers advertise similar services. Agent x , at this point, attempts to predict the policy of the providers with respect to task delegation (or resource provision). This prediction is based on policy models built from past experience with these providers (or similar agents). Assuming the predictions are as follows: (i) y_1 will accept to provide the helicopter with 0.6 confidence; (ii) y_2 will accept with 0.9 confidence; (iii) y_3 will accept with 0.7 confidence; and (iv) y_4 will decline with 0.8 confidence. If the decision to choose a provider is based on policy predictions alone, then y_2 is the best candidate.

3 Learning Agent Policies

The framework we propose here enables agents to negotiate and argue about task delegation, and use evidence derived from argumentation to build more accurate and stable models of others' policies. The architecture of our framework, sketched in Figure 1, enables agents to learn the policies and resource availabilities of others through evidence derived from argumentation, and improve those models by exploiting domain knowledge. The dialogue manager handles all communication with other agents. The learning mechanism uses machine learning

techniques to reason over the dialogue and attempts to build models of other agents' policies and resource availabilities based on arguments exchanged during encounters. The arguments include the features that an agent requires in order to make a decision about accepting a task delegation or not. The agent attempts to predict the policies of others by reasoning over policy models (built from past experience). Such reasoning is further improved by exploiting background domain knowledge and concept hierarchies in an ontology (see Section 3.4).

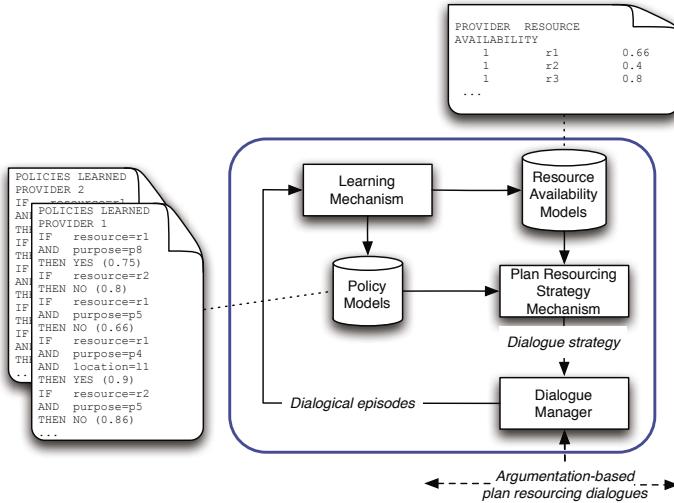


Fig. 1. Agent reasoning architecture

3.1 Policies

In this framework, agents have policies that govern how resources are deployed to others. In our model, policies are conditional entities (or rules) and so are relevant to an agent under specific circumstances only. These circumstances are characterised by a set of features, e.g., vehicle type, weather conditions, etc.

Definition 1 (Features). Let \mathcal{F} be the set of all features such that $f_1, f_2, \dots \in \mathcal{F}$. We define a feature as a characteristic of the prevailing circumstance under which an agent is operating (or carrying out an activity).

Our concept of policy maps a set of features into an appropriate policy decision. In our framework, an agent can make one of two policy decisions at a time, namely (1) *grant*, which means that the policy allows the agent to provide the resource when requested; and (2) *deny*, which means that the policy prohibits the agent from providing the resource.

Definition 2 (Policies). A policy is defined as a function $\Pi : \vec{\mathcal{F}} \rightarrow \{\text{grant}, \text{deny}\}$, which maps feature vectors of agents, $\vec{\mathcal{F}}$, to appropriate policy decisions.

Policy Id	f_1	f_2	f_3	f_4	f_5	Decision
\mathbb{P}_1	h	trm				grant
\mathbb{P}_2	av		vc			deny
\mathbb{P}_3	j					grant
\mathbb{P}_4	c		vc	xx		grant
...
\mathbb{P}_n	q	yy	w	xx	z	deny

Fig. 2. An agent's policy profile

In order to illustrate the way policies may be captured in this model, we present an example. Let us assume that f_1 is resource, f_2 is purpose, f_3 is weather report (with respect to a location), f_4 is the affiliation of the agent, and f_5 is the day the resource is required, then \mathbb{P}_1 , \mathbb{P}_2 , and \mathbb{P}_3 in Figure 2 will be interpreted as follows:

- \mathbb{P}_1 : You are **permitted** to release a *helicopter* (h), to an agent if the *helicopter* is required for the purpose of transporting relief materials (trm);
- \mathbb{P}_2 : You are **prohibited** from releasing an *aerial vehicle* (av) to an agent in bad weather conditions - e.g. volcanic clouds (vc);
- \mathbb{P}_3 : You are **permitted** to release a *jeep* (j) to an agent.

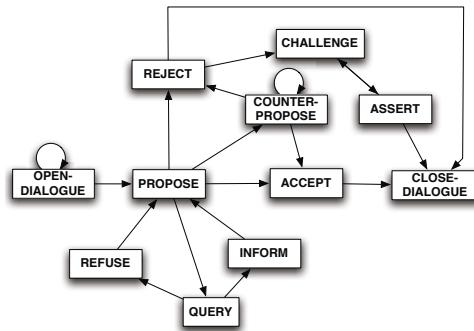
In the foregoing example, if *helicopter* is intended to be deployed in an area with volcanic clouds then the provider is forbidden from providing the resource but might offer a ground vehicle (e.g. *jeep*) to the seeker if there is no policy prohibiting this and the resource is available. Furthermore, whenever a seeker's request is refused, the seeker may challenge the decision, and seek justifications for the refusal. This additional evidence is beneficial, and could be used to improve the model, hence, the quality of decisions made in future episodes.

3.2 Argumentation-Based Negotiation

Figure 3 illustrates the protocol employed in this framework, which guides dia-logical moves. Our approach in this regard is similar to the dialogue for resource negotiation proposed by McBurney & Parsons [4].

To illustrate the sorts of interaction between agents, consider the example dialogue in Figure 4. Let x and y be seeker and provider agents respectively. Suppose we have an argumentation framework that allows agents to ask for and receive explanations (as in Figure 4, *lines 11 and 12*), offer alternatives (counter-propose in Figure 3), or ask and receive more information about the attributes of requests (*lines 4 to 9* in Figure 4), then x can gather additional information regarding the policy rules guiding y concerning provision of resources.

Negotiation for resources takes place in a turn-taking fashion. The dialogue starts, and then agent x sends a request (propose in Figure 3) to agent y , e.g. line 3, Figure 4. The provider, y , may respond by conceding to the request (accept), rejecting it, offering an alternative resource (counter-propose), or asking for more

**Fig. 3.** The negotiation protocol

information (query) such as in line 4 in Figure 4. If the provider agrees to provide the resource then the negotiation ends. If, however, the provider rejects the proposal (line 10, Figure 4) then the seeker may challenge that decision (line 11), and so on. If the provider suggests an alternative then the seeker evaluates it to see whether it is acceptable or not. Furthermore, if the provider agent needs more information from the seeker in order to make a decision, the provider agent would ask questions that will reveal the features it requires to make a decision (query, inform/refuse). The negotiation ends when agreement is reached or all possibilities explored have been rejected.

#	Dialogue Sequence	Locution Type
1	x: Start dialogue.	OPEN-DIALOGUE
2	y: Start dialogue.	OPEN-DIALOGUE
3	x: Can I have a <i>helicopter</i> for \$0.1M reward?	PROPOSE
4	y: What do you need it for?	QUERY
5	x: To transport relief materials.	INFORM
6	y: To where?	QUERY
7	x: A refugee camp near Indonesia.	INFORM
8	y: Which date?	QUERY
9	x: On Friday 16/4/2010.	INFORM
10	y: No, I can't provide you with a <i>helicopter</i> .	REJECT
11	x: Why?	CHALLENGE
12	y: I am not permitted to release a <i>helicopter</i> in volcanic eruption.	ASSERT
13	x: There is no volcanic eruption near Indonesia.	CHALLENGE
14	y: I agree, but the ash cloud is spreading, and weather report advises that it is not safe to fly on that day.	ASSERT
15	x: Ok, thanks.	CLOSE-DIALOGUE

Fig. 4. Dialogue example

3.3 Learning from Past Experience through Dialogue

When an agent has a collection of experiences with other agents described by feature vectors (see Section 3.1), we can make use of existing machine learning techniques for learning associations between sets of discrete attributes (e.g. $f_1, f_2, \dots, f_n \in \mathcal{F}$) and policy decisions (i.e., *grant* and *deny*). In previous research we investigated three classes of machine learning algorithms: (i) instance-based learning (using k-nearest neighbours); (ii) rule-based learning (using sequential covering); and (iii) decision tree learning (using C4.5). Figure 5 shows an example decision tree representing a model of the policies of some other agent learned from interactions with that agent. Nodes of the decision tree capture features of an agent's policy, edges denote feature values, while the leaves are policy decisions.

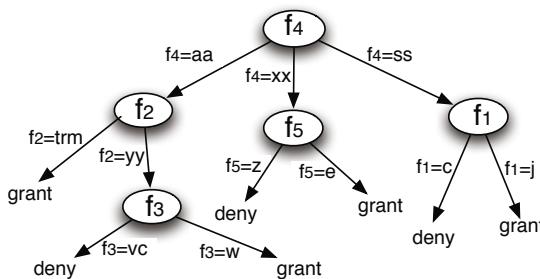


Fig. 5. Example decision tree

The machine learning algorithms were chosen to explore the utility of different classes of learning techniques. Instance-based learning is useful in this context because it can adapt to and exploit evidence from dialogical episodes incrementally as they accrue. In contrast, decision trees, and rule learning are not incremental; the tree or the set of rules must be reassessed periodically as new evidence is acquired.¹ Learning mechanisms such as sequential covering, decision trees do have a number of advantages over instance-based approaches; in particular, the rules (or trees) learned are more amenable to scrutiny by a human decision maker.

The training examples used in each learning mechanism are derived from plan resourcing episodes (or interactions), which involves resourcing a task t using provider y and may result in $(\vec{F}_y, \text{grant})$ or (\vec{F}_y, deny) . In this way, an agent may build a model of the relationship between observable features of agents and the policies they are operating under. Subsequently, when faced with resourcing a new task, the policy model can be used to obtain a prediction of whether

¹ For these algorithms we define a *learning interval*, ϕ , which determines the number of plan resourcing episodes (or interactions) an agent must engage in before building (or re-building) its policy model.

or not a particular provider has a policy that permits the provision of the resource. In this paper, we take this aspect of the research further by investigating *semantic-enriched decision trees* (*STree*), which extend C4.5 decision trees using ontological reasoning to explore how much domain knowledge can improve learning.

3.4 Learning from Domain Knowledge

In this paper, we argue that domain knowledge can be used to improve the performance of machine learning approaches. Specifically, in this section, we will describe how we can exploit domain knowledge to improve C4.5 decision trees.

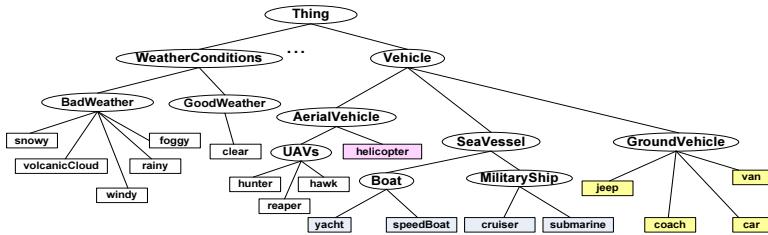


Fig. 6. A simple ontology for vehicles and weather conditions. Ellipsis and rectangles represent concepts and their instances respectively.

Domain Knowledge. Domain knowledge consists of such background information that an expert (in a field) would deploy in reasoning about a specific situation. Semantic Web technologies allow software agents to use ontologies to capture domain knowledge, and employ ontological reasoning to reason about it [3]. Figure 6 shows a part of a simple ontology about vehicles and weather conditions. The hierarchical relationships between terms in an ontology can be used to make generalisation over the values of features while learning policies as demonstrated in Example 2. Policies are often specified using numerical features (e.g., vehicle price) and nominal features (e.g., vehicle type). Each nominal feature may have a large set of possible values. Without domain knowledge, the agent may require a large training set containing examples with these nominal values. However, domain knowledge allows agents to reason about the terms unseen in the training set and learn more general policies with fewer number of training examples.

Example 2. Suppose agent x in Example 1 has learned from previous interactions with agent y_1 that there is a policy that forbids y_1 from providing a helicopter when the weather is rainy, foggy, snowy, or windy. In addition, suppose agent x has learned from previous experience that agent y_1 is permitted to provide a jeep in these conditions. This information has little value for x if it needs a helicopter when the weather is not rainy, foggy, snowy, or windy but volcanic clouds are reported. On the other hand, with the help of the ontology

in Figure 6, agent x can generalise over the already experienced weather conditions and expect that “agent y_1 is prohibited from providing helicopters in bad weather conditions”. Such a generalisation allows x to reason about y_1 ’s behavior for the cases that are not experienced yet. That is, with the help of the domain knowledge, agent x can deduce (even without having training examples involving volcanic clouds directly) that agent y_1 may be prohibited from providing a helicopter if there is an evidence of volcanic clouds in the region.

C4.5 Decision Tree Algorithm. In this section, we shortly describe the induction of C4.5 decision trees. Then, in the following section, we describe how domain knowledge can be exploited during tree induction.

The well-known C4.5 decision tree algorithm [7] uses a method known as divide and conquer to construct a suitable tree from a training set S of cases. If all the cases in S belong to the same class C_i , the decision tree is a leaf labeled with C_i . Otherwise, let B be some test with outcomes $\{b_1, b_2, \dots, b_n\}$ that produces a partition of S , and denote by S_i the set of cases in S that has outcome b_i of B . The decision tree rooted at B is shown in Figure 7, where T_i is the result of growing a sub-tree for the cases in S_i . The root node B is based on an attribute that best classifies S . This attribute is determined using information theory. That is, the attribute having the highest *information gain* is selected.

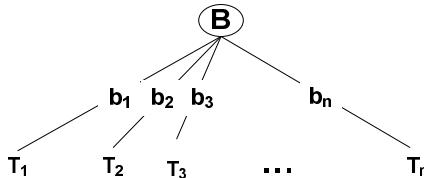


Fig. 7. Tree rooted at the test B and its branches based on its outcomes

Information gain of an attribute is computed based on *information content*. Assume that testing an attribute A in the root of the tree will partition S into disjoint subsets $\{S_1, S_2, \dots, S_t\}$. Let $RF(C_i, S)$ denote the relative frequency of cases in S that belong to class C_i . The information content of S is then computed using Equation 1. The *information gain* for A is computed using Equation 2.

$$I(S) = - \sum_{i=1}^n RF(C_i, S) \times \log(RF(C_i, S)) \quad (1)$$

$$G(S, A) = I(S) - \sum_{i=1}^t \frac{|S_i|}{|S|} \times I(S_i) \quad (2)$$

Once the attribute representing the root node is selected based on its information gain, each value of the attribute leads to a branch of the node. These branches divide the training set used to create the node into disjoint sets $\{S_1, S_2, \dots, S_t\}$.

Then, we recursively create new nodes of the tree using these subsets. If S_i contains training examples only from the class C_i , we create a leaf node labeled with the class C_i ; otherwise, we recursively build a child node by selecting another attribute based on S_i . This recursive process stops either when the tree perfectly classifies all training examples, or until no unused attribute remains.

#	Type	Age	Price	Class
1	Van	10	10,000	grant
2	Van	5	20,000	grant
3	Car	8	5,000	grant
4	Car	15	1,000	grant
5	Coach	2	200,000	grant
6	Yacht	20	300,000	deny
7	Yacht	2	500,000	deny
8	Speedboat	4	8,000	deny
9	Speedboat	15	2,000	deny
10	Cruiser	10	100,000	deny

Fig. 8. Training examples

Figure 8 lists 10 training examples, where *Type*, *Age*, and *Price* are the only features. C4.5 decision tree algorithm makes induction only over numerical attribute values. However, it could not make induction or generalisation over the nominal attribute values (i.e., terms). For instance, a decision node based on the *price* test in Figure 9 can be used to classify a new case with price \$250,000, even though there is no case in the training examples with this price value. However, a new case with an unseen type, for instance a *submarine*, cannot be classified using the decision node based on the attribute *Type*.

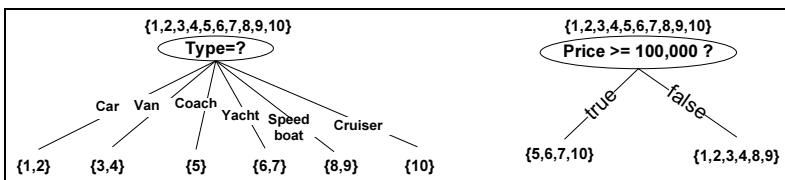


Fig. 9. Decision nodes created using the tests on *Type* (on left) and *Price* (on right)

Semantic-Enriched Decision Trees. Here, we propose semantic-enriched decision trees (*STree*) built upon the subsumptions relationships between terms in the ontology. These relationships can be derived automatically using an off-the-shelf ontology reasoner [3]. The main idea of *STree* is to replace the values of nominal attributes with more general terms iteratively during tree induction, unless this replacement results in any decrease in the classification performance.

Algorithm 1 summarises how the values of A are generalised for S . First, we compute the original gain $G(S, A)$ (line 3). Second, we create a set called

banned, which contains the terms that cannot be generalised further (line 4). Initially, this set contains only the top concept *Thing*. Third, we create the set T that contains A 's values in S (line 5). While there is a generalisable term $t \in T$ (lines 6-18), we compute its generalisation t' using ontological reasoning (line 8) and create the set T' by replacing more specific terms in T with t' (line 9). If this term is an instance of a concept, then the generalisation of the term is the concept, e.g., *Boat* is generalisation of *Yacht*. If the term is a concept, its generalisation is its parent concept, e.g., *SeaVessel* is generalisation of *Boat*. For instance, let S be the data in Figure 8, then T would contain *Yacht*, *Speedboat*, *Cruiser*, *Van*, *Car*, *Coach*, and *Cruiser*. If *Car* is selected as t , t' would be *GroundVehicle*. In this case, T' would contain *Yacht*, *Speedboat*, *Cruiser*, and *GroundVehicle*. Next, we check if the generalisation leads to any decrease in the information gain. This is done by creating a temporary training set s from S by replacing A 's values in S with the more general terms in T' (line 10) and then comparing $G(s, A)$ with the original gain g (line 11). If there is no decrease in the information gain, S and T are replaced with s and T' respectively; otherwise t is added to *banned*. We iterate through until we cannot find any term in T to generalise without any decrease in the information gain.

Algorithm 1. Generalising values of nominal attribute A in training set S

```

1: Input :  $S, A$ 
2: Output:  $T$ 
3:  $g = G(S, A)$ 
4:  $banned = \{\text{Thing}\}$ 
5:  $T = \text{getAttributeValues}(S, A)$ 
6: while true do
7:   if  $\exists t$  such that  $t \in T \wedge t \notin banned$  then
8:      $t' = \text{generalise}(t)$ 
9:      $T' = \text{replaceWithMoreSpecificTerms}(T, t')$ 
10:     $s = \text{replaceAttributeValues}(S, A, T')$ 
11:    if  $G(s, A) = g$  then
12:       $S = s$  and  $T = T'$ 
13:    else
14:       $banned = banned \cup \{t\}$ 
15:    end if
16:  else
17:    break
18:  end if
19: end while

```

The output of the algorithm would be $\{\text{SeaVessel}, \text{GroundVehicle}\}$ for the examples in Figure 8, because any further generalisation results in a decrease in information gain. Hence, a decision node based on *Type* attribute would be as shown in Figure 10 (left hand side). A new test case (11, *Submarine*, 40years, \$800,000) would be classified as *deny* using this decision node, because a submarine is a sea vessel and all known sea vessels are labeled as *deny*. If the actual classification of the case is *grant* instead of *deny*, the decision node would be

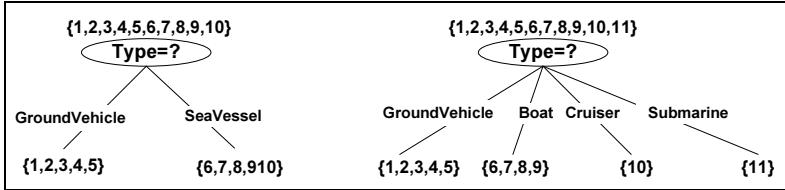


Fig. 10. Decision nodes using the generalisation of cases in Figure 8 (left hand) and after the addition of a new case (11, *Submarine*, 40, 800, 000, *grant*) (right hand)

updated as seen in Figure 10 (right hand side), because generalisation of *Submarine* or *Cruiser* now results in a decrease in the information gain.

4 Evaluation

In evaluating our approach, we employed a simulated agent society where a set of seeker agents interact with a set of provider agents with regard to resourcing their plans over a number of runs. Each provider is assigned a set of resources. Providers also operate under a set of policy constraints that determine under what circumstances they are permitted to provide resources to seekers. In the evaluation reported in this section, we demonstrate that it is possible to use domain knowledge to improve models of others' policies, hence increase their predictive accuracy, and performance. To do this, we consider two experimental conditions (i.e. closed and open). There are five features that are used to capture agents' policies, namely *resource type*, *affiliation*, *purpose*, *location*, and *day*. In the open scenario, each feature can have up to 20 different values, whereas only 5 different values are allowed in the closed scenario. In each scenario, six agent configurations (*RS*, *SM*, *C4.5*, *kNN*, *SC*, and *STree*) are investigated. In configuration *RS*, random selection is used. In *SM*, simple memorisation of outcomes is used. In *C4.5*, *C4.5* decision tree classifier is used. In *kNN*, k-nearest neighbour algorithm is used. In *SC*, sequential covering rule learning algorithm is used. Lastly, in *STree*, agents use semantic-enriched decision trees to learn policies of others.

Seeker agents were initialised with random models of the policies of providers. 100 runs were conducted in 10 rounds for each case, and tasks were randomly created during each run from the possible configurations. In the control condition (random selection, *RS*), the seeker randomly selects a provider to approach. In the *SM* configuration, the seeker simply memorises outcomes from past interactions. Since there is no generalisation in *SM*, the *confidence* (or prediction accuracy) is 1.0 if there is an exact match in memory, else the probability is 0.5.

Figure 11 gives a graphical illustration of the performance of six algorithms we considered in predicting agents' policies in the closed scenario. The results show that *STree*, *SC*, *kNN*, *C4.5* and *SM* consistently outperform *RS*. Furthermore, *STree*, *SC* and *kNN* consistently outperform *C4.5* and *SM*. It is interesting to see that, with relatively small training set, *SM* performed better than *C4.5*. This is,

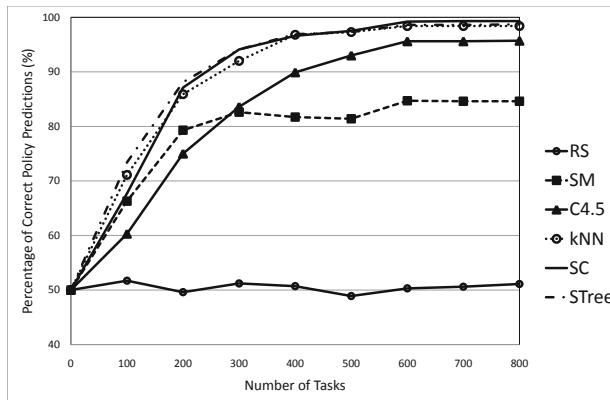


Fig. 11. The effectiveness of exploiting domain knowledge in learning policies (closed)

we believe, because the model built by C4.5 overfit the data. The decision tree was pruned after each set of 100 tasks and after 300 tasks the accuracy of the C4.5 model rose to about 83% to tie with SM and from then C4.5 performed better than SM. Similarly, STree performed much better than SC with relatively small training set. We believe, this is because STree takes advantage of domain knowledge and so can make informed inference (or guess) with respect to feature values that do not exist in the training set. After 400 tasks the accuracy of SC reached 96% to tie with STree. We believe, at this point, almost all the test instances have been encountered and so have been learned (and now exist in the training set for future episodes).

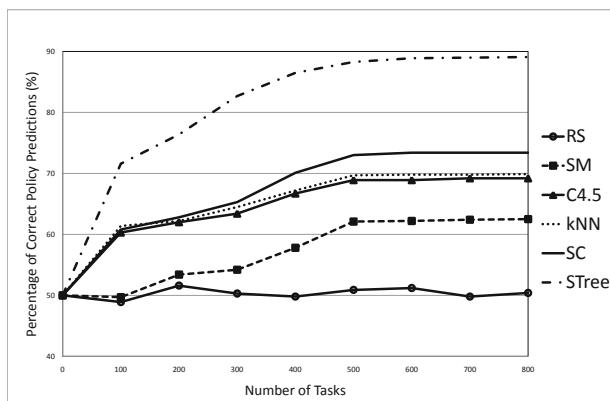


Fig. 12. The effectiveness of exploiting domain knowledge in learning policies (open)

Figure 12 illustrates the effectiveness of four learning techniques (C4.5, kNN, SC, and STree) and SM in learning policies in the open scenario. The result shows that the technique that exploits domain knowledge (STree) significantly outperforms the other techniques that did not. The decision trees (i.e. STree and C4.5) were pruned after each set of 100 tasks and after 300 tasks the accuracy of the STree model had exceeded 82% while that of C4.5 was just over 63%. These results confirm that exploiting appropriate domain knowledge in learning policies mean that more accurate and stable models of others' policies can be derived more rapidly than without exploiting such knowledge.

5 Discussion

We have proposed an agent decision-making mechanism where models of other agents are refined through argumentation-derived evidence from past dialogues, and these models are used to guide future task delegation. Our evaluations show that accurate models of others' policies could be learned by exploiting domain knowledge. We believe that this research contributes both to the understanding of argumentation strategy for dialogue among autonomous agents, and to applications of these techniques in agent support for human decision-making. Sycara *et al.* [6] report on how software agents can effectively support human teams in complex collaborative planning activities. One area of support that was identified as important in this context is guidance in making policy-compliant decisions. This prior research focuses on giving guidance to humans regarding their own policies. Our work complements the approach of Sycara *et al.* by allowing agents to support humans in developing models of others' policies and using these in decision making. Our approach extends decision trees with ontological reasoning. Zhang and Honavar have also extended C4.5 decision trees with Attribute-value taxonomies [8]. Their approach is similar to *STree*, but it does not allow ontological reasoning during tree induction. Unlike their approach, our approach can directly incorporate existing domain ontologies and exploits these ontologies during policy learning.

In future, we plan to extend other machine learning methods with domain knowledge and explore how much this extension improves policy learning and enhances agents' support for human decision-making.

Acknowledgements. This research was sponsored by the U.S. Army Research Laboratory and the U.K. Ministry of Defence and was accomplished under Agreement Number W911NF-06-3-0001. The views and conclusions contained in this document are those of the author(s) and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Army Research Laboratory, the U.S. Government, the U.K. Ministry of Defence or the U.K. Government. The U.S. and U.K. Governments are authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation hereon.

References

1. Cao, L., Gorodetsky, V., Mitkas, P.: Agent mining: The synergy of agents and data mining. *IEEE Intelligent Systems* 24(3), 64–72 (2009)
2. Emele, C.D., Norman, T.J., Parsons, S.: Argumentation strategies for plan resourcing. In: Proceedings of AAMAS 2011, Taipei, Taiwan (to appear, 2011)
3. Hitzler, P., Krötzsch, M., Rudolph, S.: Foundations of Semantic Web Technologies. Chapman & Hall/CRC (2009)
4. McBurney, P., Parsons, S.: Games that agents play: A formal framework for dialogues between autonomous agents. *Journal of Logic, Language and Information* 12(2), 315–334 (2002)
5. Norman, T.J., Reed, C.A.: Delegation and Responsibility. In: Castelfranchi, C., Lespérance, Y. (eds.) ATAL 2000. LNCS (LNAI), vol. 1986, pp. 136–149. Springer, Heidelberg (2001)
6. Sycara, K., Norman, T.J., Giampapa, J.A., Kollingbaum, M.J., Burnett, C., Masato, D., McCallum, M., Strub, M.H.: Agent support for policy-driven collaborative mission planning. *The Computer Journal* 53(1), 528–540 (2009)
7. Witten, I.H., Frank, E.: Data Mining: Practical machine learning tools and techniques, 2nd edn. Morgan Kaufmann, San Francisco (2005)
8. Zhang, J., Honavar, V.: Learning decision tree classifiers from attribute value taxonomies and partially specified data. In: Proceedings of the International Conference on Machine Learning (2003)

Data Mining to Support Human-Machine Dialogue for Autonomous Agents

Susan L. Epstein¹, Rebecca Passonneau², Tiziana Ligorio¹, and Joshua Gordon³

¹ Hunter College and The Graduate Center of The City University of New York,
Department of Computer Science, New York, NY

susan.epstein@hunter.cuny.edu, tligorio@gc.cuny.edu

² Center for Computational Learning Systems, Columbia University, New York, NY
becky@cs.columbia.edu

³ Department of Computer Science, Columbia University, New York, NY
joshua@cs.columbia.edu

Abstract. Next-generation autonomous agents will be expected to converse with people to achieve their mutual goals. Human-machine dialogue, however, is challenged by noisy acoustic data, and by people’s preference for more natural interaction. This paper describes an ambitious project that embeds human subjects in a spoken dialogue system. It collects a rich and novel data set, including spoken dialogue, human behavior, and system features. During data collection, subjects were restricted to the same databases, action choices, and noisy automated speech recognition output as a spoken dialogue system. This paper mines that data to learn how people manage the problems that arise during dialogue under such restrictions. Two different approaches to successful, goal-directed dialogue are identified this way, from which supervised learning can predict appropriate dialogue choices. The resultant models can then be incorporated into an autonomous agent that seeks to assist its user.

Keywords: spoken dialogue systems, Wizard of Oz, human-machine interaction.

1 Introduction

A *spoken dialogue system (SDS)* is an autonomous agent that communicates with people in the way most natural to them — through spoken language. In pursuit of a common goal, however, such an agent must not only speak to the person, but also listen. People want human-machine dialogue to be both *successful* (accomplish their goal) and *habitable* (demonstrate people’s tacit knowledge about how dialogue should be conducted). To that end, this paper studies how one person manages to help another achieve a goal during interactions that simulate dialogue between a human and an autonomous agent. The primary result reported here is the identification of two different strategies for effective, goal-directed dialogues. One is service oriented, and the other is data driven. Each could serve as a model for an autonomous agent engaged in goal-directed human-machine dialogue.

This paper describes the collection and mining of a rich corpus in an ambitious domain. The corpus describes dialogues between two people: a person (here, the *caller*) and a *wizard*, whom the caller believes to be a computer system but is actually another human. The wizard, however, is *ablated*, that is, her input and permitted actions are restricted to those that would be available to an autonomous agent [1]. She is also *embedded* [2], that is, the wizard and the system process some of the same inputs concurrently, so that run-time system features can be mined to model the wizard's actions. Thus the wizard experiences dialogue much the same way that an SDS would — she accepts automated transcriptions of human speech as input and generates speech as output.

We emphasize that the wizard is not intended to be a model of an expert system, but rather a model of how to solve problems that arise during dialogue. Therefore the wizard need not be quick or even perfectly accurate. Rather, the wizard should try to understand what the caller requires from what the caller says, and help achieve that goal. Because automated speech recognition for arbitrary speakers across a large vocabulary is extremely difficult, however, the wizard must actually try to understand what the caller requires from a noisy transcription of what the caller has said.

The assembled corpus includes caller speech, wizard behavior, and a broad spectrum of descriptive features available to a traditional SDS. From it, we identify the most expert wizards, and examine how they made their decisions. The next section of this paper provides details on our domain of investigation. Subsequent sections discuss related work, and summarize the preliminary experiments that supported the approach in a full dialogue experiment. The paper then details the design and results of that experiment, and discusses them and current work.

2 Speaking with the Librarian

Our real-world domain presents considerable challenges, but also provides an extensive supporting knowledge base. We investigate book requests to the Andrew Heiskell Braille and Talking Book Library, a branch of the New York Public Library and a Regional Library of the National Library Service for the Blind and Physically Handicapped of the Library of Congress. Most of Heiskell's holdings are books recorded in a proprietary format and mailed to its patrons on request. Heiskell's patrons receive a monthly newsletter that describes the newest titles, along with their authors and catalog numbers.

Patrons order books from a Heiskell librarian by telephone. As they speak with the patron, Heiskell's librarians search their *book database* of titles and authors, to identify what they believe the patron wants. As of 2007, there were only 5 librarians to serve 5,028 active patrons, many of whom order books several times each week. At that time, there were no plans to increase the staff, and even the recording mechanism for off-hours calls was at capacity. Clearly an autonomous agent that could handle the simplest of these calls would be of great value, both to Heiskell's patrons and to those of other libraries that provide a similar service.

CheckItOut is an SDS that accepts mock “book orders” from callers on its dedicated *VOIP* (Voice Over Internet Protocol) line. During a call, *CheckItOut* introduces itself; identifies the caller; handles up to four book requests by title, author, or catalog number; summarizes the order if the caller wishes; and signs off. *CheckItOut* has available to it the 2007 versions of Heiskell’s full book and transaction databases, and a sanitized version of Heiskell’s database on its active patrons. Only the catalog number is guaranteed to identify a book uniquely.

An SDS receives a continuous stream of acoustic data that includes background noise as well as human speech. The SDS relies on *ASR* (Automated Speech Recognition) to translate that input into one or more *recognition hypotheses* (sequences of words, or text strings). *CheckItOut*’s speech recognition is intentionally not state-of-the-art, because we seek to develop interpretation methods and dialogue strategies that are robust to noisy ASR. During dialogue with a caller, *CheckItOut* must contend with a large vocabulary (54,448 distinct words alone from 71,166 titles and 28,031 authors), a varied speaker population, and callers in environments with background noise, all of which make ASR considerably more difficult. Table 1 provides some examples of noise-ridden ASR output for spoken book titles.

Table 1. Book titles and their noisy ASR output

	Title	ASR OUTPUT
1	<i>Into the Night</i>	Into than 9
2	<i>Helen and Teacher: The Story of Helen Keller and Anne Sullivan Macy</i>	Helen an teacher distort tell until an am Sullivan Macy
3	<i>Map of Bones</i>	Nah don’t bones
4	<i>I Lived to Tell it All</i>	Elusive total man

Clearly, noisy ASR is likely to produce both misunderstandings (incorrect semantic interpretations) and *non-understandings*, where the system cannot interpret the ASR at all. There is signal within the noise, however, and our previous work shows that people identify it well, given sufficient context. We seek to develop an SDS that does so too.

3 Related Work

3.1 Spoken Dialogue Systems

CheckItOut is built within a traditional *pipeline architecture* for SDSs, shown in Figure 1. In this architecture, an *audio manager* analyzes the incoming acoustic stream for voice activity, and forwards a sequence of overlapping audio frames to an *interaction manager*. The interaction manager (*IM*) performs an initial segmentation (determines utterance boundaries) that it sends to the ASR module. The ASR module forwards its recognition hypotheses (as orthographic text strings) to a natural language understanding (*NLU*) module. The NLU produces one or more parses, each of which is a mapping from ASR to concepts, and then a confidence annotator identifies the best parse.

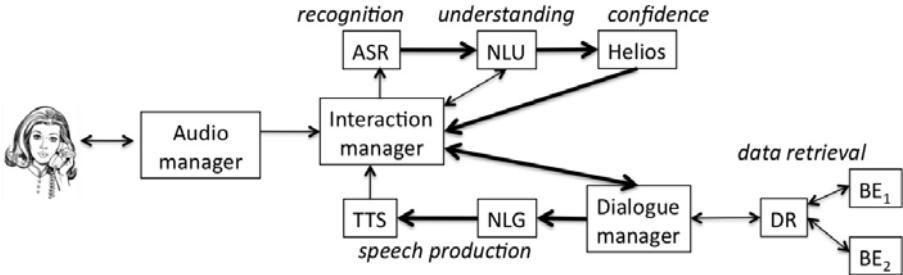


Fig. 1. A traditional pipeline architecture for spoken dialogue systems. Heavy arrows emphasize the pipeline

The focus of our work here is the decision maker: the dialogue manager and its relationship to the four modules (IM, ASR, NLU, and confidence annotator) involved in spoken language understanding. (Ultimately, the IM uses input from the NLU, the confidence annotator, and the dialogue manager to decide where the next utterance boundary lies, and thereby to which parsed concepts the dialogue manager should respond.) The dialogue manager decides what action to take next. It can use the user-designed, application-specific *domain reasoner* (*DR*) to query the knowledge stored in one or more backends (*BE*), or it can decide to speak. A decision to speak is forwarded to the natural language generator (*NLG*), which formulates text and then passes it to a text-to-speech (*TTS*) module. Finally the speech output is directed to the IM, which transmits it to the caller. Each such utterance is referred to here as a *system prompt*.

The *Olympus/RavenClaw* pipeline architecture [3, 4] has been the framework for at least a dozen SDSs. The components in an Olympus/Ravenclaw SDS communicate through the Galaxy hub [5]. Olympus/Ravenclaw supports a variety of modules for each of its components. CheckItOut, our SDS for the simulated Heiskell library world, uses the *Apollo* interaction manager [6, 7] and the *PocketSphinx* speech recognizer [8]. For our domain we adapted freely available acoustic models of Wall Street Journal dictation speech with about eight hours of spontaneous speech. CheckItOut also uses the *Phoenix* context-free grammar semantic parser [9], the *Helios* utterance-level confidence annotator [10], and the *Kalliope/Swift* TTS [11]. CheckItOut's dialogue manager is built within *RavenClaw*, which separately defines domain-dependent conversational goals in an explicit task hierarchy and provides domain-independent error-handling strategies.

3.2 Challenges in Spoken Dialogue

In commercial SDSs intended for many callers, accurate speech recognition performance is achieved at the price of a system that generates an inflexible sequence of prompts, has limited strategies to address communication difficulties, and handles very small subsets of language. These factors are partially responsible for the frustrating telephone conversations people often experience with them. Speech

recognizers rely on pre-existing acoustic models to relate acoustic energy to speech sounds, and on trained, domain-specific language models to predict which sequences of speech sounds correspond to known words. Although large-vocabulary ASR has improved dramatically for single-party applications, such as the transcription of broadcast news, its accuracy in dialogue lags substantially.

To limit communication errors incurred by faulty ASR, an SDS may use enriched strategies to detect and respond to incorrect recognition output [12]. It may repeatedly request caller confirmation to avoid misunderstanding. This confirmation may be either *explicit* (e.g., “I heard you say *Grapes of Wrath*. Is that correct?”) or *implicit* (e.g., “By Steinbeck. That’s available.”). Implicit confirmation uses language that elicits responses from the caller that the system can handle [6]. If the caller adds new information in response to a system prompt, two-pass recognition can consider the extra information contained in that response to restrict the language expected in the second pass and thereby achieve better recognition [13].

Despite careful engineering in the laboratory, ASR performance in fielded research dialogue systems commonly has a word error rate (*WER*) at best near 30%-35% and as high as 70% [4]. One way to minimize both misunderstandings and non-understandings despite high WER is with accurate semantic interpretation. The need to correct the system’s misunderstandings, however, can frustrate the caller, and can elicit speech that is more poorly recognized than non-correction utterances [14]. When an SDS re-prompts the caller for the same information after non-understanding, it often fails to understand because the caller then hyperarticulates, which generally results in even worse recognition.

Another source of caller frustration during human-computer dialogue is that the SDS maintains *system initiative*, that is, it alone controls the path of the dialogue. This results in a rigid sequence of spoken commands to the caller. In contrast, a *mixed-initiative* SDS permits the caller to volunteer information, to interrupt or to correct the system when it fails to understand the caller’s intent. Once a system shares the initiative with the caller, however, it risks confusion. RavenClaw’s dialogue task tree offers a way for the SDS to remember and anticipate what is under discussion. Thus, if a CheckItOut caller signals that the system has misunderstood her with “That’s not what I said,” CheckItOut will remove the incorrect book from the order. The system also uses a subset of RavenClaw’s domain-independent error-handling strategies. Among these is one that, after three or four consecutive non-understandings, has the SDS abandon the call, that is, simply hang up.

3.3 Wizardry in Human-machine Dialogue

During dialogue, people may fail to understand or may misunderstand one another. To avoid frustration, people often use creative ways to re-elicit needed information — they use contextual clues to fill in gaps, and to confirm that some communication has occurred. We seek to exploit for an autonomous agent a similar repertoire of responses to support habitable and successful dialogue.

To acquire knowledge about human strategies that would support robust interpretation of noisy ASR, we use a *Wizard of Oz* study, where, unbeknownst to the human caller, another person (the wizard) plays the role of the system. Wizard of Oz studies were originally intended to elicit caller behavior, so that system designers

could anticipate likely system input [15]. They have also been used to study how people make decisions when given the same input and set of actions that would be available to a system. Figure 2 portrays how a wizard can be embedded in an SDS, with the wizard's dataflow indicated by heavy arrows.

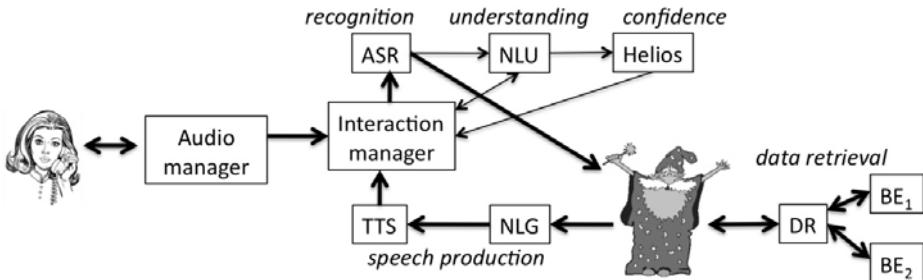


Fig. 2. A wizard embedded in CheckItOut. Heavy arrows indicate the dataflow for the wizard

Other Wizard of Oz studies that focused on the wizard during full spoken dialogues have included efforts to predict the wizard's response when the caller is not understood [12], the wizard's use of multimodal clarification strategies [16], and the wizard's use of application-specific clarification strategies [17]. The full dialogue experiment presented in Section 5 differs from that work in three ways (detailed in our earlier work on partial dialogue [2]): it analyzes several wizards' behavior, it recognizes differences among wizards, and it identifies distinctive and successful behavior, so that the SDS will ultimately benefit only from models of the most skilled wizards.

Wizards interpret noisy ASR much better than machines do, because they know which aspects of context are relevant. Given real or simulated ASR output instead of the caller's speech during dialogue, human subjects engage in problem solving about the task, the ASR errors, or both [18]. One particularly useful technique is *voice search*, where the wizard directly queries a knowledge base with ASR output, and receives returns ranked by a similarity score [19]. Voice search provides context to seemingly unintelligible ASR output. For example, SOONER SHEEP MOST DIE is readily resolved in the context of the candidate book title matches *Soon She Must Die*, *Why Someone Had to Die*, and *The Messenger Must Die* [2].

4 Preparatory Experiments

This section summarizes several experiments, first reported elsewhere, that were performed in preparation for the full dialogue experiment in Section 5. They demonstrated that CheckItOut's task is more difficult than that of a well-known, fielded system, and that people acting as ablated wizards are surprisingly good at CheckItOut's book order task.

4.1 Task Exploration

Caller requests for books by title are more challenging than those by catalog number or author, because the book title field is much more like free text than like structured data, and thus much less predictable. Under four WERs (from perfect to very poor), we compared how difficult it was for the Phoenix parser to identify the request type of callers' utterances in two Olympus/RavenClaw systems [20]. We parsed requests for books by title and by author from CheckItOut, and requests for bus information by route and by destination place name from Let's Go Public! [4]. Caller utterances in CheckItOut proved longer and more challenging at every level of WER. For example, at the same WER of 0.4, concept accuracy was about 74% for Let's Go Public! but only about 36% for CheckItOut. Thus the semantic parsing approach used by CheckItOut and Let's Go Public! has more difficulty understanding how to interpret input to CheckItOut as a request type. In contrast, an alternative machine-learning approach to NLU performed much better on the longer utterances in CheckItOut (85%) than the short utterances of Let's Go Public! (36%).

In an offline *pilot experiment* of requests for books by title, we also demonstrated that people were remarkably effective at matching error-ridden ASR strings to items in a large text file. We trained the language model for PocketSphinx on a random sample of 500 Heiskell book titles, which contained 1,400 distinct words. Each of the three participants then received, in text format, CheckItOut's ASR output from a single speaker on 50 randomly-chosen titles from those 500. (ASR output quality was even poorer than in Table 1. WER ranged from 0.69 to 0.83, depending upon the speaker.) Participants were given the frequency with which each individual word occurred in the 500 titles, the borrowing frequency for each book, and a text file of all 71,166 book titles. Participants were asked to identify or guess the title in each instance, and were permitted to search the text file in any way they chose, with no time limit. Although the ASR rendered only 9% of all 150 titles perfectly, participants identified a startling 61.7 – 71.7% of their books correctly. Subjects were also asked to explain how they made their decisions. See [21] for further details.

4.2 The Book Title Experiment

People's ability to make sense of poor ASR output in the pilot experiment motivated a large-scale Wizard of Oz *book title experiment*. In our domain, requests for books by title pose the greatest challenge. Thus our initial ablated wizard study had multiple wizards of varying expertise address partial dialogues consisting of a single *turn exchange*: the caller spoke a title and in response the wizard either offered a book title or asked a question. With 7 subjects who played both roles, we collected 4,172 book title requests with poor ASR between all possible caller-wizard pairs.

The caller and the wizard each had a graphical user interface (*GUI*) and a microphone. When the caller read a book title into a speech recognizer through her microphone, the corresponding ASR appeared on the wizard's GUI, where the wizard formulated a query from the ASR for the database. Given the results of the query, the wizard then indicated on her GUI what she believed to be the correct book, asked

through her microphone a question that she believed would help identify the correct book, or indicated on her GUI that she gave up. The caller scored any wizard-identified title as correct or incorrect, and that score was displayed on the wizard's GUI. (The caller also scored any questions, but that information was not shared with the wizard.) All 7 wizards could identify correct matches returned by their query, with individual accuracy that ranged from 69.5% to 85.5%. For further details on the mechanics of this process, how we created a dialogue-like environment, and how we encouraged the best possible performance from our subjects, see [2, 22, 23].

During this experiment, we collected data on 60 features available at run time. These features were motivated by comments from the subjects in the pilot experiment on how they matched titles to the database; we identified those likely relevant to dialogue management. System features, most of which were unavailable to the wizard, described the speech signal, the ASR output, the recognition process, the ability of the SDS to interpret the ASR string, and two Olympus/Ravenclaw confidence scores that combine recognition with language understanding. Other features described the session history (e.g., number of correctly identified titles so far), the database return (e.g., number of returned titles), and similarities between the ASR string and the returned titles (e.g., number of matching words).

Although wizards regularly detected correct matches, few responded appropriately when the correct title was not returned by the query. In that case (28.43% of all turn exchanges), the wizard should have asked a question. Despite careful instructions to the contrary, all wizards did so in only 22.32% of those situations. Only the two most accurate (85.50% and 81.33%) wizards recognized when none of the database returns was a likely match; they asked questions in 64% and 43% of these situations, respectively. The next most frequent questioner asked only 20% of the time. Clearly, recognition that no match is present is important to accuracy in this task.

We used linear regression, logistic regression, and decision trees on the features monitored during the book title experiment to model the behavior of each individual wizard and of the wizards as a group. For the all-wizard models, logistic regression had 75.2% accuracy, and decision trees 82.2%. Linear regression had root mean squared error 0.483, and decision trees had 0.306. The predictive ability of the models for individual wizards was similar. The features our best wizards used, their propensity to ask questions, and the kinds of questions they asked provided further guidance for the full dialogue experiment recounted in Section 5.

4.3 A Baseline for the Wizard Dialogues

To establish a baseline in CheckItOut for comparison of human-wizard dialogues with human-system dialogues, 562 dialogues were collected, with 10 caller subjects, 5 male and 5 female. Each caller made 50 book-order calls to CheckItOut over three days. For each call, the subject received (from a web site) a new *scenario*: a patron identity, a list of four books, and instructions to request, in any order, one book by catalog number, one by title, one by author, and one by any of those methods.

CheckItOut's domain reasoner performs a partial matching against the database on the words in the parse in which Helios had the most confidence. This procedure uses Ratcliff/Obershelp pattern recognition (*R/O*) [24] to evaluate the similarity of a submitted string to a book title, author, or catalog number in the database. The *R/O*

score is the number of matching characters divided by the total number of characters. For example, for ROLL DWELL the three top-candidate titles and their R/O scores were *Cromwell* (0.666), *Colin Powell* (0.636), and *Robert Lowell* (0.608). On average during the baseline experiment CheckItOut identified 2.39 books correctly on each call (range 0 – 4, standard deviation $\sigma = 1.3$).

5 Experimental Design

The *full dialogue experiment* described in this section serves as a paradigm for the development of dialogue skills for autonomous agents. It places wizards in a more challenging environment, one almost identical to that of an autonomous dialogue agent, except that instead of responding promptly to callers' turns, wizards could take time to problem solve. Voice transmission was by telephone, and book requests were by title, author, or catalog number. Caller and system participated in a dialogue, complete with *disfluencies* (e.g., pauses, repetitions, and self-corrections), corrections ("That's not what I said"), and subtasks (e.g., identification of the caller, task summary). People were thereby freed to pursue the same conversational goals by creatively different means.

5.1 Subjects and Preparation

Both wizards and callers were recruited by email and flyers to students at three local universities. We solicited volunteers for the (more difficult and more remunerative) role of wizard. A single trainer instructed four male and five female wizard trainees, one at a time. To familiarize them with the custom database query (described in Section 4.2), trainees were given 24 ASR strings with 5 candidate search results from the book title experiment, and asked to select which, if any, of the search results matched the ASR. Then each trainee was given a visual and verbal description of a new wizard GUI. (The wizard's GUI in the book title experiment offered fewer choices and less information. The new one is described in Section 5.2 below.) The trainee watched the trainer perform as wizard on a sample call. Finally, each trainee made five test calls during which she could ask questions and talk to the trainer. We then chose as wizards those trainees who were most motivated and skilled at the task. Trainees who were not selected as wizards did not participate further in the experiment.

Callers assumed that they were speaking to an SDS. The lack of real-time response was explained by telling callers that the system was highly experimental, and would be developing and rejecting many hypotheses before responding. They were forewarned that the calls would be long and frustrating. Although wizards were instructed to wrap calls up after six minutes, callers were not permitted to terminate any call. Each caller also made five training calls, during which she could question the trainer by chat.

5.2 Software

Phoenix is a semantic parser that typically does not rely on syntactic structure, a feature that makes it more robust to ASR noise. A Phoenix parse is a semantic frame consisting of a set of concepts; each concept has its own context-free-grammar (*CFG*). For example, a phone number parse has an area code concept and a concept for the remainder of the phone number. Noisy tokens in an ASR string can be skipped between frames or between concepts, and wild cards can be used sparingly to handle noise within concepts. Such a grammar can handle concepts like phone numbers that have a small vocabulary (digits) and a fixed length, but does not extend well to concepts like book titles with large vocabularies and varying length. To produce rules for the Phoenix book title *CFG* that preserved its robustness to noise while adding syntactic information, we wrote a transducer to produce Phoenix rules from MICA parses of book titles. (MICA is a broad-coverage dependency parser [25].) Before this experiment, we produced Phoenix rules this way for 3,000 books randomly selected from the 71,166 in Heiskell’s database. We then constructed a web page that, on demand, randomly generated a scenario with patron identity (telephone number, name, and address) plus a list of four books randomly selected from those 3,000 titles. As in the baseline experiment, the scenario provided the title, author, and catalog number for each of the four books.

As in Figure 2, the *wizard version* of CheckItOut that answered the telephone included all the modules in Figure 1 and the customized query and matching facilities described in Sections 4.2 and 4.3. In the wizard version, however, the wizard replaced the dialogue manager. Although the Phoenix parser and the Helios confidence annotator did not participate in decisions or select ASR strings for queries, features that described their output and performance were captured for data mining.

A wizard interacted with a caller through two similarly-organized GUIs, one for the login and the other (shown in Figure 3) for the book requests. (This paper focuses on the latter, a considerably more complex task.) In Figure 3, the two frames at the top contain scrollable output from the ASR for the entire dialogue, and from the book database. The material in the two center frames provides the wizard with a dialogue history and sets of basic and auxiliary actions (described further in Section 5.4). The dialogue-history frame in Figure 3 displays how many books have been ordered in the call thus far, their titles, how many questions the wizard has asked, and how often she has asked the caller to repeat. The four frames at the bottom offer *clarifications*, prompts intended to advance the dialogue when the wizard cannot formulate a query or cannot match a book to the current ASR.

The clock in the upper left in Figure 3 changed color after 6 minutes. Wizards were instructed to complete the current book request at that point if it were almost identified, and then end the call, even if all four books had not yet been ordered. This reflects both our desire to minimize the caller’s frustration, and our focus on problem solving behavior rather than full task completion.

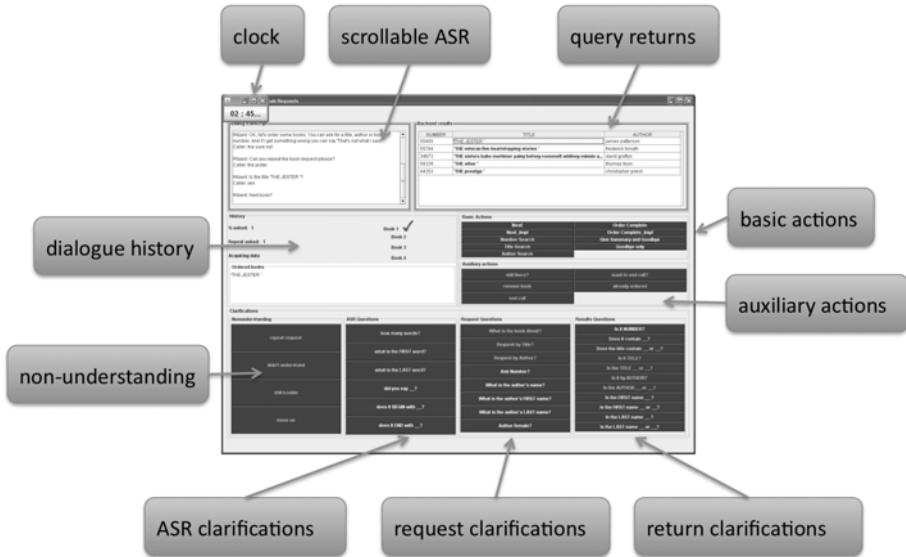


Fig. 3. Annotated screen shot of the wizard GUI during dialogue

5.3 The Task

Before each call, the caller accessed the scenario web page for an identity and a list of books. The caller was instructed to first identify herself during the *login* for patron identification, and then *request* (orally ask for) the four books in any order: one by title, one by author, one by catalog number, and the fourth by any of those request types. In the full book database, 35% of the titles contain a *subtitle* (an extra phrase that follows the title and is separated from it by a colon). In the random sample of titles covered by the scenarios for this experiment, 39% contained subtitles. On any given title request, callers spoke the title, the subtitle, or both.

Callers used a telephone of their choice (either cellular phone or landline) to call the wizard version of CheckItOut, which answered each call with its customary greeting. Then the wizard took over. The wizard responded only by selection of a pre-specified action on the GUI. Any action that involved speech to the user was transmitted to the NLG. Unless awaiting the caller's response to a system prompt, the wizard was expected to select some sequence of actions and clarifications to advance the dialogue.

5.4 How Wizards Conducted Dialogues

The basic actions on the GUI are either database queries or prompts that the wizard can use to advance the dialogue. Figure 4, drawn from the corpus, shows how a proficient wizard used clarifications to decipher error-ridden ASR. For reference here, lines are numbered, and W denotes speech generated when the wizard clicked a selection on the GUI.

Basic actions on the GUI allow the wizard to request the next book (with or without implicit confirmation of the book just ordered), inform the caller that the order is complete (explicitly or implicitly), offer the caller an optional summary of the order, or say goodbye. Line 1 in Figure 4, for example, is a basic action used by the wizard to obtain a new book request from the caller. C denotes ASR output based on speech from the caller. Although W lines and C lines for the most part alternate, as one would expect in a dialogue, note that lines 6 and 7 represent ASR output for what the system perceived as two consecutive utterances from the caller.

- 1 W: Next book?
- 2 C: .PAGAN. .GEORGE. AMERICA
- 3 W: Did you ask for a title?
- 4 C: YES
- 5 W: Can you repeat the book request please?
- 6 C: .PAGAN. .TURNER. DIABETES
- 7 C: DIDN'T I AM DONE
- 8 W: Did you speak the word "diabetes"?
- 9 C: YES
- 10 W: Is the title *Conquering Diabetes: A Cutting Edge Comprehensive Program for Prevention and Treatment?*
- 11 C: NO
- 12 W: What is the author's last name?
- 13 C: .STARDANCE. SERVICES
- 14 W: Is the title *Controle Su Diabetes Guia Para El Ciudadado De Su Salud: Take Charge Of Your Diabetes A Guide For Care?*
- 15 C: YES
- 16 W: OK, we now have four books. Would you like a summary of your order?

Fig. 4. Part of a wizard dialogue for one book request. Italics provided for clarity. All caller (C) language is ASR output; all wizard (W) language is generated by the system in response to a command from the wizard

The book the scenario had assigned to the caller in Figure 4 was *Controle Su Diabetes Guia Para El Ciudadado De Su Salud: Take Charge Of Your Diabetes A Guide For Care* whose author the database records as "US Dept of Health and Human Services." The ASR suggests that the caller ignored the Spanish title and read only the English. When a word is bracketed by periods (e.g., .PAGAN. in line 2) it denotes a low confidence score from the ASR. Line 2 offered little information, but the wizard suspected it was a title, which the caller confirmed, so the wizard asked for the title again (lines 3-5). The quality of the ASR output did not improve, but this time it included an unusual and confident word (DIABETES).

Some clarifications indicate to the caller that the system has not understood the last ASR output (request to repeat, explicit statement of non-understanding, repeated statement of non-understanding, decision to go on to the next book). Others ask about what the wizard saw in the ASR (e.g., "How many words?"), about words in the ASR (e.g., "Did you say __?"), about words in the search returns (e.g., "Does it begin with

_), or about the book request itself (e.g., “Did you ask for a book title?”). Still other clarifications might elicit a change in request type (e.g., “What is the author’s name?”) or allow the wizard to select from elements of the search results (e.g., “Is the book title __?”). In Figure 4, the wizard chose to clarify that the caller had indeed said DIABETES (lines 8 – 9), and entirely ignored line 7.

The three database-query basic actions (by title, author, or catalog number) allow the wizard to do voice search. If any such action is selected, an additional small display (not shown) provides only the ASR output from the caller’s responses for the current book request. To formulate the query, the wizard selects words from the small display with a mouse. The customized query then performs partial matching on the submitted ASR string against the book database. Wizards knew that database returns appeared in order of decreasing similarity, but the computed R/O scores did not appear on the GUI. A search by title or catalog number returned the top-scoring five books (title, author, and catalog number). A search by author returned up to 3 candidates each for up to 5 matching authors, a maximum of 15 search results. After line 9 in Figure 4, the wizard did a title search on PAGAN TURNER DIABETES. The five returns from the query, in descending R/O score order, were

Conquering Diabetes: A Cutting Edge Comprehensive Program for Prevention and Treatment? by Anne L Peters

Coping With Diabetes by Pat Kelly

Controle Su Diabetes Guia Para El Ciudadado De Su Salud: Take Charge Of Your Diabetes A Guide For Care? by US Dept of Health and Human Services

I'm Tougher than Diabetes by Alden R Carter

The Other Diabetes: Living and Eating Well with Type 2 Diabetes by Elizabeth Hiser

From among these returns, the wizard incorrectly guessed the first one, and then asked for the author (lines 10 – 12). One confident word (SERVICES) in the ASR output was now enough to identify the book (lines 13–15). Finally, in line 16, the wizard selected a basic action.

Auxiliary actions were also provided. They allowed the wizard to remove a book from the order, to end the call, to ask if the caller was still there, to ask if the caller wanted to end the call, and to inform the caller that the book just requested had already been ordered during the current call.

Even with fully accurate ASR, to provide a careful match a wizard may use clarifications to choose judiciously among search results, as in Figure 5. In the first dialogue fragment (lines labeled A), the wizard did an author search on NANCY WHITELAW, but both the top two returns (out of five) were by Nancy Whitelaw. The wizard then asked for the catalogue number, which matched one of them, *They Wrote Their Own Headlines: American Women Journalists*. The second dialogue fragment (lines labeled B), shows that even when a clarification introduces errors, it can be of help. The wizard did an author search on JOHN EGERTON, but again the top two returns had that author. Rather than disambiguate between a 15-word and a 9-word title, the wizard again requested the RC number. Although the ASR output was on two lines

and ROHAN is not a digit, *Speak Now Against The Day: The Generation Before The Civil Rights Movement In The South* has catalogue number 40546, so the four good digits sufficed to identify the book.

- | | |
|-------|---|
| A1 | W: Next book? |
| A2 | C: NANCY WHITELAW |
| A3 | W: Can I have the RC number? |
| A4 | C: THREE EIGHT NINE EIGHT ZERO |
| A5 | W: Is the title <i>They Wrote Their Own Headlines: American Women Journalists</i> ? |
| A6 | C: YES |
| A7 | W: Would you like <i>They Wrote Their Own Headlines: American Women Journalists</i> by Nancy Whitelaw? |
| A8 | C: YES |
| <hr/> | |
| B2 | W: Next book? |
| B2 | C: JOHN EGERTON |
| B3 | W: Can I have the RC number? |
| B4 | C: FOUR |
| B5 | C: .ROHAN. FIVE FOUR SIX |
| B6 | W: Is the title <i>Speak Now Against the Day: The Generation before the Civil Rights Movement in the South</i> ? |
| B7 | C: YES |
| B8 | W: Would you like <i>Speak Now Against the Day: The Generation before the Civil Rights Movement in the South</i> by John Egerton? |
| B9 | C: YES |
| B10 | W: Next book? |

Fig. 5. Author and catalog number requests in two dialogue fragments. Both lead the wizard to the correct book, even though the second includes an ASR error

During the experiment, the trainer remained in the room with the wizard, and could communicate with the caller via chat to coordinate breaks between calls and to restart the system if necessary. This facilitated the complex wizard-caller pair scheduling and dealt with any unforeseen difficulties. On the rare occasion of a system crash, the call was not preserved.

5.5 Data Collection

The knowledge acquired from the work reported here is ultimately intended to support the construction of an improved dialogue manager. The decisions any dialogue manager makes must be based on data available to the SDS at run-time. Therefore we collected 163 run-time features that describe the system, the caller's speech and the wizard's behavior, and characterize the dialogue:

- Features that describe each call, each caller utterance, and each *adjacency pair* (portion of a dialogue that began with a system prompt and ended just before the next system prompt [26]). An adjacency pair contains one or more caller utterances and zero or more database searches.
- Features that describe the input and output of CheckItOut’s ASR, natural language understanding, and confidence annotation modules.
- Features that describe the wizard’s behavior, including her clickstream and queries.
- Features that describe dialogue cost, such as correct book identification and frequencies of non-understanding and misunderstanding.

Surveys were administered automatically to wizards on their first, 60th, and 120th calls. The survey allowed the wizards to report on their ease with and progress on the task, and elicited information about their strategies. Callers also completed surveys after their 15th, 30th, 60th, and 90th calls.

Table 2. Selected statistics from the full dialogue experiment. (Despite instructions to correct the system with “That’s not what I said,” on two calls five books were ordered)

<i>Dialogues</i>	μ	range	σ
Ordered books	2.45	0 – 5	1.44
Correctly identified books	2.26	0 – 5	1.45
Utterances per call	22.36	4 – 40	5.06
Words per utterance	2.99	1 – 10	2.27
Queries per adjacency pair	1.09	1 – 6	0.33
Questions from wizard to caller per book request	3.41	0 – 9	2.49
Title length in words	5.96	1 – 34	4.38
Adjacency pairs with at least one database query	32%		
Fully successful calls	28%		
Failed calls (no books ordered)	17%		
<i>Voice searches</i>	By title	By author	By catalog number
Number	43%	31%	26%
Return includes correct book	28%	33%	58%

6 Results

6.1 The Dialogues

Ten callers (5 male, 5 female) each made 15 calls to each of 6 wizards (3 male, 3 female), for a targeted total of 900 calls. (We actually collected 913 dialogues due to the exigencies of data collection.) Each dialogue addressed the task posed in the scenario: to assume an identity and then order four books. The dialogues covered

3,394 book requests in all, and 20,415 caller utterances. There were 17,288 adjacency pairs, 32% of which contained at least one database query. A sample of the ASR indicated a WER of about 50%. Table 2 further summarizes the corpus.

Most important was that the wizards, despite their ablation, understood what the callers wanted. For the most part, wizards identified the books in the scenario: 92% of all ordered books (2.26 of the 2.45 per call) had actually been requested by the callers. Because they operated under the 6-minute time limit, wizards terminated 63% of all calls. Of the non-terminated calls, 76% were *fully successful* (all 4 books correctly identified and ordered) despite a WER of about 50%. This result confirms that wizards were able to identify the signal within the noise through context and appropriate interaction strategies.

Table 3. Comparative wizard performance. WA and WB were the two most proficient wizards; WE and WD were the least proficient

<i>Question type</i>	<i>All wizards</i>	<i>WA</i>	<i>WB</i>	<i>WD</i>	<i>WE</i>
Signal non-understanding	37%	34%	42%	40%	33%
On ASR string	8%	2%	0%	12%	15%
On query results	36%	37%	40%	26%	32%
On requests	19%	27%	18%	22%	20%
Total	11562	2321	1540	2013	1868
<i>Actions per call</i>					
Question before any search	0.35	0.29	0.21	0.67	0.42
Explicit confirmations	6.07	6.76	4.18	5.32	6.59
Implicit confirmations	0.40	0.62	0.68	0.20	0.86
Move-on strategy	0.67	0.39	1.19	0.65	0.43
<i>Actions per book request</i>					
Database searches	1.77	2.10	1.72	1.70	1.70
Questions	3.41	4.09	2.28	3.68	3.90
Confirmations	1.74	2.05	1.10	1.56	2.36
<i>Call statistics</i>					
Fully successful calls	28%	33%	32%	24%	16%
Failed calls (no books ordered)	17%	7%	11%	16%	24%
Correct titles	2.26	2.69	2.54	2.05	1.09

Each dialogue represented considerable interaction with the caller (22.36 caller utterances). Ultimately, an autonomous agent that relies on strategies learned from this corpus should achieve the same success, yet at far lower cost (e.g., number of turns). Note too that the book titles in the scenarios were often considerably longer than what a traditional SDS elicits from its callers — the titles averaged about 6 words but ranged as high as 34. As one would expect, returns from catalog number queries to the database more often (58%) contained the requested book than queries

by author (33%,) or title (28%). In the remainder of this paper, any cited difference is significant on a paired *t*-test at the 95% confidence level.

6.2 What Wizards Did

Among all searches, wizards queried 43% of the time by title, 31% by author and 26% by catalog number. When uncertain about the search results, wizards sometimes attempted more than one query, on different ASR substrings or with different search types. They averaged about one query per adjacency pair, but often searched on multiple ASR substrings, more for title searches than for searches by author or catalog number. When the correct book appeared among the search results, it was typically high on the list returned by the query: first 85% of the time, second 8% of the time, third 3%, and later 4%.

When uncertain about the ASR or query results on either patron identity or book requests, wizards asked questions. The nature of these 11,562 questions is summarized in Table 3. Only 1% of all questions came before the wizard had made any database query at all. Wizards could ask for explicit confirmation of a full concept (e.g., ask the caller to confirm the title with a yes/no answer, as in line 14 of Figure 4) or of part of a concept (e.g., ask the caller to confirm a single word with a yes/no answer, as in line 8 of Figure 4), or confirm implicitly (e.g., have the TTS speak the title and then ask for the next book).

6.3 Exceptional Wizardry

Among our six wizards, WA and WB were considerably more successful than the others, as indicated by Table 3. WA and WB identified the most correct books per call, and had the fewest failed calls among all the wizards. Despite these similarities, WA and WB displayed very different approaches to their task.

WA, our *service-driven wizard*, worked hard to understand the caller. Per book request, WA did more searches than any other wizard, and asked more questions than any other wizard. One of the actions available on the wizard GUI under non-understanding was *move on*, which told the caller that the system was having difficulty understanding the current book request, and that the caller should continue on to a different book request and return to this one later. Among all wizards, WA used move on least often. WA often asked for confirmation, with the second-highest confirmation rate per book request. WA wanted to be sure; 92% of WA's confirmations were explicit ones.

In contrast, we theorize that WB, our *data-driven wizard*, had a pragmatic strategy: obtain high-quality ASR output to use for queries, search from it once or twice, and then either identify a book quickly or move on. Our data support this theory. WB indicated non-understanding more often, presumably to get better ASR output. Unlike the poorly-performing WD, however, WB never once asked a question about ASR output. Instead, WB used it to search. (Four of the 6 wizards, including WB, searched 1.70 – 1.73 times per request.) Moreover, WB asked questions far less often than any other wizard. (Other than WB's 2.28, question frequency was 3.53 – 4.09 per book

request.) More confident than the other wizards, WB confirmed the least often of any, and 14% of WB's confirmations were implicit. Finally, WB used the move-on strategy more than any other wizard, nearly twice as often as the next most frequent user.

6.4 Caller Impact

The caller population was deliberately varied to provide the wizards with a range of recognition difficulties. Table 4 offers some comparisons. The best caller, C1, had the most correctly identified books per call on average. In contrast, the two worst callers, C0 and C2, averaged about one correct book per call, and the wizards made more queries to try to help them. Nonetheless, C0 and C2 had hardly any fully successful calls, and more than 40% of their calls failed to order any books at all.

Our best caller, C1, was more readily understood. Speech from C1 had the best recognition across all request types. Whether the wizards searched for a title, an author, or a catalog number, C1 had the highest percentage of database returns that included the correct book. Indeed the book C1 requested was often returned by the first query. C1 required the fewest database queries per adjacency pair on average. C1's well-recognized speech also produced the shortest calls, both in number of utterances and in elapsed time.

Table 4. Comparative caller performance. C1 was the most successful caller; C0 and C2 were the least successful

	All callers	C1	C0	C2
Correct books per call	2.26	3.26	0.96	1.03
Fully successful calls	28%	63%	3%	5%
Failed calls (no books ordered)	17%	6%	41%	43%
<i>Wizard searched based on caller utterances by</i>				
Title	43%	32%	44%	47%
Author	33%	27%	37%	35%
Catalog number	26%	41%	19%	18%
<i>Wizard found the caller's book on searches by</i>				
Title	28%	42%	12%	11%
Author	33%	55%	20%	18%
Catalog number	58%	77%	35%	44%
Queries per book request	1.77	1.45	2.05	2.01
Utterances per call	22.36	19.29	23.97	21.99
Words per utterance	2.99	2.82	2.98	3.11

In contrast, speech from C0 had the worst recognition among all callers, and had the most utterances per call. Caller performance was not correlated with utterance

length, however. Among the 10 callers, C1 had the third fewest words per utterance, C2 had the third highest, and C0 the fifth. Speech from C2 had the worst recognition for titles and authors.

C1 is male; C0 and C2 are female; all three are native speakers of English. (C0 and C1 have an Eastern seaboard regional accent; C2 has a very slight Indian English accent.) Demographic data collected prior to the experiment indicated that C1 is age 18–25, while C0 and C2 are age 25–35. All three have a relatively fluent speech quality, although C0’s speech rate is slow. Among the other callers, one was a native speaker of Korean, another of Spanish, and two more described themselves as bilingual.

Caller success was based on more than ASR quality, however. Recall that the caller was permitted to select how she wanted to request the fourth book (by title, author, or catalog number). C1, our best caller, not only had the highest percentage of correctly identified books across request type, but also preferred the most readily recognized query type — speech from C1 evoked the highest percentage of queries by catalog number, and the lowest percentage for title and author. In contrast, speech from the poorly performing C0 evoked the most queries by author of any caller. Recognition distribution was not uniform across callers, For example, C3’s title and author searches were equally successful (30%), while C4’s title searches returned more correct titles (38%) than did her author searches (30%).

Differences among callers also emerged in the surveys. C3 reported that the system had difficulty recognizing catalog numbers, and was better with titles and authors. C9 reported that the system recognized author names poorly and often mispronounced them.

7 Discussion

The next generation of autonomous agents should be able to conduct a dialogue with their callers to achieve a common goal. Those callers are unlikely to be in soundproof facilities, and so the agent will have to contend with noisy acoustic data. Even if careful engineering eventually achieves perfect ASR, that will not prevent the agent’s confusion as its human partner mumbles or coughs her way through their conversation. The key, we believe, lies in a mixed initiative system with voice search, which permits the agent to apply knowledge and clarification dialogues to understand its caller.

7.1 Generality and Applicability

Because the fundamental features of spoken dialogue systems and the tolerance of their callers are fairly domain-independent, much of the work reported here is applicable to other domains where a rich knowledge base is readily available to provide the wizard with context. Even many of the clarifications and basic actions would readily transfer with a bit of rewording. A particularly helpful feature of this domain was an ability to describe the desired object (a book) in several ways: by title,

author, or catalog number. There was no assumption that the description was unique, but ability to shift to another descriptive mode was definitely helpful.

A different domain, however, would require a thoughtfully-designed wizard GUI, and the scale of the endeavor would have to justify the design and collection efforts. As for interface design, the preliminary work in Section 4 helped identify which actions should be available on the GUI for the full dialogue experiment. Considerably less effort went into the GUI’s layout; our focus was on problem solving, not speed or ergonomics. In our view, what is required for good wizard GUI design is expertise — expertise not in the domain itself, but in wizardry within it. Several wizards commented in the survey that they would have liked two additional basic actions on their GUI: “thank you” and “sorry.” We expect to include both in future experiments.

Asynchronous SDS architectures (e.g., [6, 27, 28]) are not hampered by the traditional pipeline. An asynchronous architecture can bring to bear features like those collected here to disambiguate human communication. An asynchronous SDS can also profit from the work reported here, which is fundamentally about modes of problem solving. We have under construction an asynchronous SDS architecture, *FORRSooth*, that interleaves spoken language understanding and dialogue management [29, 30]. Models derived from the full dialogue experiment are expected to identify additional features and provide guidance on their use in the new system. Indeed, simulation of some wizard behaviors may require asynchronicity. Often, while a wizard decides what action to take next, she may also update her beliefs about the book request, decide whether an ASR output line is worthy of attention at all, decide whether a request is for a title or an author or a catalog number, and estimate the utility of her choices. In a pipeline architecture, these interrelations for the most part must be ignored.

7.2 How Wizards Solved Problems

Regardless of the SDS architecture, every agent designer need not mount a full-scale experiment like the one reported here. We have detected two different but equally successful approaches for an autonomous agent that understands during dialogue. WA’s service-oriented approach persists in its attempt to understand, and asks many questions. It uses voice search extensively, and regularly seeks confirmation (particularly explicit confirmation) from the caller. WB’s data-driven approach is more focused on the goal, and also less chatty and determined. It has more confidence in its own decisions, and seeks less guidance and reinforcement from the caller. We suspect that its matching mechanism is more elaborate and that it would more readily accelerate problem solving, and look forward to studying it further.

The reader is well-advised to compare WA and WB in terms of throughput. WB is faster: 48% of WB’s tasks ran over the 6-minute limit, while 61% of WA’s did. WA spent the extra time asking questions and confirming results, while WB preferred to abandon problematic book requests quickly. (For this reason, the dialogues in Figures 4 and 5 were chosen from calls to WA. Transcripts of calls to WB are far less evocative.) WB processed 20% more book requests than WA did, and yet achieved

the same accuracy. While WA’s questions focused more on the book request, WB wanted search-worthy ASR output.

The two least successful wizards, WD and WE, had the fewest fully successful calls, and the fewest correct titles per call. (WE also had the most failed calls.) WD and WE did try hard; among all the wizards, they had by far the most calls that exceeded the 6-minute limit (70% and 79%, respectively). WD and WE, however, tried to understand the ASR with less reliance on voice search: 12% and 15% of their questions, respectively, concerned the ASR, compared to 0% – 6% for the other wizards. They also recorded the most questions per request before any database search at all. Wizards who focused on the ASR’s version of the callers’ words, rather than on their intent, were less successful.

One can readily imagine situations in which one or the other of the equally successful WA and WB approaches would be preferable, as reflected by SDS caller studies.

Users talk less to WB (who had the fewest user utterances per call of all wizards) and WB talks less to them (and also had the fewest questions per call of all wizards). WB’s confidence, however, sometimes confused the callers. Two of them indicated by survey that when “the system” had asked for the next book without telling them that it had just identified one, they asked for a summary at the end of the order and were surprised to find that “it” had gotten it right. An implemented version of WB should probably confirm more often.

There are presumably many reasons for the differences between our two successful wizards. WA majored in linguistics as an undergraduate and is now a Masters student in computer science, while WB is an undergraduate in computer science. WA is female and WB is male. Although their behaviors are consistent with gender-based styles of verbal communication once noted in the sociolinguistic literature, that is no longer a widely-accepted result [31, 32]. Rather, it appears to us that WA and WB brought different skill sets and attitudes to the task, and used the GUI to exploit them.

We do not claim that there are only two ways to succeed at this experiment as a wizard, merely that we observed only two in our data. We sought to identify, and now intend to implement, simple, rapid mechanisms that would improve CheckItOut’s accuracy in ways familiar and acceptable to its callers. We are training models of both WA and WB at this writing. An open question is which of them would be most appropriate. That choice may be application-dependent or even caller-dependent. We anticipate further experiments with both models.

7.3 System Performance

Some comparisons between the baseline and the full dialogue experiment are instructive. Our wizards had extensive information on the GUI to process, and, as expected, they did not respond in “real time” — callers, as forewarned, waited unnaturally long for a response. Wizards were far slower (required about twice as

much time per call) than CheckItOut. Given the wizards' instructions and ASR output where about half the words were incorrect, however, the wizards' persistence and the callers' tolerance despite lengthy pauses were exceptional. Indeed, one caller who had also participated in the baseline experiment volunteered that he was proud to participate in an experiment where the software had shown such marked improvement!

Wizards had been asked to facilitate orders as best they could and, compared to the baseline, they understood more often. Wizards identified more books correctly than CheckItOut: 2.26 books per call compared to 2.15 for the baseline. Our two best wizards did even better, with 2.69 and 2.54 books per call. Wizards misunderstood less often than CheckItOut: callers signaled misunderstanding ("That's not what I said") five times more often with the baseline than with the wizards (0.54 per call compared to 0.11 with the baseline). Moreover, wizards ordered the wrong books about four times less often: 0.18 per call compared to 0.79 with the baseline.

The next step is to build new dialogue managers for CheckItOut that model our most successful wizards. To do so, we will further mine the corpus developed here. Other work in this area has considered a small set of features (e.g., 10 in [33] and 17 in [16]). We successfully learned models from the preliminary book title experiment [22]. In that work we began with 60 features, and learned, for example, decision trees that successfully predicted when the best wizard would select a title ($F = .91$) or ask a question ($F = .68$). Learning from full dialogues is considerably more complex than learning from book titles alone, however. For the experiment recounted here, we extracted 163 features. We expect that different feature combinations best predict different wizard actions, and that feature selection informed by knowledge about SDS components will support learning the best models. To this end, we have developed a new, general method for feature selection prior to learning a wizard model and an SDS-specific modification to it [34]. Once learned, models of WA and WB and their particularly relevant features will provide decision rationales in a repertoire of competing strategies for FORRSooth (described in Section 7.1).

8 Conclusion

A Wizard of Oz study provides data that allows computer scientists to model an agent's conversational skill on people. The corpus developed here will be released to the research community in 2012. It is distinguished from other wizard studies (described in Section 3.3) by its size, its richness, and its real (rather than simulated) ASR. It is also noteworthy for its 163 features that describe the experience of the system (ASR, NLU, confidence annotator, backend), the experience of the wizard, and the dialogue history.

To support the next-generation of autonomous agents in human-machine dialogue, we have mined this corpus for insight into the ways that people manage to understand one another during dialogue. Contextual reference (here, as voice search) is clearly more useful than extensive attention to the words detected by the ASR. Repeated non-understandings make CheckItOut terminate a call, and repeated misunderstandings force the caller to repeat "That's not what I said." In contrast, our wizards asked questions, and thereby understood what the caller did and did not want more often. Of

the two successful approaches for an agent identified here, one is primarily service oriented, and the other is more data driven. Together they serve as guidelines for how an agent should, and should not, converse with people.

Acknowledgments. This research was supported in part by the National Science Foundation under awards IIS-084966, IIS-0745369, and IIS-0744904.

References

1. Levin, E., Passonneau, R.: A WOz Variant with Contrastive Conditions. In: Interspeech Satelite Workshop, Dialogue on Dialogues: Multidisciplinary Evaluation of Speech-Based Interactive Systems (2006)
2. Passonneau, R.J., Epstein, S.L., Ligorio, T., Gordon, J., Bhutada, P.: Learning About Voice Search for Spoken Dialogue Systems. In: 11th Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL HLT 2010), pp. 840–848 (2010)
3. Bohus, D., Rudnicky, A.: The Ravenclaw Dialogue Management Framework: Architecture and Systems. *Computers in Speech and Language* 23, 332–361 (2009)
4. Raux, A., Langner, B., Black, A., Eskenazi, M.: Let's Go Public! Taking a Spoken Dialog System to the Real World. In: Interspeech 2005, Eurospeech (2005)
5. Seneff, S., Hurley, E., Lau, R., Pao, C., Schmid, P., Zue, V.: Galaxy II: A Reference Architecture for Conversational System Development. In: 5th International Conference on Spoken Language Systems, ICSLP 1998 (1998)
6. Raux, A., Eskenazi, M.: A Multi-Layer Architecture for Semi-Synchronous Event-Driven Dialogue Management. In: IEEE Workshop on Automatic Speech Recognition and Understanding, ASRU 2007 (2007)
7. Raux, A., Eskenazi, M.: Optimizing Endpointing Thresholds Using Dialogue Features in a Spoken Dialogue System. In: SIGdial 2008 (2008)
8. Huggins-Daines, D., Kumar, M., Chan, A., Black, A.W., Ravishankar, M., Rudnicky, A.: Pocketsphinx: A Free, Real-Time Continuous Speech Recognition System for Hand-Held Devices. In: International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 185–189 (2008)
9. Ward, W., Issar, S.: Recent Improvements in the CMU Spoken Language Understanding System. In: ARPA Human Language Technology Workshop, pp. 213–216 (1994)
10. Bohus, D., Rudnicky, A.: Integrating Multiple Knowledge Sources for Utterance-Level Confidence Annotation in the Cmu Communicator Spoken Dialogue System. Technical report, Carnegie Mellon University (2002)
11. SWIFT: Small Footprint Text-to-Speech Synthesizer, <http://www.cepstral.com/>
12. Bohus, D.: Error Awareness and Recovery in Task-Oriented Spoken Dialogue Systems. Ph.D. thesis proposal, Carnegie Mellon University (2004)
13. Stoyanchev, S., Stent, A.: Predicting Concept Types in User Corrections in Dialogue. In: EACL Workshop SRSL, pp. 42–49 (2009)
14. Litman, D., Hirschberg, J., Swerts, M.: Characterizing and Predicting Corrections in Spoken Dialogue Systems. *Computational Linguistics* 32, 417–438 (2006)
15. Dix, A., Finlay, J., Abowd, G.D., Beale, R.: Human-Computer Interaction. Prentice Hall (2003)
16. Rieser, V., Lemon, O.: Using Machine Learning to Explore Human Multimodal Clarification Strategies. In: COLING/ACL 2006, pp. 659–666 (2006)

17. Skantze, G.: Exploring Human Error Recovery Strategies: Implications for Spoken Dialogue Systems Speech Communication. Special Issue on Speech Annotation and Corpus Tools 45, 207–359 (2005)
18. Rieser, V., Kruijff-Korbayová, I., Lemon, O.: A Corpus Collection and Annotation Framework for Learning Multimodal Clarification Strategies. In: Sixth SIGdial Workshop on Discourse and Dialogue, pp. 97–106 (2005)
19. Sherwani, J., Yu, D., Paek, T., Czerwinski, M., Acero, A.: Voicepedia: Towards Speech-Based Access to Unstructured Information. In: Interspeech 2007 (2007)
20. Gordon, J.B., Passonneau, R.J.: An Evaluation Framework for Natural Language Understanding in Spoken Dialogue Systems. In: Seventh International Conference on International Language Resources and Evaluation (LREC 2010). European Language Resources Association, ELRA (2010)
21. Passonneau, R., Epstein, S.L., Gordon, J.B.: Help Me Understand You: Addressing the Speech Recognition Bottleneck. In: AAAI Spring Symposium on Agents that Learn from Human Teachers. AAAI (2009)
22. Ligorio, T., Epstein, S.L., Passonneau, R.J., Gordon, J.B.: What You Did and Didn't Mean: Noise, Context, and Human Skill. In: Cognitive Science - 2010 (2010)
23. Passonneau, R.J., Epstein, S.L., Gordon, J.B., Ligorio, T.: Seeing What You Said: How Wizards Use Voice Search Results. In: IJCAI 2009 Workshop on Knowledge and Reasoning in Practical Dialogue Systems. AAAI Press (2009)
24. Ratcliff, J.W., Metzener, D.: Pattern Matching: The Gestalt Approach. Dr. Dobb's Journal (1988)
25. Bangalore, S., Boulllier, P., Nasr, A., Rambow, O., Sagot, B.: Mica: A Probabilistic Dependency Parser Based on Tree Insertion Grammars. In: NAACL HLT 2009 Companion Volume: Short Papers, pp. 185–188 (2009)
26. Sacks, H., Schegloff, E.A., Jefferson, G.: A Simplest Systematics for the Organization of Turn-Taking for Conversation. Language 50, 696–735 (1974)
27. Allen, J., Ferguson, G., Stent, A.: An Architecture for More Realistic Conversational Systems. In: 6th International Conference on Intelligent User Interfaces, pp. 1–8 (2001)
28. Skantze, G., Gustafson, J.: Attention and Interaction Control in a Human-Human-Computer Dialogue Setting. In: Tenth Annual Meeting of the Special Interest Group in Dialogue and Discourse (SIGdial 10), pp. 310–313 (2009)
29. Gordon, J.B., Passonneau, R.J., Epstein, S.L.: Helping Agents Help Their Users Despite Imperfect Speech Recognition. In: AAAI Symposium Help Me Help You: Bridging the Gaps in Human-Agent Collaboration (2011)
30. Gordon, J., Epstein, S.L., Passonneau, R.J.: Learning to Balance Grounding Rationales for Dialogue Systems. In: 12th SIGDIAL on Dialogue and Discourse (2011)
31. Cameron, D.: The Myth of Mars and Venus: Do Men and Women Really Speak Different Languages?, Oxford (2007)
32. Cameron, D.: Sex/Gender, Language and the New Biolism. Applied Linguistics 31, 173–192 (2010)
33. Skantze, G., Edlund, J.: Early Error Detection on Word Level. In: ISCA Tutorial and Research Workshop on Robustness Issues in Conversational Interaction (2004)
34. Ligorio, T.: Feature Selection for Error Detection and Recovery in Spoken Dialogue Systems. Ph.D. thesis, Computer Science, The Graduate Center of The City University of New York, New York (2011)
35. Cao, L., Gorodetsky, V., Mitkas, P.: Agent Mining: The Synergy of Agents and Data Mining. IEEE Intelligent Systems 24(3), 64–72 (2009)

An Instance-Window Based Classification Algorithm for Handling Gradual Concept Drifts

Vahida Attar¹, Prashant Chaudhary¹, Sonali Rahagude¹,
Gaurish Chaudhari¹, and Pradeep Sinha²

¹ College of Engineering, Pune (CoEP), Shivajinagar, Pune 411 005, India
`{vahida.comp,chaudharypm07.comp,rahagudesp07.comp,
chaudharigs07.comp}@coep.ac.in`

² Centre for Development of Advanced Computing (C-DAC), Pune 411007, India
`psinha@cdac.in`

Abstract. Mining concept drifting data stream is a challenging area for data mining research. In real world, data streams are not stable but change with time. Such changes termed as drifts in concept of the data stream are categorized into gradual and abrupt, based on the amount of drifting time, i.e. the time steps taken to replace the old concept completely by the new one. In traditional online learning systems, this categorization has not been exploited in developing different approaches for handling different types of drifts in the data stream. Such handling of concept drifts according to their type can help improve the performance of the classification system and hence, the issue can be explored further. Among the most popular and effective approaches to handle concept drifts is ensemble learning, where a set of models built over different time periods is maintained and the predictions of models are combined, usually according to their expertise level regarding the current concept. If early instances of new concept are stored and used for ensemble learning once the drift is detected, this may help increase the overall accuracy after the drift. Moreover, if an ensemble learns with zero diversity for instances of a new concept during the drifting period, the ensemble may learn the new concept faster, thus boosting recovery. The paper presents the above mentioned approach for effective handling of gradual concept drifts in the data streams.

Keywords: Gradual concept drift, Online Learning, Ensemble, Boosting, Instance Window, Diversity.

1 Introduction

Online learning has a wide variety of applications in which training data is available continuously in time and there are time and space constraints. For example, web traffic monitoring, network security, sensor signal processing, credit card fraud detection etc. Online learning algorithms process each training example once on arrival, without the need for storage or reprocessing and maintain a current hypothesis that reflects all the training instances so far [11]. In this

way the learning algorithm takes as input a single training instance as well as a hypothesis input and outputs an updated hypothesis [5].

Online learning environments are often non-stationary and the variables to be predicted by the learning machine may change with time, this change is referred to as concept drift. Concept drifts can be categorized based on their speeds. Speed is the inverse of drifting time (*drifting_time*), which can be measured as the number of time steps taken for a new concept to completely replace the old one [9]. In this way, a higher speed is related to a lower number of time steps and a lower speed is related to a higher number of time steps. According to the speed, drifts can be categorized as either abrupt, when the complete change occurs in only one time step, or gradual, otherwise [9].

For example, sudden change in buying preferences of a share purchaser due to a dip in stock price of some particular company (abrupt concept drift), whereas adapting from an old mailing system to a new one where people use both systems for some time initially and then switch to the new mailing system (gradual concept drift).

Ensemble learning is among the most popular and effective approaches to handle concept drift, in which a set of concept descriptions built over different time intervals is maintained, predictions of which are combined using a form of voting, or the most relevant description is selected [16]. Ensembles of classifiers have been successfully used to improve the accuracy of single classifiers in online learning [11,5,14,8].

In this paper we propose a novel ensemble approach of handling gradual concept drift in which all the classifiers in the ensemble are trained for examples of the new concept, during the drift period using techniques of instance window and zero diversity.

The paper is further organized as follows : Section 2 explains the related work that includes concept drift categorization, the existing drift detection techniques, drift handling techniques and the existing EDDM based approach, Section 3 explains the proposed approach and its theoretical basis, Section 4 presents the experiments and the analysis for determining the parameters of algorithm, Section 5 presents the experimental results and Section 6 concludes the paper and suggests the future work related to agents.

2 Related Work

2.1 Concept Drift Categorization

Concept refers to the target variable, which the model is trying to predict. Concept change is the change of the underlying concept over time. Section 1 explains concept drift as non-stationary on-line environments. The term concept drift can be formally defined as follows : Concept drift is the change in the distribution of a problem, which is characterized by the joint distribution $p(x, w)$, where x represents the input attributes and w represents target classes [6,10].

Two kinds of concept drift may occur in the real world normally – abrupt and gradual [19]. Let S_o and S_n be two sources which generate instances

corresponding to old and new concepts, respectively. Let time t be the time at which the drift occurs, before which all the instances are from source S_o .

Abrupt Drift. The simplest pattern of a change is abrupt drift, when at time t a source S_o is suddenly replaced by source S_n and is continued further.

Gradual Drift. Another kind of change is gradual drift which refers to a certain period after t when both sources S_o and S_n are active. As time passes, the probability of sampling from source S_o decreases, probability of sampling from source S_n increases. Note, that at the beginning of this gradual drift, before more instances are seen, an instance from the source S_n might be easily mixed up with random noise.

The period when both sources S_o and S_n are active is called as drifting period (*drifting_time*). The value of *drifting_time* is in inverse relation with the speed of the drift. The value of *drifting_time* is 1 for abrupt drift and a high value of *drifting_time* refers to gradual drift. Hence, greater the *drifting_time*, less is the speed, i.e. change is more gradual. We adopt this definition of gradual drift as the basis for developing the proposed algorithm to handle gradual changes in data streams.

2.2 Drift Detection Techniques

Drift detection can be achieved by detecting changes in the distribution of the training instances monitors the online error-rate of the algorithm [19]. The method controls the trace of the online error of the algorithm. For the actual context they define a warning level, and a drift level.

Drift Detection Method (DDM) [6], uses number of classification errors to detect change. As it uses number of errors, this approach has good behavior detecting abrupt drifts and fast gradual drifts (small *drifting_time*). But, it has difficulties detecting slow gradual drifts.

Early Drift Detection Method (EDDM) [1] uses the distance between classification errors (number of instances between two classification errors) to detect change, instead of using number of classification errors. Hence, this method detects both abrupt and gradual drifts (slow and fast) efficiently. This technique is explained in Section 2.3 in detail.

2.3 Drift Handling Techniques

Pure Ensemble Learning. An ensemble consists of a set of individually trained classifiers, with disagreement among them, whose predictions are combined when classifying novel instances. The implicit aim of having disagreement or diversity is to have at least one classifier in the ensemble trained for each distinct concept. Thus, when tackling non-stationary concepts, ensembles of classifiers can adapt to change quickly by pruning under-performing parts of the ensemble, and they therefore usually also generate more accurate concept descriptions.

Learning with Drift Detection. The problem with pure ensemble learning is that recovery time is very high due to no drift detection, especially, in case of gradual drifts. Here, using an ensemble of classifiers is preferred as they improve accuracy of single classifiers. This approach is explained as follows with EDDM as drift detection technique.

EDDM Based Approach. EDDM calculates the average distance between two errors obtained by the classifier system (p_i) and its standard deviation (s_i) and stores their maximum values so far (p_{max} and s_{max}) [1].

A learning system which uses EDDM behaves in the following way. During the warning period ($(p_i+2s_i) / (p_{max}+2s_{max}) < \alpha$), a new classifier system is created and trained for all the incoming instances till the drift level is detected by EDDM. If a warning level is cancelled, the new classifier system is discarded. If a drift level ($(p_i+2s_i) / (p_{max}+2s_{max}) < \beta$) is confirmed, p'_{max} and s'_{max} are reset and the original classifier system is replaced by the new one. New p'_{max} and s'_{max} are considered only after 30 errors have occurred [9]. Thus, EDDM approach adopts the strategy of learning a new classifier from scratch when a drift is detected. This approach has been implemented in MOA as the classifier named *SingleClassifierDrift*.

3 Proposed Approach

In the initial experimentation on drifting data, it was observed that accuracy of an online learning system for gradual drifts is lesser than that for abrupt drifts. Moreover, recovery in case of gradual drifts was less and slower. The reason for less accuracy in case of gradual drifts is because of the composition of stream in case of a gradual drift. As mentioned in Section 2.1, data stream during the drifting period consist of instances from the old as well as the new concept. Due to this, the classifier system is trained on both the concepts whenever a concept drift occurs. As opposed to this, in abrupt drift, data stream after the drift consists only of instances from the new concept. Since the classifier learns on the new concept only in this case, it learns better, hence, leads to greater classification accuracy and recovery. Thus, the main aim was to make the classifier learn on the new concept only when a gradual drift occurs. The classifier should be trained with enough instances from the new concept in order to attain greater classification accuracy. In case of an ensemble classifier system, the classification of a given data instance depends on voting of all the individual classifiers. So, a way of making the classifier learn better on the new concept, is to make all the individual base classifiers learn on concepts from the new stream. The ensemble will be trained well on the new concept as all the classifiers learn the concept. We call this method of making all base classifiers learn as zero diversity.

Another way of improving the accuracy of classifier for gradual drifts, is to store instances of new concept in advance so that enough instances of the new concept are available to the new classifier when the drift is confirmed by the drift detection method. Thus, an instance-window containing instances from the

new concept only can improve learning of the classifier system thus improving its recovery.

The proposed approach called as GPS combines both the above described methods in an attempt to improve classifier accuracy and recovery in case of gradual drifts.

We use EDDM as the drift detection method because it is a recent method which has shown to attain similar accuracy to a previous methods when the drifts are abrupt and a better accuracy when the drifts are gradual. For efficient handling of gradual concept drift, it is desirable that the new concept is learnt well by the ensemble. For achieving this, the proposed approach uses two techniques:

Algorithm 1. Training Algorithm for Ensembles used in Proposed Approach (Use of Zero Diversity)

Input: *Input inst, flag , λ_m^{sc} , λ_m^{sw} , BaseLearner = Hoeffding Tree*

```

1: Set weight of instance  $\lambda_d \leftarrow 1$ 
2:  $l \leftarrow ensembleLength$ 
3: for  $m = 1,2,...,l$  do
4:   if  $flag == normalDiverse$  then
5:     Set  $k \leftarrow Poisson(\lambda_d)$ 
6:   else if  $flag == zeroDiverse$  then
7:     Set  $k \leftarrow 1$ 
8:   end if
9:   if  $k > 0.0$  then
10:    Update  $h_m$  with the current instance inst
11:   end if
12:   if  $h_m$  correctly classifies instance inst then
13:      $\lambda_m^{sc} \leftarrow \lambda_m^{sc} + \lambda_d$ 
14:      $\lambda_d \leftarrow \lambda_d(N/2\lambda_m^{sc})$ 
15:   else
16:      $\lambda_m^{sw} \leftarrow \lambda_m^{sw} + \lambda_d$ 
17:      $\lambda_d \leftarrow \lambda_d(N/2\lambda_m^{sw})$ 
18:   end if
19: end for
```

Output: *Updated ensemble*

3.1 Use of Zero Diversity

When a warning level is triggered by EDDM, we start training a new ensemble with zero diversity i.e. all the individual base classifiers in the ensemble learn on the given instance. This new ensemble is a copy of the current ensemble. Also, only those instances which are wrongly classified by the current ensemble are given for training to the new ensemble (Algorithm 1). As mentioned previously in this section, the ensemble should learn on new concept only for better

classification. Here, we make an assumption whenever a warning is flagged for the possibility of a drift, instances misclassified by the classifier should belong to the new concept as the classifier has still not learnt the new concept well. Hence, the training is done only on misclassified instances.

In an online learning system, *Poisson distribution* is used to decide if a classifier in the ensemble will be presented an incoming instance for training [12]. So, not all the classifiers in the ensemble are trained for the same given instance. In this approach, we tune the *Poisson* parameter in order to obtain different diversities for the ensemble. As shown in Algorithm 1, the training method is passed a flag to determine the diversity for training the ensemble on the current instance. When the *flag* is *normalDiverse*, the value of k is determined by *Poisson* parameter λ . Thus, when the *flag* is *zeroDiverse*, the value of k is set to 1, due to which all the classifiers in the ensemble get trained on the current instance. Thus, for new concept instances, training all the classifiers improves the accuracy of the ensemble.

The reason for training the new ensemble with zero diversity is to help the new ensemble learn the new concept more efficiently and quickly. Hence, it leads to better accuracy and faster recovery. Since, it learns only on misclassified instances (supposed to be of the new concept) the new ensemble learns better on the new concept as all of its individual classifiers get trained on the given instance.

3.2 Use of Instance Window

An instance window of fixed size is maintained for storing the recent instances to be trained on the new ensemble. These instances are stored in first-in-first-out fashion. The instance window size was experimentally concluded to be 50.

For the proposed approach, the instances on which the new ensemble is trained with zero diversity after the warning level, are the instances misclassified by the current ensemble. This is because, in a gradual drift stream, whenever a warning level is flagged for the possibility of a drift, instances misclassified by the current ensemble should belong to the new concept as the current ensemble has not still learnt the new concept well. Hence, training is done only on the misclassified instances. For the same reason, we store only those instances in the window, that have been misclassified by the current ensemble, so that during the drift period, the misclassified instances will be those of the new concept and hence the instance window will start containing instances from the new concept.

The new ensemble is trained on this window by adding a new EDDM level (Post Warning level) in EDDM and then learning instances from the instance window. (Algorithm 2).

Post Warning Level : A New EDDM Level. The proposed approach uses EDDM for drift detection. It has been observed that EDDM detects false drifts and raises false warnings as well. Hence, while learning from the instance window,

Algorithm 2. Proposed Approach (Use of Instance Window)

Input: *inst*, *CurrentEnsemble*, *NewEnsemble*

- 1: *predictedClass* \leftarrow *PredictClassOfInstance*(*inst*)
- 2: *classification* \leftarrow (*predictedClass* == *inst.Class*)
- 3: **if** *classification* == *false* **then**
- 4: add *inst* to *window*[]
- 5: **end if**
- 6: *level* \leftarrow *computeEDDMLevel*(*inst*, *classification*)
- 7: **if** *level* == *STABLE* **then**
- 8: *NewEnsemble.train*(*inst*, *normalDiverse*)
- 9: **else if** *level* == *WARNING* **then**
- 10: **if** *classification* == *false* **then**
- 11: *NewEnsemble.train*(*inst*, *zeroDiverse*)
- 12: **end if**
- 13: **else if** *level* == *POSTWARNING* **then**
- 14: **for** *instTemp* IN *window*[] **do**
- 15: *NewEnsemble.train*(*instTemp*, *zeroDiverse*)
- 16: **end for**
- 17: **if** *classification* == *false* **then**
- 18: *NewEnsemble.train*(*inst*, *zeroDiverse*)
- 19: **end if**
- 20: **else if** *level* == *DRIFT* **then**
- 21: *resetCurrentEnsemble*()
- 22: *CurrentEnsemble* \leftarrow *NewEnsemble*
- 23: *resetNewEnsemble*()
- 24: **end if**
- 25: *CurrentEnsemble.train*(*inst*, *normalDiverse*)

Output: Updated *NewEnsemble* and *CurrentEnsemble*

it is important to ensure that the window contains instances relevant to the new concept only. If the classifier is made to learn from the window on the warning level itself, it may be possible that the warning raised is false or due to noise. In that case, instances in the window might not be relevant to the new concept.

Also, it is necessary that the classifier learns the new concept well before the drift level is actually detected by EDDM. Only in this way, greater recovery can be ensured. Hence, we introduce a new level in EDDM viz. Post Warning level. The Post Warning level has been introduced so that more relevant instances of new concept get stored in the error instance window. The optimal value for this level has been determined experimentally as 0.925.

Learning Instances from Window. Whenever Post Warning level is triggered, the new ensemble is learnt on all the instances in the error instance window. The advantage of this window is that, it contains latest misclassified instances (supposed to belong to the new concept). So, training the new ensemble on this window instances will lead to faster recovery from the gradual drift.

3.3 Switching to New Ensemble

Once the drift level occurs, the new ensemble is set as the current ensemble used for classification. The now-current ensemble then starts learning with normal diversity. The new ensemble is then reset which is used for future drifts.

4 Experimentation

4.1 Artificial Datasets

When working with real-world datasets, it is not possible to know exactly when a drift starts to occur, which type of drift is present, or even if there really is a drift. So, it is not possible to perform a detailed analysis of the 0 of algorithms in the presence of concept drifts using only pure real-world datasets. In order to analyze the strong and weak points of a particular algorithm, it is necessary first to check its behavior using artificial datasets containing simulated drifts. Depending upon the type of drift in which the algorithm is weak, it may be necessary to adopt a different strategy to improve it, so that its performance is better when applied to real-world problems. To generate the datasets with concept drift, we use the approach used by Minku L., 2008 [10].

We created the datasets for the following problems (Table 1): Circle, Sine, Line and Plane2d.

In circle, line and plane2d, r , a_0 and a_0 represent the concepts respectively. For sine, both c and d represent concepts, thus generating two problems viz. SineH and SineV. The instances in the datasets contain x or x_i and y as the input attributes and the concept (which can assume a value 0 or 1) as the output attribute. See Table 1.

We choose these problems for generating datasets because of the wide variety of problems that get covered. Circle dataset represents second degree problems, Sine represents trigonometric problems, Line represents two dimensional linear problems and Plane represents 3 dimensional problems.

Gradual drift is introduced in the datasets by decreasing the speed of the drift. The speed of the drift can be modelled by degree of dominance function representing the probability that an instance of the old and new concept will be presented to the learning system. N is number of time steps before drift started to occur and $drifting_time$ is the number of time steps for a complete replacement of the old concept.

The $drifting_time$ is 1 for abrupt drifts whereas for gradual drifts, it has been set to $0.25N$ and $0.50N$. Since we are dealing with gradual drifts, we created datasets with gradual drifts only. Thus for each of the 5 problems, we have 2 datasets corresponding to 2 different drift speeds for gradual, giving 10 different datasets for a given size. We have created datasets of various sizes such as 2000, 50000, 100000. In the datasets with 2000 and 50000 instances, drifts have been introduced at 1000 and 25000 respectively. In the datasets of size

100000 with 1 drift, drift is at position 25000. Also, a dataset of size 100000 having 3 drifts at positions 20000, 40000, 75000 has been created. Thus, giving 40 datasets in all.

Table 1 describes the various problems and values used for generating datasets from these problems. Three ranges of concepts are given corresponding to each problem. Only the first range is used for generating datasets with one drift whereas all three ranges are used for generating datasets with three drifts.

Also, to increase the difficulty of the problems, we added 8 irrelevant attributes and 10% class noise to the plane2d datasets. The code for generating these datasets is written in C language.

Apart from these, we have also used some standard datasets such as SEA [15] (size 100000) with 3 drifts at 25000, 50000 and 75000, Waveform (size 100000) with 3 drifts at 25000, 50000 and 75000, Waveform [3] (size 150000) with no drift, Hyperplane [18] (size 50000) with 1 drift at 25000 which were generated in MOA.

The reason for using the datasets of 100000 instances with 1 drift at 25000 is to evaluate the performance of the GPS at a time long after the drift has been occurred. Drift-less datasets have been used to evaluate the performance of GPS in the absence of drifts in the data stream. Noisy datasets are used to check the noise sensitivity of the algorithm. Also, datasets with 3 drifts have been created so evaluate the performance in case of multiple drifts, moreover, the drifts have been positioned closely to check for good and quick recovery. Thus, we have covered wide variety of situations that can appear in the data streams.

Table 1. Artificial Datasets

Problem	Fixed Values	Range of Attributes	Range of Concepts
Circle $(x - a)^2 + (y - b)^2 \leq r^2$	$a=0.5, b=0.5$	$x:[0,1], y:[0,1]$	$r : 0.2 \rightarrow 0.5$ $r : 0.5 \rightarrow 0.1$ $r : 0.1 \rightarrow 0.5$
SineV $y \leq \text{asin}(bx + c) + d$	$a=1, b=1, c=0$	$x:[0,10], y:[-10,10]$	$d : -8 \rightarrow 7$ $d : 7 \rightarrow -7$ $d : -7 \rightarrow 6$
SineH $y \leq \text{asin}(bx + c) + d$	$a=5, d=5, b=1$	$x:[0,4\pi], y:[0,10]$	$c : 0 \rightarrow -\pi$ $c : -\pi \rightarrow 0$ $c : 0 \rightarrow -\pi$
Line $y \leq -a_0 + a_1x$	$a_1=0.1$	$x:[0,1], y:[0,1]$	$a_0 : -0.1 \rightarrow -0.8$ $a_0 : -0.8 \rightarrow -0.2$ $a_0 : -0.2 \rightarrow -0.8$
Plane $y \leq -a_0 + a_1x_1 + a_2x_2$	$a_1=0.1, a_2=0.1$	$x:[0,1], y:[0,5]$	$a_0 : -0.7 \rightarrow -4.4$ $a_0 : -4.4 \rightarrow -0.5$ $a_0 : -0.5 \rightarrow -4.3$

Datasets consisting of 50,000 instances contain 1 drift at position 25,000. Datasets consisting of 1,00,000 instances contain 3 drifts at positions 20,000, 40,000 and 75,000.

4.2 Real Datasets

Spam Corpus. For testing on real world data, we chose the spam corpus dataset. This is a real world textual dataset uses SpamAssassin data collection [7]. This spam dataset consists of 9324 instances with 40,000 attributes and represents the gradual concept drift. There are 2 classes legitimate and spam, with the ratio around 20%.

Forest Cover (UCI Repository). The dataset [17] contains Geo-spatial descriptions of different types of forests. It contains 7 classes and 54 attributes and around 581,000 instances. We normalize the dataset, and arrange the data so that in any chunk at most 3 and at least 2 classes co-occur, and new classes appear randomly.

4.3 Implementation

The proposed algorithm in this paper is implemented in Java programming language on Linux platform. We have used the MOA – Massive Online Analysis [2] tool for all the experimentation of the proposed approach.

To build a picture of accuracy and time, we use the Interleaved Test-Then-Train evaluation model available in MOA. In Interleaved Test-Then-Train model, each individual instance can be used to test the model before it is used for training, and from this the accuracy can be incrementally updated. It also ensures a smooth plot of accuracy over time, as each individual instance will become increasingly less significant to the overall average.

The experiments were performed on a 2.59 GHz Intel Core 2 Duo processor with 3 GB main memory, running Ubuntu 10.04. For comparison of the performance different ensemble techniques like OzaBoost, OCBoost [13], OzaBag, OzaBagADWIN [2] are used. The first 3 with wrapper class *SingleClassifierDrift* (the EDDM Approach as implemented in MOA) which includes drift detection methods.

Hoeffding tree is used as the base learner for all ensembles. The ensemble size kept for each ensemble was 10.

For artificial datasets, the accuracy presented in the next section is reset at the time step $N + 1$ when the drift starts to happen. This was done to allow the evaluation of the behavior of the approach when the drift starts to occur. For real dataset, the accuracy is never reset.

For plotting the graphs of accuracy versus number of instances in order to compare the recovery and accuracy of the system at various time steps of the proposed approaches with existing ones, we used GNUplot4.2.

4.4 Measures Analyzed

Size of Instance Window. We experimented with window sizes ranging from 10 to 100 in steps of 10 and compared them for each dataset. It was observed that a window size of 50 gives the better results in case of gradual drift for all the datasets. So, we have fixed the optimal value of window size as 50 (Table 2).

Table 2. Comparison of Classification Accuracies for different sizes of Instance Window, Dataset-Size : 50,000

DATASETS	10	20	30	40	50	60	70	80	90	100
<i>drifting_time = 0.25N</i>										
Circle	0.866	0.865	0.866	0.865	0.87	0.869	0.866	0.861	0.854	0.852
SineV	0.916	0.915	0.905	0.914	0.919	0.91	0.905	0.909	0.917	0.916
SineH	0.819	0.817	0.816	0.813	0.823	0.825	0.818	0.824	0.81	0.824
Line	0.93	0.935	0.927	0.932	0.937	0.924	0.932	0.932	0.918	0.932
Plane	0.838	0.825	0.825	0.839	0.832	0.829	0.831	0.834	0.835	0.834
<i>drifting_time = 0.50N</i>										
Circle	0.852	0.852	0.854	0.856	0.84	0.86	0.85	0.862	0.852	0.847
SineV	0.893	0.89	0.888	0.89	0.894	0.886	0.889	0.886	0.888	0.886
SineH	0.813	0.813	0.813	0.808	0.821	0.816	0.815	0.821	0.812	0.815
Line	0.91	0.911	0.91	0.908	0.915	0.913	0.909	0.915	0.892	0.905
Plane	0.827	0.817	0.803	0.826	0.82	0.816	0.82	0.824	0.824	0.824

Ratio value for Post Warning Level. The Post Warning level is a new level added to EDDM between the warning level and drift level so as to determine a time during the drift period when the instance window is properly dominated by the instances of the new concept. In EDDM, the ratio $(p_i + 2s_i)/(p_{max} + 2s_{max})$ is checked against predefined values α (0.95) and β (0.90) to detect the warning and drift level respectively. Hence we needed to define a new value between α and β for detecting Post Warning level. Hence, we experimented with different values between the Warning and Drift levels i.e. 0.95 and 0.90 respectively. The values used in experimentation were 0.91, 0.92, 0.925, 0.93 and 0.94 (Table 3). It was found that the value 0.925 gave better results for the training of the new ensemble.

5 Results

The proposed approach GPS with Post Warning level = 0.925 and Instance-Window Size = 50 was compared with different online learning algorithms viz. EDDM approach (base learner – OzaBoost), EDDM approach (base learner – OzaBag), EDDM Approach (base learner – OCBoost) and OzaBagADWIN.

We performed comparisons for all the datasets with single as well as multiple drifts. Also, drifting time(speed of the drift) was varied (0.25N and 0.50N). To test the sensitivity of the algorithm, we also experimented on driftless datasets and noisy datasets.

Table 3. Comparison of Classification Accuracies for different values of Post Warning Level, Dataset-Size : 50,000

DATASETS	0.91	0.92	0.925	0.93	0.94
<i>drifting_time = 0.25N</i>					
Circle	0.865	0.870	0.870	0.869	0.834
SineV	0.908	0.906	0.916	0.915	0.914
SineH	0.807	0.787	0.823	0.816	0.82
Line	0.933	0.932	0.930	0.931	0.932
Plane	0.829	0.826	0.832	0.83	0.826
<i>drifting_time = 0.50N</i>					
Circle	0.852	0.853	0.840	0.851	0.830
SineV	0.891	0.889	0.895	0.893	0.892
SineH	0.801	0.786	0.821	0.815	0.812
Line	0.912	0.91	0.911	0.913	0.902
Plane	0.822	0.803	0.824	0.822	0.816

5.1 Accuracy

The results for datasets of size 50,000 with 1 drift, size 1,00,000 with 1 drift and size 1,00,000 with 3 drifts are tabulated in Table 4. The proposed algorithms improves performance for each of these datasets. The graphs for Circle (1 drift) and SineH (3 drift) are shown in Fig. 1 and Fig. 2 respectively. The figures for other artificial datasets were omitted due to space limitations.

As seen from the graph, the recovery time has considerably decreased in case of GPS. It is also seen the accuracy before the drift is also higher. This is because EDDM detects many false drifts. In the GPS algorithm the current ensemble is replaced by the new ensemble which is trained since the beginning of data stream whereas in EDDM approach the new ensemble is trained only after the warning level is detected, which reduces accuracy in case of false drifts.

Table 5 shows the results for various datasets like waveform, SEA, hyperplane and real datasets spam corpus and forest cover (covtype). The results for spam corpus shows a considerable increase in the accuracy as well as the recovery of the algorithm in the drift period is good. See Fig. 3. The results for other real dataset, covtype are also good. The accuracy rise is relatively higher than other algorithms.

5.2 Noise Sensitivity

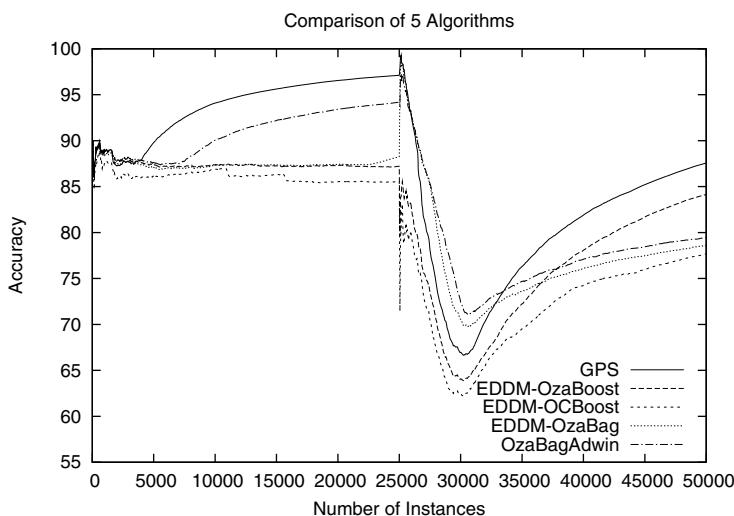
Class noise is defined as the percentage of the total instances whose actual class labels have been changed. Different amount of class noise was added in the

Table 4. Comparison of Classification Accuracies

Dataset	EDDM OzaBoost	EDDM OzaBag	EDDM OCBoost	OzaBag ADWIN	GPS
Size:50,000; No. of Drifts:1; <i>drifting_time</i> = 0.25 <i>N</i>					
Circle	0.786	0.841	0.777	0.796	0.876
Line	0.926	0.925	0.913	0.934	0.941
Plane	0.841	0.842	0.828	0.844	0.866
SineH	0.743	0.817	0.733	0.786	0.872
SineV	0.908	0.924	0.899	0.918	0.932
Size:50,000; No. of Drifts:1; <i>drifting_time</i> = 0.50 <i>N</i>					
Circle	0.761	0.778	0.747	0.778	0.790
Line	0.875	0.876	0.835	0.887	0.889
Plane	0.803	0.803	0.788	0.804	0.819
SineH	0.756	0.718	0.696	0.740	0.835
SineV	0.866	0.856	0.840	0.867	0.877
Size:1,00,000; No. of Drifts:1; <i>drifting_time</i> = 0.25 <i>N</i>					
Circle	0.854	0.806	0.827	0.835	0.895
Line	0.929	0.919	0.930	0.932	0.946
Plane	0.836	0.801	0.826	0.839	0.838
SineH	0.818	0.724	0.768	0.785	0.870
SineV	0.922	0.898	0.914	0.917	0.936
Size:1,00,000; No. of Drifts:1; <i>drifting_time</i> = 0.50 <i>N</i>					
Circle	0.825	0.792	0.818	0.829	0.855
Line	0.904	0.878	0.904	0.912	0.922
Plane	0.821	0.774	0.804	0.823	0.831
SineH	0.781	0.678	0.779	0.770	0.857
SineV	0.889	0.867	0.895	0.893	0.909
Size:1,00,000; No. of Drifts:3; <i>drifting_time</i> = 0.25 <i>N</i>					
Circle	0.790	0.814	0.777	0.763	0.845
Line	0.884	0.892	0.895	0.899	0.908
Plane	0.749	0.796	0.786	0.806	0.818
SineH	0.761	0.789	0.669	0.735	0.833
SineV	0.865	0.872	0.866	0.878	0.891
Size:1,00,000; No. of Drifts:3; <i>drifting_time</i> = 0.50 <i>N</i>					
Circle	0.790	0.814	0.774	0.763	0.846
Line	0.884	0.892	0.895	0.899	0.901
Plane	0.749	0.797	0.786	0.806	0.818
SineH	0.761	0.789	0.669	0.735	0.833
SineV	0.865	0.872	0.866	0.878	0.891

Table 5. Comparison of Classification Accuracies, Miscellaneous Datasets

Dataset	EDDM OzaBoost	EDDM OzaBag	EDDM OCBoost	OzaBag	GPS
Other Artificial Datasets					
Hyperplane 1 Drift	0.842	0.830	0.799	0.819	0.864
Waveform 3 Drifts + Noise	0.361	0.649	0.416	0.678	0.684
SEA 3 Drifts	0.826	0.830	0.775	0.836	0.851
Waveform Drift-less + Noise	0.733	0.736	0.439	0.738	0.761
Real Datasets					
Spam Corpus	0.777	0.822	0.869	0.815	0.888
Covertype	0.667	0.739	0.725	0.667	0.734

**Fig. 1.** Comparison: Dataset—Circle Instances—50,000 *drifting_time*—0.25*N*

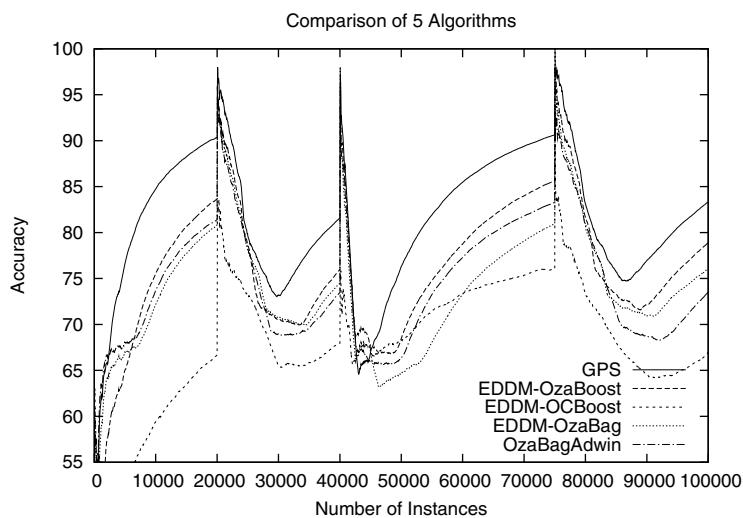


Fig. 2. Comparison: Dataset—SineH Instances—1,00,000 *drifting_time*—0.50N

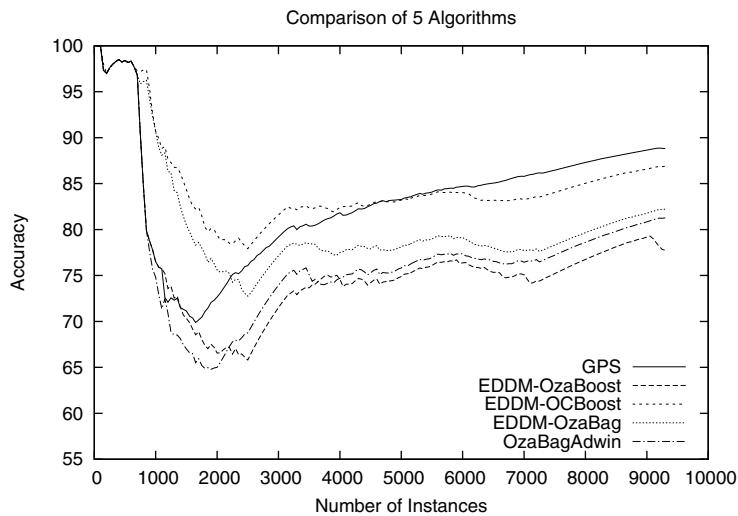


Fig. 3. Comparison: Dataset—Spam Corpus

plane dataset to test the sensitivity of the algorithms. It was found that the algorithm performs better till noise level 20% in noisy data streams. However, the performance of algorithm degrades for higher percentage of noise in the data. This is because of the presence of more noisy instances in the instance window that unnecessarily get trained during the drifting period.

5.3 Memory and Time Bounds

The GPS algorithm is single pass and incremental due to use of Interleaved Test-Then-Train model. As each instance is processed only once as soon as it arrives, the algorithm can ideally process an infinite stream of data. Analysis of the plot of time taken by the algorithm against the number of instances in the data stream show that the time taken by GPS is linear with respect to number of instances. However, GPS needs comparatively more time (in terms of constant of linearity) and memory due to two ensemble and instance window approach.

6 Conclusion and Future Work

This paper presents a new approach GPS to handle gradual concept drifts in data stream mining. This classification approach is based on zero diversity for ensemble and instance window technique. Experimental results show that the new algorithm performs better with respect to accuracy in classifying both artificial and real datasets compared to the existing approaches. Also, it reduces the recovery time for the drift.

Future work may include incorporating the approach to handle abrupt drifts as well. The approach can be implemented on parallel processors to reduce the time required for classification.

The GPS approach in data mining can be enhanced by integrating it with the Agent technology. The value of the Post Warning level and window size can be determined automatically according to the speed of drift for which agent technology can be used. This interaction of Data mining and Agents may yield more promising results. The proposed approach GPS can be integrated with software agents for various applications such as condition monitoring and alarm generation so as to enrich autonomous agents by employing the knowledge derived from data stream mining.

References

1. Baena-Garcia, M., Campo-Avila, J., Del, F.R., Bifet, A.: Early Drift Detection Method. In: Proceedings 24th ECML PKDD International Workshop on Knowledge Discovery From Data Streams (IWKDDS 2006), Berlin, Germany, pp. 77–86 (2006)
2. Bifet, A., Kirkby, R.: Data Stream Mining – A Practical Approach, <http://moa.cs.waikato.ac.nz/downloads/>
3. Breiman, L., Friedman, J.H., Olshen, R.A., Stone, C.J.: Classification and Regression Trees, p. 368. Wadsworth International Group (1984)

4. Cao, L., Gorodetsky, V., Mitkas, P.A.: Agent Mining: The Synergy of Agents and Data Mining. *IEEE Intelligent Systems* 24(3), 64–72 (2009)
5. Fern, A., Givan, R.: Online Ensemble Learning: An Empirical Study. *Machine Learning* 53, 71–109 (2003)
6. Gama, J., Medas, P., Castillo, G., Rodrigues, P.: Learning with Drift Detection. In: Bazzan, A.L.C., Labidi, S. (eds.) *SBIA 2004. LNCS (LNAI)*, vol. 3171, pp. 286–295. Springer, Heidelberg (2004)
7. Katakis, I., Tsoumakas, G., Vlahavas, I.: Tracking Recurring Contexts using Ensemble Classifiers: An Application to Email Filtering. *Knowledge and Information Systems* 22, 371–391 (2010)
8. Minku, F.L., Inoue, H., Yao, X.: Negative Correlation in Incremental Learning. *Natural Computing Journal - Special Issue on Nature-inspired Learning and Adaptive Systems*, 32P (2008)
9. Minku, L., White, A., Yao, X.: The Impact of Diversity on On-line Ensemble Learning in the Presence of Concept Drift. *IEEE Transactions on Knowledge and Data Engineering* (2008)
10. Minku, F.L., Yao, X.: Using Diversity to Handle Concept Drift in On-line Learning. *IEEE Transactions on Knowledge and Data Engineering* 99(1) (2009)
11. Oza, N.C., Russell, S.: Experimental Comparisons of On-line and Batch Versions of Bagging and Boosting. In: *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, San Francisco, California, August 26–29, pp. 359–364 (2001)
12. Oza, N.C., Russell, S.: Online Bagging and Boosting. In: *Proceedings of the 2005 IEEE International Conference on Systems, Man and Cybernetics*, vol. 3, pp. 2340–2345. Institute for Electrical and Electronics Engineers, New Jersey (2005)
13. Pelossof, R., Jones, M., Vovsha, I., Rudin, C.: Online Coordinate Boosting (2008), <http://arxiv.org/abs/0810.4553>
14. Polikar, R., Udupa, L., Udupa, S.S., Honavar, V.: Learn ++: An Incremental Learning Algorithm for Supervised Neural Networks. *IEEE Transactions on Systems, Man and Cybernetics - Part C* 31(4), 497–508 (2001)
15. Street, W.N., Kim, Y.: A streaming ensemble algorithm (SEA) for large-scale classification. In: *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining KDD 2001*, pp. 377–382. ACM Press (2001)
16. Tsymbala, A., Pechenizkiy, M., Cunningham, P., Puuronen, S.: Dynamic Integration of Classifiers for Handling Concept Drift. *Information Fusion* 9(1), 56–68 (2008)
17. UCI Repository Covertype Dataset,
<http://archive.ics.uci.edu/ml/datasets/Covertype>
18. Wang, H., Fan, W., Yu, P.S., Han, J.: Mining concept-drifting data streams using ensemble classifiers. In: *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2003*, p. 226 (2003)
19. Zliobaite, I.: Learning Under Concept Drift- An Overview, Technical Report, Faculty of Mathematics and Informatics, Vilnius University, Vilnius, Lithuania (2009)

Towards a Multiagent-Based Distributed Intrusion Detection System Using Data Mining Approaches

Imen Brahmi¹, Sadok Ben Yahia¹, Hamed Aouadi², and Pascal Poncelet³

¹ Faculty of Sciences of Tunis, Tunisia

sadok.benyahia@fst.rnu.tn

² ISLAIB, Beja, Tunisia

Hamed_aouadi@yahoo.fr

³ LIRMM UMR CNRS 5506, 161 Rue Ada, 34392 Montpellier Cedex 5, France
poncelet@lirmm.fr

Abstract. The system that monitors the events occurring in a computer system or a network and analyzes the events for sign of intrusions is known as Intrusion Detection System (IDS). The IDS need to be accurate, adaptive, and extensible. Although many established techniques and commercial products exist, their effectiveness leaves room for improvement. A great deal of research has been carried out on intrusion detection in a distributed environment to palliate the drawbacks of centralized approaches. However, distributed IDS suffer from a number of drawbacks *e.g.*, high rates of false positives, low efficiency, etc. In this paper, we propose a distributed IDS that integrates the desirable features provided by the multi-agent methodology with the high accuracy of data mining techniques. The proposed system relies on a set of intelligent agents that collect and analyze the network connections, and data mining techniques are shown to be useful to detect the intrusions. Carried out experiments showed superior performance of our distributed IDS compared to the centralized one.

Keywords: Intrusion Detection System, Multi-agents, Misuse Detection, Anomaly Detection, Data Mining Techniques.

1 Introduction

Since the cost of information processing and Internet accessibility is dropping, more and more organizations are becoming vulnerable to a wide variety of cyber threats. Therefore, it has become increasingly important to make our information systems, especially those used for critical functions such as military and commercial purpose, resistant to and tolerant of such attacks. *Intrusion Detection Systems* (IDS) are an integral part of any security package of a modern networked information system. An IDS detects intrusions by monitoring a network or system and analyzing an audit stream collected from the network or system to look for clues of malicious behavior.

Basically, two main IDS can be distinguished: *anomaly detection* and *misuse detection* [9]. Indeed, misuse detection systems [9] use patterns of well known attacks or weak spots of the system to match and identify known intrusions. However, misuse detection techniques, in general, are not effective against novel attacks that have no already matched rules or patterns [9]. On the contrary, the anomaly detection systems flag activities, that significantly deviate from the established normal usage profiles as abnormal or in other words as intrusions. Anomaly detection techniques have been shown to be effective against unknown or novel attacks since no prior knowledge about specific intrusions is required. Nevertheless, the main moan that can be addressed to the anomaly detection systems is that they tend to generate more false alarms than do misuse detection systems, *i.e.*, an anomaly can be simply a new normal behavior [32].

The conventional approaches to intrusion detection involving a central unit¹ to monitor an entire system have several drawbacks [14]. To palliate them, the dedicated research witnessed a wealthy number of works heading towards a distributed² framework of monitors that carry out local detection and provide information to perform global detection of intrusions [14].

Nevertheless, current distributed IDS also suffer from some drawbacks [19]. In fact, most commercial IDS are built in a hierarchical architecture: a tree structure with a control system at the top; information aggregation units at the internal nodes; and sensor units at the leaf nodes. Within these systems, the local intrusion detection components look for local intrusions and transmit their analysis results to the upper levels of the hierarchy. The components at the upper levels scrutinize the refined data from multiple lower level components and seek to establish a global view of the system state. Such IDS are not truly distributed systems, because of the centralized data analysis performed at the higher levels of the hierarchy [14]. Moreover, these systems suffer from the problem of single point of failure. In addition, most IDS detect attacks by analyzing information from a single host, or a single network interface, at many locations throughout the network. Thus, the designed feature of communication and cooperation between an IDS components are badly missing. This fact hampers the capability to efficiently detect large-scale distributed attacks [14].

In the past twenty years, two of most prominent, dynamic and exciting research areas: multi-agent systems [42] and data mining [12] have emerged and developed separately. On the one hand, IDS needed to be able to resist attacks on themselves, fault tolerant, highly adaptable and configurable [29]. Given these characteristics, the multi-agent technology seemed to be an appropriate alternative for developing IDS [18]. Indeed, a distributed IDS based on multi-agent technology can effectively improve the detection accuracy, the detection speed,

¹ An IDS is considered as centralized whenever the analysis of data is performed at a fixed number of locations, independent of how many hosts are being monitored [39].

² An IDS is considered as distributed whenever the analysis of the data is performed in a number of locations proportional to the number of hosts that are being monitored [39].

and enhance the system's own security [19]. Several IDS based on the multi-agent methodology were developed [4,22,19,29,39]. On the other hand, along with the increasing complexity of networks, protecting a system against new and complex attacks, while keeping an automatic and adaptive framework, is a thriving issue within the intrusion detection domain. One answer to the problem could rely on data mining techniques [6,28]. Following this trend, a wide variety of data mining techniques have been successfully applied into the intrusion detection domain [3,8,20,24,37].

In this paper, we investigate another way of tackling the aforementioned problems. Thus, we introduce a new distributed IDS, called MAD-IDS (*Multi-Agent using Data mining based Intrusion Detection System*). MAD-IDS is based on the integration of the multi-agents technology and the data mining techniques. In this respect, our proposed system uses a set of agents that can be applied to a number of tasks, namely: data capturing, detecting the known and unknown attack categories, extracting the set of signatures-based rules and ultimately alerting the administrator. Moreover, MAD-IDS efficiently merge both anomaly and misuse detection strategies by exploiting the advantages of the one to palliate the limitations of the other and vice versa. The performance of the IDS can be improved by combining anomaly and misuse analysis [10,17,33]. The main thrust of our approach stands in the use of an unsupervised anomaly detection technique based on the clustering algorithm. Furthermore, given the very high volume of connections observed per unit time, the association rule mining technique is shown to be useful in enabling a security analyst to understand and characterize emerging threats. The obtained model is more effective since the integration of multi-agent technology and data mining techniques makes an IDS more autonomous and efficient. Through extensive carried out experiments on intrusion detection benchmark datasets and real life network traffic, we show the effectiveness of our proposal in terms of (i) the detection delay; (ii) the scalability-related criteria such as network bandwidth and system response time; and (iii) the detection ability.

The remaining of the paper is organized as follows. Section 2 sheds light on the related work that focuses on the integration of the multi-agent systems with the data mining techniques. We introduce our new distributed intrusion detection system based on the multi-agent technology in Section 3. We also relate the encouraging results of the carried out experiments in Section 4. Finally, Section 5 concludes and points out avenues of future work.

2 Scrutiny of the Related Work

In recent years and within the intrusion detection domain, an increasingly evident trend has emerged. The trend stands within the crossroads of multi-agent systems and data mining. Its development has reached to the level as a new and promising research area. In this respect, various distributed intrusion detection architectures using the multi-agent design methodology and the data mining techniques have been developed.

One of the well known examples of applying a combination of multi-agents methodology and the data mining techniques is JAM (*Java Agents for Meta-learning*) [40]. The original idea behind JAM was the use of data mining techniques to correlate knowledge to build better models for fraud and intrusion detection. Thus, JAM uses association rules and frequent episode mining algorithms to determine the relationships between the different fields in audit trails. Moreover, a meta-learning classifier learns the signatures of attacks. For instance, JAM seemed to be shared between multiple organizations. However, it suffers from serious shortcomings that include the increase demand for runtime system resources and the inability to combine multiple models computed over distributed datasets with different schemas.

With the aim to improve the JAM's approach, Helmer et al. introduced in [16] a distributed IDS that integrates the data mining techniques, as well as mobile and static agents. Indeed, static agents collect the information and transform it in a common format. Besides, the mobile agents travel between monitored systems in a network, to obtain the information from the static agents, classify, correlate and report it to a user interface and a database via mediators. Finally, data mining agents use machine learning approaches towards an automated discovery of concise rules from system logs and audit data, to facilitate building, monitoring, and analyzing intrusions on distributed systems. In contrary to JAM, the proposed IDS focuses on data mining within an organization. It uses a different data representation that provides a single signature for each process.

Motivated by the issue that most of the existing commercial network-based IDS products are signature-based but not adaptive, Lui et al. [25] proposed an adaptive system using various data mining techniques and agent architecture. The data mining approaches include: the clustering, association and sequential rule extraction algorithms. These algorithms are used to accurately capture the actual behavior of the network traffic, and the mined portfolio is useful for differentiating “normal” and “attack” traffics.

Zhang et al. [44] combined the advantages of agent-based distributed analysis and clustering-based intrusion detection technique. The proposed approach focuses on the use of the clustering algorithm: (i) to select the candidate anomalies at the level of IDS agents; and (ii) to choose the true attack at the central IDS. However, the results of the clustering algorithm depend on the choice of the values of the cluster width. Whenever the latter is an unappropriate value, then the algorithm might label some intrusions as “normal”, and some normal ones as “intrusion”.

Moreover, Rehák et al. [35] introduced a prototype of agent-based IDS designed for deployment on high-speed backbone networks. The main contribution of the system, called CAMNEP, is the integration of several anomaly detection techniques by means of collective trust modeling within a group of collaborative detection agents, each featuring a specific detection algorithm. These algorithms maintain a model of expected traffic on the network and compare it with real traffic to identify the discrepancies that are identified as possible attacks.

However, CAMNEP is not able to detect attacks that consist of few packets, e.g., *Buffer Overflow* attack.

Palomo et al. [30] proposed an approach that integrates the Self-Organization Map (SOM) clustering approach within a multi-agents system. Indeed, the approach describes a multi-agents system with capabilities to analyze and discover knowledge gathered from distributed agents. These enhanced capabilities are obtained through a dynamic SOM and a multi-agents based communication system. The central administrator agent dynamically obtains information about the intrusions from the distributed agents and maintains a knowledge pool using a proposed growing SOM.

Recently, Shyu and Sainani [38] proposed a data mining assisted multi-agent-based IDS, called DMAS-IDS. DMAS-IDS employs three layers, called *Host*, *Classification* and *Manager layers*. Each one of these layers comprises agents capable of communicating the obtained results with each other. The agents are facilitated with a supervised classification algorithm called the *Collateral Representative Subspace Projection Modeling* (C-RSPM). However, C-RSPM is an ineffective classification algorithm with datasets possessing a very high number of classes, especially when the classes' distributions are very similar. In addition, the results of C-RSPM rely on a labeled training data instances which is not suitable for an on-line IDS.

In this paper, we propose a new distributed IDS, called MAD-IDS, which integrates the data mining techniques and the multi-agent technology. Unlike most of the surveyed works, where only one strategy is used for the detection of various attacks; MAD-IDS exploits the advantages of both misuse and anomaly detection strategies by merging them to remedy the corresponding disadvantages. Particularly, MAD-IDS focuses on the use of: (i) an unsupervised anomaly-based clustering technique to detect the unknown intrusions; and (ii) an association rule mining algorithm to summarize the network connections that are classified as anomalous.

3 The MAD-IDS System

The distributed structure of MAD-IDS is composed of different cooperative, communicant and collaborative agents for collecting and analyzing massive amounts of network traffic, called respectively: *Sniffer*, *Filter*, *Misuse Detection*, *Anomaly Detection*, *Rule Mining* and *Reporter Agent*. Figure 1 sketches the overall architecture of MAD-IDS.

The processing steps of MAD-IDS can be summarized as follows:

1. The Sniffer Agent is the first agent that connects to the network and gathers the packets;
2. The gathered packets are send to the Filter Agent which filters them;
3. The Misuse Detection Agent analyzes the collected and filtered packets, to detect the network connections that correspond to attacks for which signatures are available, and then to remove them from further analysis;

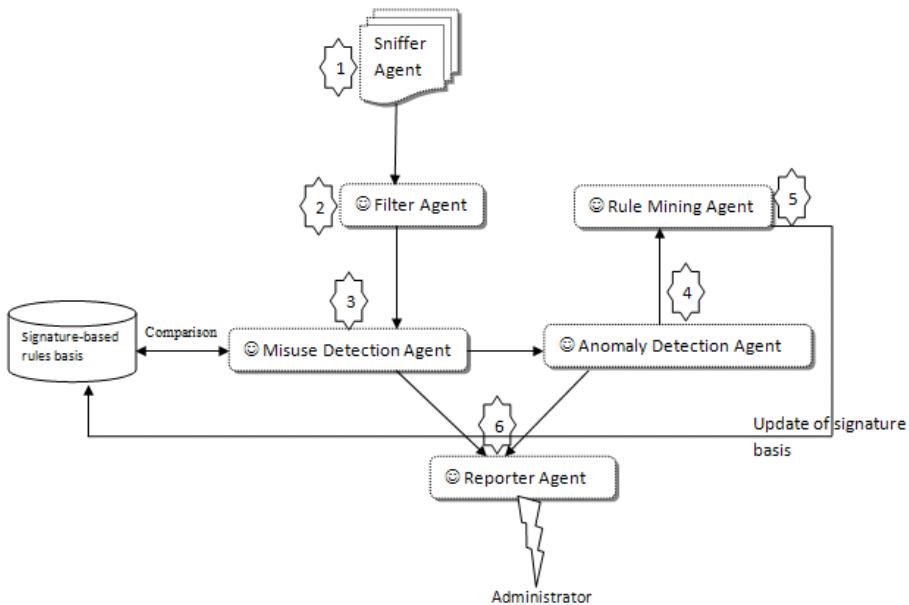


Fig. 1. The architecture of MAD-IDS

4. The remaining network packets are fed into an Anomaly Detection Agent, which uses the clustering technique to detect the abnormal connections;
5. The Rule Mining Agent aims at providing a concise representation of the network traffic. Typically, it summarizes the network connections that are selected as anomalous by the Anomaly Detection Agent. The obtained set of generic association rules can be periodically fed to the Misuse Detection Agent to update its signature database allowing the detection of known attacks;
6. Finally, the Reporter Agent generates reports and logs.

Each of these agents is individually described in the following subsections.

3.1 The Sniffer Agent (SA)

A sniffer is a device that is able to intercept and log traffic passing over a network. It allows the capture of each packet and, if needed, it analyzes its content. The traffic can be IP, IPX, or AppleTalk network packets. In general, the sniffing can be used to: *i*) analyze network problems; *ii*) detect network intrusion attempts; and *iii*) documenting regulatory compliance through logging all perimeter and endpoint traffic; etc.

Firstly, the Sniffer Agent (SA) captures the incoming packets by reading them from the network card in the machine and caches them in the memory at the interval of every 5 seconds. The benefits of this kind of agents include: *i*) the

cloning and the distribution throughout the network; and *ii)* the duplication in order to lighten the network charge. Finally, the captured packets will be the input of the next agent, *i.e.*, the Filter Agent.

3.2 The Filter Agent (FA)

A distributed IDS must undertake to analyze a huge volumes of events collected from different sources around the network. Consequently, the FA filters the packets already captured by the SA. It will treat these crude packets by achieving the following tasks:

- Distinguish the various fields of the packets collected in crude such as destination address and the protocol;
- Sort the packets by the category of packets (TCP, UDP, ICMP, etc.) concerned by a specific kind of intrusion.

The FA performs its tasks as a pretreatment phase, which precedes the analysis phase carried out by the following agent.

3.3 The Misuse Detection Agent (MDA)

This kind of agents analyzes the packets captured by the Sniffer Agent and then pre-processed by the FA. In fact, MDA searches for attack signatures³ in these packets. Hence, if there is a similarity between the filtered packets and attacks signatures, then the agent raises an alert to the Reporter Agent, and then removes these anomalous packets from further analysis.

Within MAD-IDS, the MDA detects known attacks in the network connections, using a rule set based on the intruder signatures as well as an efficient pattern-matching using the RETE algorithm [13]. Attack signatures are a specific rule set provided by the Rule Mining Agent. Thus, each rule has the following structure:

1. ***Rule antecedent part:*** Contains the basic information about the rule, including:
 - a **Protocol:** The protocol used by the packet being analyzed.
 - b **Source information:** IP address and port of the source computer that originated the packet. It is also possible to ignore these addresses to apply the rule on all packets irrespective of the IP address or the port number;
 - c **Destination information:** IP address and port of the destination computer in the packet.
2. ***Rule conclusion part:*** Indicates the alert that will be generated whenever the rule antecedent conditions are met.

³ An attack signature is a known attack method that exploits the system vulnerabilities and causes security problem.

Example 1. Let consider the rule: $\{Protocol=TCP, Src_IP=206.163.37.95, Dst_Port = 139\} \Rightarrow \{Attack\}$. This rule will generate an alert, each time a packet using the IP source $\{206.163.37.95\}$ and the destination port $\{139\}$ is detected for any IP destination and for any source port.

Even though, the known attacks are detected, it remains nevertheless the problem of the new attacks detection. In this respect, to protect a system against new and complex attacks, while building an automatic IDS, researchers are increasingly looking at using data mining techniques for anomaly detection [7,32,33,34]. In particular, the clustering technique can be considered as one of the most important unsupervised learning algorithms [5,7,15,45].

Thus, we introduce an anomaly detection agent based on the clustering technique. The agent is described in the following.

3.4 The Anomaly Detection Agent (ADA)

The Anomaly Detection Agent provides the combination of distributed IDS with the anomaly-based clustering technique.

Clustering is a widely used data mining technique [11,26] which groups similar items, to obtain meaningful groups/clusters of data items in a data set. These clusters represent the dominant modes of behavior of the data objects determined using a similarity measure. A data analyst can get a high level understanding of the characteristics of the data set by analyzing the clusters.

Clustering provides an effective solution to discover the expected and unexpected modes of behavior and to obtain a high level understanding of the network traffic [7]. The main advantage that clustering affords is the ability to learn and detect intrusions in the audit data, while not requiring the system administrator to provide explicit descriptions of various attack classes/types [32]. The idea behind this technique is that the amount of normal connection data is usually overwhelmingly larger than the number of intrusions [15,32]. Whenever this assumption holds, the anomalies and attacks can be detected based on cluster sizes, *i.e.*, large clusters correspond to normal data, and the rest of the data points, which are outliers, correspond to attacks [15,32].

Indeed, the ADA detects the abnormal packets using an unsupervised anomaly-based clustering algorithm that we introduce, called *AD-CLUST* (*Anomaly Detection-based Clustering*) [5]. *AD-CLUST* combines two prominent categories of clustering, namely: distance-based as well as density-based (*e.g.* K-MEANS [26] and DBSCAN [11], respectively). It exploits the advantages of one to palliate the limitations of the other and vice versa. On the one hand, K-MEANS is fast, easy to implement and not memory greedy. However, it is easily affected by the noise⁴ and suffers from a greater time complexity, which becomes an extremely important factor within intrusion detection due to the very large dataset sizes [34]. Moreover, the *number of clusters dependency* and

⁴ A noisy data point does not belong to any cluster properly [11].

the *degeneracy* constitute the drawbacks that hamper the use of K-MEANS for anomaly detection [15]. On the other hand, while the density-based clustering can solve the problem of the number of clusters dependency and handles noise well, it is not suitable as a stand-alone tool for intrusion detection. Indeed, the density-based clustering produces the suboptimal partitions of the dataset and puts lots of instances into noises which are dropped out, which makes the detection rates and the false positif rates worse [45]. Thus, the *AD-CLUST* algorithm enables DBSCAN to efficiently handle very large data and improves the quality of the K-MEANS algorithm by removing the noisy data and solving the problem of the number of clusters dependency.

The processing steps of our algorithm *AD-CLUST* can be summarized as follows [5]:

1. Extraction of the density-based clusters that are considered as candidate initial cluster centers. The density-based clustering is used as a preprocessing step for the *AD-CLUST* algorithm;
2. Compute the Euclidean distance between the candidate cluster center and the instance that will be assigned to the closest cluster. For an instance x_i and a cluster center z_i , the Euclidean distance is defined as:

$$\text{distance}(x_i, z_i) = \sqrt{\sum_{i=1}^n (x_i - z_i)^2} \quad (1)$$

3. The size of a neighborhood of instances is specified by an input parameter. We use k' parameter to distinguish it from the k parameter used by the K-MEANS algorithm. Hence, k' specifies the minimal number of instances in a neighborhood and controls the granularity of the final clusters of the clustering-based density. If k' is set to a large value, then a few large clusters are found. To reduce the number of candidate clusters k' to the expected number k , we can iteratively merge the two most similar clusters. Otherwise, if k' is set too small, then many small clusters will be generated. The clusters will be split, new clusters will be created to replace the empty ones and the instances will be re-assigned to existing centers. This iteration will continue until there is no empty cluster. Consequently, the outliers⁵ of clusters will be removed to form new clusters, in which instances are more similar to each other. In this way, the value of initial cluster centers k will be determined automatically by splitting or merging clusters;
4. Within the detecting phase, the *AD-CLUST* algorithm performs the detection of intrusions. Thus, for each novel instance I the algorithm proceeds as follows:
 - (a) Compute the Euclidean distance and find the cluster with the shortest distance to I .

⁵ An outlier is defined as an object where an instance p of the dataset is further than distance D from the object, where p and D are parameters specified by the users.

- (b) Classify I by the category of the closest cluster. Clearly, if the distance between I and the cluster of “normal” is the shortest one, then I will be a normal instance. Otherwise, I is an intrusion.

The pseudo-code is shown by Algorithm 1 and the used notations are summarized in Table 1.

Table 1. List of used notations in the *AD-Clust* algorithm

\mathcal{D}	: The network dataset which contains n packets $\{p_1, \dots, p_n\}$.
ε	: The Euclidean neighborhood radius.
η	: The minimum number of neighbors required in ε .neighborhood to form a cluster.
N	: The set of points in ε .neighborhood of p_i
C	: The set of clusters that outputs the algorithm.
Z	: The final cluster centers.
k	: The number of clusters to be found.
C'	: The clusters initially found by density based clustering algorithm.
Z'	: The initial cluster centers.
k'	: The number of clusters initially found by density based clustering algorithm.

Firstly, the algorithm starts by the use of the density-based clustering algorithm to produce a set of initial candidate clusters (*cf.* lines 2-14). Thus, it determines the neighborhood of each instance. If the neighborhood of instances is smaller than η , then a new cluster is created. Otherwise, the current instance is assigned to noise. The resulted clusters are considered as candidate initial cluster centers. The algorithm calculates the distance between the instances and the centers of the candidate clusters. Next, each instance will be assigned to the closest cluster (*cf.* lines 15-17). In addition, *AD-CLUST* splits or merges the clusters to automatically determine the value of initial cluster centers k (*cf.* lines 18-26). Finally, a new set of initial cluster centers is obtained and used in the K-MEANS algorithm to classify the instances into ‘normal’ clusters and ‘abnormal’ clusters. (*cf.* lines 27-28).

Once the anomaly network connections are selected, the Rule Mining Agent is ready to carry out its task, which is described in the following subsection.

3.5 The Rule Mining Agent (RMA)

Even though improving the detection rate and reducing the false alarm rate for attacks is an objective of paramount importance for an IDS, this task alone is not sufficient if the rate of network traffic is very high. For example, suppose a security analyst examines the output of an anomaly detector every 10 minutes during which two million connections are established. Even if a hundred of these connections are reported as anomalous, it is unfeasible for the analyst to examine

Algorithm 1. The *AD-CLUST* algorithm

```

Input:  $\mathcal{D}$ : A network dataset.
Output: C: The set of classified clusters.

1 Begin
2   C' := 0 ;
3   k' := 0 ;
4   Foreach  $p_i \in \mathcal{D}$  do
5     | Compute the number of points N in the neighborhood of  $p_i$  defined
       | with  $\varepsilon$  ;
6     | If  $N < \eta$  then
7       |   | Mark  $p_i$  as noise ;
8     | Else
9       |   | Add  $p_i$  to cluster C' ;
10      |   | C' := C' + 1 ;
11      |   | k' := k' + 1 ;
12      |   | return (N) ;
13    | // Now we obtain C' clusters based on density.
14   Foreach cluster  $C'$  do
15     | Find the cluster centers  $z'_j$  by computing the mean ;
16     | Assign each instance  $p_i$  to the closest cluster such that  $C'(p_i) =$ 
       | min_distance( $p_i, z'_j$ );
17   repeat
18     | If  $k' > k$  then
19       |   | Select two clusters based on density and join them;
20       |   | Find the new cluster center  $z'_j$  ;
21       |   | // Finally we will have  $k$  centers
22     | Else
23       |   | Select a cluster based on density and split it using the K-MEANS
             |   | clustering algorithm ;
24       |   | Find the new cluster center  $z'_j$  ;
25       |   | //Finally we will have  $k$  centers ;
26   until achieving  $k$  clusters;
27   Z := Z';
28   C := K-MEANS( $k, Z, \mathcal{D}$ );
29 End

```

them in 10 minutes. Thus, there is a need for an approach whereby anomalous connections can be summarized into patterns that capture the essence of the anomalous behavior. In addition to making the human analyst's task manageable, this also had the advantage of providing templates from which signatures of novel attacks can be built for augmenting the database of signature-based IDSs.

In this respect, the RMA uses the association rule mining to achieve summarization of detected attacks. The formalization of the association rule mining problem was initially introduced by Agrawal et al. [1]. Given a set of records, the

objective of mining association rules is to extract all rules of the form $\mathcal{X} \Rightarrow \mathcal{Y}$ that satisfy a user-specified minimum support and minimum confidence thresholds. In fact, \mathcal{X} is the premise of the rule and \mathcal{Y} is its conclusion.

In the intrusion detection context, the association rule mining have been found to be valuable for analyzing a network traffic data [24,41,43]. Often times, the number of anomalous connections flagged by an IDS can be very large, thus requiring analysts to spend a large amount of time interpreting and analyzing each connection. By applying the association rule mining techniques, analysts can obtain a high-level summary of anomalous connections [7].

Generally, standard association rule mining technique relies on a user-specified minimum support threshold to eliminate patterns that occur infrequently in the data. For a network intrusion data, the proportion of network traffic that corresponds to an attack is considerably smaller than that of normal traffic [7]. As a result, we should set the *minsupp*⁶ value to a very low value to detect patterns involving the attack class [23]. However, this fact will degrade the performance of the association rule mining algorithm considerably and produces an overwhelming large number of rules for the normal class.

Although the association rules can detect sets of features that occur frequently in the network traffic data, the number of mined rules can be quite large, depending on the value of *minsupp*, which affects the speed of IDS and hampers its whole performance [7]. Some of these rules are redundant since they contains patters that correspond to the subsets of other patterns.

Example 2. Given two association rules:

1. $R_1: Protocol=TCP, Dst_Port=8888 \Rightarrow Attack$
2. $R_2: Dst_Port=8888 \Rightarrow Attack$

The first association rule is more descriptive than the second one. The premise of R_2 , $Dst_Port=8888$, is a subset of the premise of R_1 . If the support of these two rules is close, then R_2 is considered as redundant.

As consequence, to generate association rules without redundancy, we apply the *generic association rule basis*, which provides a concise representation of association rules introduced in [31]. Particularly, we apply the *Informative Generic Basis (IGB)* [2]. In addition to the elimination of redundancy, the application of the *IGB* basis during an intrusion detection process provides the increase of the overall coverage of detectable attacks.

Thus, once the anomalous connections are detected by the ADA, then the RMA is ready to mine the generic association rule set using *IGB*. The benefit is the reduction of information overloading for the security analysts. The extracted rule set may be a candidate signature-based rule for addition to the signature basis used by the MDA. This means that the database of signatures is updated regularly in order to ensure an adequate protection.

⁶ *minsupp* is the minimum support threshold pre-defined by the user.

3.6 The Reporter Agent (RA)

The MDA and the ADA report their findings to the RA which transmits them to the administrator. Whenever an intrusion is detected, it will send an alert to the system administrator. This alert can be a message on the screen or a message to a centralized machine or an alert file.

4 Experimental Results

In order to assess the overall performance of MAD-IDS in a realistic scenario, a prototype of the proposed architecture was implemented using Sun's Java Development Kit 1.4.1, the well known platform JADE 3.7, the Eclipse and the JPCAP 0.7.

JADE (*Java Agent DEvelopment Framework*)⁷ is a software Framework, which simplifies the implementation of multi-agent systems. The agent platform can be distributed by moving agents from one machine to another one.

In addition, JPCAP (*Java library for CAPturing and sending network Packets*)⁸ is an open source library for capturing and sending network packets.

All experiments were carried out on equivalent machines equipped with a 3GHz Pentium IV and 2GB of main memory running under Linux Fedora Core 6.

Through the carried out experiments, we have a twofold aim: first, we focus on the assessment of the *AD-CLUST* detection ability. Second, we have to stress on evaluating the overall performance of the MAD-IDS system.

4.1 *AD-Clust* performances assessment

To assess the performance of the *AD-CLUST* algorithm, we choose to compare *AD-CLUST* vs. K-MEANS and DBSCAN algorithms respectively. During experiments, we partly use the common benchmark, namely, the KDD99 dataset⁹. The dataset provides a standard corpus for evaluating intrusion detection algorithms. It also introduced more insider attacks. The attacks can roughly split into four main categories:

1. *DoS (Denial of Service);*
2. *R2L (Remote to User);*
3. *U2R (Unauthorized Access to Root);*
4. *Probing (Surveillance and Probing).*

Indeed, Table 2 shows the distributions of record types in training and testing datasets, used during our experiments. The first row shows the numbers of normal network packets, while the second row gives the distributions of the network attacks.

⁷ Available at: <http://jade.tilab.com>

⁸ Available at: <http://netresearch.ics.uci.edu/kfujii/jpcap/doc/>

⁹ Available at: http://www.ll.mit.edu/IST/ideval/data/data_index.html

Table 2. The considered KDD99 datasets

Record Type	Training Set	Testing Set
Normal	897277	1260592
Intrusion	116743	180436
Total	1014020	1441028

Generally, to evaluate the detection ability of the intrusion detection algorithms, two interesting metrics are usually of use [34]: the *Detection Rate* (DR) and the *False Positive Rate* (FR).

1. The DR equals the number of correctly detected intrusions divided by the total number of intrusions in the dataset;
2. The FR equals the total number of normal instances that were incorrectly considered as attacks divided by the total number of normal instances in the dataset.

The value of the DR is expected to be as large as possible, while the value of the FR is expected to be as small as possible.

Table 3 sketches the DR and FR for the considered datasets, using, respectively the K-MEANS, the DBSCAN and the *AD-CLUST* algorithms.

Table 3. The DR and FR of *AD-CLUST* vs. K-MEANS and DBSCAN

Algorithm	k	DR (%)	FR (%)
K-means	5	26.29	0.32
	23	57.52	3.09
	300	87.98	8.90
	600	97.19	12.50
	700	99.11	15.92
	621	94.64	24.58
AD-Clust	621	99.12	1.41

Indeed, we test K-MEANS using different numbers of the initial cluster centers k . According to Table 3, we remark that both DR and FR increase as far as the number of clusters increases. However, this relationship is not suitable, since within an effective anomaly detection algorithm the FR should decrease along with the increase of both number of clusters and DR [15]. In addition, whenever the values of k are equal to 5, 23 and 300, the DR value is low. Otherwise, when the value of k is equal to 600, the DR increases to 97%. The increase of DR is due to the partition optimal of K-MEANS. Thus, within the K-MEANS algorithm, we have to try different values of k in order to find the optimal one.

In addition, Table 3 shows that the DR of *AD-CLUST* is 99.12% and is greater than that of DBSCAN. The FR of *AD-CLUST* is only 1.41% and is by far lower

than that of DBSCAN. The bad results of DBSCAN can be explained by the suboptimal partitions of the dataset and the consideration of several instances as noises which are dropped out.

Finally, Table 3 shows that the *AD-CLUST* algorithm automatically partitions the same data set into 621 clusters. Moreover, it allows the reduction of FR along with the increase of DR.

During the following experiments, we set the value of k of K-MEANS at the optimal value, *i.e.* 621.

The rates DR and FR change in relation to each other. Whenever the DR is the highest, the FR is the lowest, and vice versa. Consequently, these two metrics can be of use to plot a ROC curve (*Receiver Operating Characteristic*) [27]. Figure 2 (Right) compares the ROC curve of *AD-CLUST* *vs.* those of K-MEANS and DBSCAN. The ROC curve assesses the accuracy of the intrusion detection algorithm [27]. Thus, we conclude that *AD-CLUST* is more accurate than K-MEANS and DBSCAN.

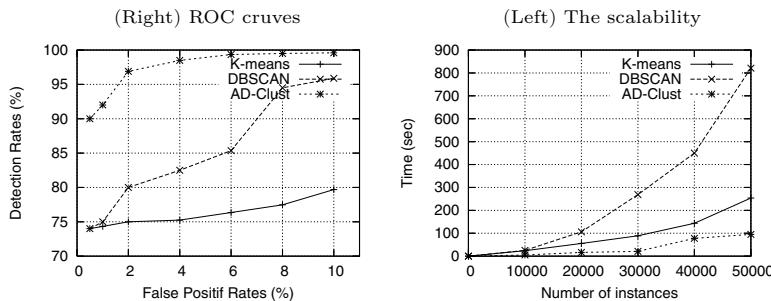


Fig. 2. Accuracy and scalability of *AD-CLUST* *vs.* K-MEANS and DBSCAN

In addition, we tested the scalability of *AD-CLUST*, DBSCAN and K-MEANS. Figure 2 (Left) plots the running time against different numbers of data points. These results show that the running time of the three algorithms linearly increases with the number of instances. Thus, we can conclude that *AD-CLUST* is faster than both DBSCAN and K-MEANS.

4.2 Overall Performance of the MAD-IDS System

In the following, the performance of the entire MAD-IDS architecture will be evaluated. In this respect, we used machines that were connected via a switch, thus forming a switched network. For a realistic testing environment, attacks needed to be interjected into a volume of network traffic. Consequently, we simulated attacks using the well known tool *Metasploit*¹⁰ version 3.5.1. Metasploit is both a penetration testing system and a development platform for creating

¹⁰ Available at: <http://www.metasploit.com/>

security tools. It is used by the security researchers world-wide to test an IDS. The description of the eight different attack types used in the evaluation is shown in Table 4.

Table 4. The simulated attacks at a glance

Attack Name	Description	
attack1: DoS Smurf	ICMP echo reply flood, caused by an ICMP echo packet with spoofed address (of victim) sent to a network broadcast address.	
attack2: Backdoor Back Office	A remote administration tool that allows almost complete control over a computer by the remote attacker.	
attack3: SPYWARE-PUT Hijacker	The spyware Hijacker is a type of malware that can be installed on computers, and which collects small pieces of information about users without their knowledge.	
attack4: Nmap Scan	TCP	Scans many ports to determine available services on a single host using UDP packets.
attack5: Finger User	Allows an attacker to disrupt a network using the redirection capability in the finger daemon.	
attack6: RPC Linux Statd Over-flow	Linux	Buffer overflow vulnerability exists making it possible for malformed requests by an attacker to be devised giving root privileges.
attack7: DNS Zone Transfer	Zone	DNS server provides information for all DNS resource records registered with DNS server that can be used by attackers to better understand a network.
attack8: HTTP IIS	IIS	An attacker could send a specially crafted URL containing Unicode characters to access files and folders on the Web server with the privileges of the user account.

We have conducted evaluations in terms of (*i*) the detection delay; (*ii*) the scalability-related criteria such as network bandwidth and system response time; and (*iii*) the detection ability. During the evaluations, we compare the results of the MAD-IDS system *vs.* that of SNORT. SNORT is an open source network IDS. It is able to perform the analysis of network traffic in a real-time using a rule-driven language, which combines the benefits of signature, protocol and anomaly based inspection methods [36].

Multi-agents *vs.* Centralized IDS. Firstly, we answer to the question: why the realization of the multi-agents IDS is advantageous ? Indeed, Figure 3 (Right) plots the detection delay against the number of packets, using the MAD-IDS and SNORT systems. Clearly, the results show that the detection delay of both

systems linearly increases with the number of packets. Thus, the figure highlights that our proposed system MAD-IDS is faster than the system SNORT.

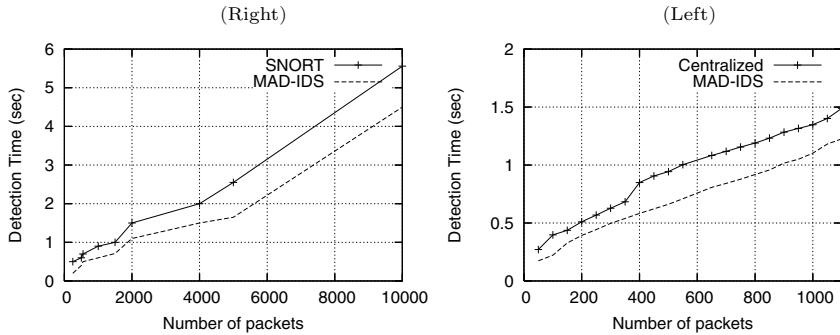


Fig. 3. Multi-agents *vs.* centralized IDS in terms of detection delay

Moreover, we also implemented a centralized IDS with local sensor that forward filtered data to a central analysis node. Figure 3 (Left) shows the detection time needed, using the MAD-IDS system *vs.* that of need by a centralized one. Thus, our proposed distributed IDS is much faster than the centralized one. This can be explained by the fact that agents operate directly on the host, where an action has to be taken, their response is faster than systems where actions were taken by the central coordinator.

Bandwidth Consumption and Response Time. It is known that two of the most important elements of network performance are bandwidth and latency. On the one hand, the bandwidth is the transmission capacity of the network, usually measured in bits per second.

On the other hand, the network latency is the amount of time it takes for a packet to travel from the source to the destination. We use latency to describe the amount of time it takes once an attack takes place till it gets resolved.

As depicted in Figure 4 (Right), the maximum bandwidth consumed by MAD-IDS is 0.06 Mbits/sec, which is very low as well. This makes our proposed system low cost which is definitely a desirable feature for any distributed system.

Figure 4 (Left) illustrates the network latency, *i.e.*, the response times required by MAD-IDS with respect to the attack types. According to this figure, the detection of all attack types, on average, result in very similar and low response time. This is definitely a desirable feature.

In conclusion, it is clear from the obtained results that in our proposed architecture, the performance of the system will not deteriorate too much with the increase in the number of attacks, which is justified by its low bandwidth consumption and quick response time behavior. Also, in case of more machines are connected to the network, the MAD-IDS system still withstand the load and swiftly deliver the results.

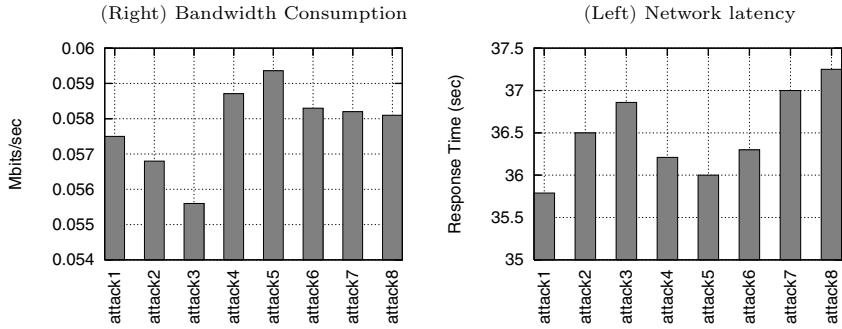


Fig. 4. The bandwidth Consumption and the network latency of MAD-IDS

Detection Ability. According to Figure 5 (Right), we can see that the false alarm rates of our adaptive system is significantly lower compared to that of SNORT. For instance, adaptive mechanisms used by the agents can change normal profiles correspondingly, enabling MAD-IDS to better suit the environment. Consequently, false alarms can be reduced correspondingly.

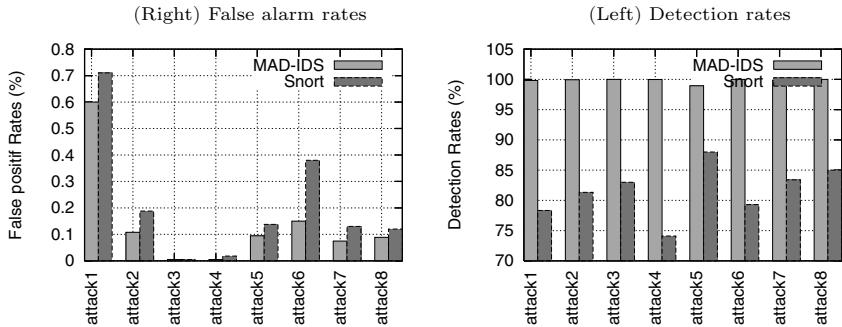


Fig. 5. Detection ability of MAD-IDS vs. SNORT

This result is confirmed by Figure 5 (Left) that shows that the detection rates of MAD-IDS is by far better than SNORT configuration.

Knowing that a main challenge of existing IDSs is to decrease the false alarm rates, the main benefit of our adaptive system is to lower the false alarm rate, while maintaining a good detection rate.

5 Conclusion

In this paper, a novel distributed IDS, called MAD-IDS was introduced. MAD-IDS incorporates the desirable features of the multi-agents methodology with

the highly accurate data mining techniques. On the one hand, the multi-agent system is efficient for enhancing security, flexibility and cooperative detective ability of distributed IDS. On the other hand, the MAD-IDS system uses the clustering and the association rules techniques to analyze large network packets more intelligently and automatically. In fact, the anomaly-based clustering technique, particularly the \mathcal{AD} -CLUST algorithm, will try to automatically determine which data instances fall into the normal class and which ones are intrusions. The \mathcal{AD} -CLUST algorithm was able to detect a large number of intrusions while keeping the false rate reasonably low. As consequence, our system addresses the specific limitations described by the central approaches and the monolithic systems. It performs a distributed search and analysis using multi-agent system, eliminating then the single point of failure drawbacks. In addition, MAD-IDS guarantees that the system can be extended in a straightforward manner. For example, new task agents can be added after the MAD-IDS is initially deployed, without requiring any changes to the existing set of agents. The experimental results showed that MAD-IDS is an efficient system that yields promising results, indicating a high accuracy, a good scalability and a low response time.

Future work include the following issues:

- First, the network data is temporal (*i.e., streaming*) in nature, and the development of algorithms for mining data streams is necessary for building real-time IDS;
- Second, the low frequency of computer attacks requires modification of standard data mining algorithms for their detection;
- Finally, new approaches have been suggested using ontologies as a way to represent and understand the attacks domain knowledge, expressing the IDS much more in terms of their domain and performing intelligent reasoning [21]. Thus, as future work, we need to integrate ontology within MAD-IDS. This fact can provide agents with a common interpretation of the environment signatures which are matched through the data structure based on the MDA.

References

1. Agrawal, R., Imielinski, T., Swami, A.: Mining Association Rules Between Sets of Items in Large Databases. In: Proceedings of the International Conference on Management of Data, Washington, D.C, pp. 207–216 (1993)
2. Ben Yahia, S., Gasmi, G., Nguifo, E.M.: A New Generic Basis of Factual and Implicative Association Rules. Intelligent Data Analysis 13(4), 633–656 (2009)
3. Bouzida, Y., Cuppens, F.: Detecting known and novel network intrusion. In: Proceedings of the 21st IFIP International Conference on Information Security, Karlstad, Sweden, pp. 258–270 (2006)
4. Brahmi, I., Yahia, S.B., Poncelet, P.: A Snort-Based Mobile Agent For A Distributed Intrusion Detection System. In: Proceedings of the International Conference on Security and Cryptography, Seville, Spain (to appear, 2011)
5. Brahmi, I., Yahia, S.B., Poncelet, P.: \mathcal{AD} -CLUST: Détection des Anomalies Basée sur le Clustering. In: Atelier Clustering Incrémental et Méthodes de Détection de Nouveauté en conjonction avec 11ème Conférence Francophone d’Extraction et de Gestion de Connaissances EGC 2011, Brest, France, pp. 27–41 (2011)

6. Chalak, A., Bhosale, R., Harale, N.D.: Effective data mining techniques for intrusion detection and prevention system. In: Proceedings of the International Conference on Advanced Computing, Communication and Networks 2011, Chandigarh, India, pp. 1130–1134 (2011)
7. Chandola, V., Eilertson, E., Ertoz, L., Simon, G., Kumar, V.: Data Mining for Cyber Security. In: Singhal, A. (ed.) Data Warehousing and Data Mining Techniques for Computer Security, pp. 83–103. Springer, Heidelberg (2006)
8. Christine, D., Hyun Ik, J., Wenjun, Z.: A New Data-Mining Based Approach for Network Intrusion Detection. In: Proceedings of the 7th Annual Conference on Communication Networks and Services Research, Moncton, New Brunswick, Canada, pp. 372–377 (2009)
9. Debar, H., Dacier, M., Wespi, A.: Towards a Taxonomy of Intrusion-Detection Systems. *Computer Networks* 31, 805–822 (1999)
10. Depren, O., Topallar, M., Anarim, E., Ciliz, M.K.: An Intelligent Intrusion Detection System (IDS) for Anomaly and Misuse Detection in Computer Networks. *Expert System with Applications* 29, 713–722 (2005)
11. Ester, M., Kriegel, H.-P., Sander, J., Xu, X.: A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In: Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining, Portland, Oregon, pp. 226–231 (1996)
12. Fayyad, U., Piatetsky-Shapiro, G., Smyth, P.: The KDD Process of Extracting Useful Knowledge from Volumes of Data. *Communications of the ACM* 39(11), 27–34 (1996)
- 13.Forgy, C.: RETE: A Fast Algorithm for the many Pattern/many Object Pattern match Problem. *Artificial Intelligence* 19(1), 17–37 (1982)
14. Gopalakrishna, R., Spafford, E.H.: A Framework for Distributed Intrusion Detection using Interest Driven Cooperating Agents. In: Proceedings of the 4th International Symposium on Recent Advances in Intrusion Detection, Davis, CA, USA (2001)
15. Guan, Y., Ghorbani, A., Belacel, N.: Y-MEANS: A Clustering Method for Intrusion Detection. In: Proceedings of Canadian Conference on Electrical and Computer Engineering, Montréal, Québec, Canada, pp. 1083–1086 (2003)
16. Helmer, G., Wong, J.S.K., Honavar, V.G., Miller, L.: Automated Discovery of Concise Predictive Rules for Intrusion Detection. *Journal of Systems and Software* 60(3), 165–175 (2002)
17. Helmy, T.: Adaptive Ensemble Multi-Agent Based Intrusion Detection Model. In: Ragab, K., Helmy, T., Hassanien, A.E. (eds.) Developing Advanced Web Services through P2P Computing and Autonomous Agents: Trends and Innovations, pp. 36–48. IGI Global (2010)
18. Herrero, Á., Corchado, E.: Multiagent Systems for Network Intrusion Detection: A Review. In: Herrero, Á., Gastaldo, P., Zunino, R., Corchado, E. (eds.) CISIS 09. Advances in Soft Computing, vol. 63, pp. 143–154. Springer, Heidelberg (2009)
19. Huang, W., An, Y., Du, W.: A Multi-Agent-Based Distributed Intrusion Detection System. In: Proceedings of the 3rd International Conference on Advanced Computer Theory and Engineering, Chengdu, Sichuan province, China, pp. 141–143 (2010)
20. Iren, L.-F., Francisco, M.-P., José, M.-G.F., Rogelio, L.-F., Antonio, G.-M.-A.J., Diego, M.-J.: Intrusion Detection Method Using Neural Networks Based on the Reduction of Characteristics. In: Proceedings of the 10th International Work-Conference on Artificial Neural Networks, Salamanca, Spain, pp. 1296–1303 (2009)

21. Isaza, G.A., Castillo, A.G., Duque, N.D.: An Intrusion Detection and Prevention Model Based on Intelligent Multi-Agent Systems, Signatures and Reaction Rules Ontologies. In: Proceedings of the 7th International Conference on Practical Applications of Agents and Multi-Agent Systems, PAAMS 2009, Salamanca, Spain, pp. 237–245 (2009)
22. Kolaczek, G., Juszczyszyn, K.: Attack Pattern Analysis Framework for Multiagent Intrusion Detection System. *International Journal of Computational Intelligence Systems* 1(3) (2008)
23. Lee, W.: A Data Mining Framework for Constructing Features and Models for Intrusion Detection Systems. Phd thesis, Columbia University, New York, NY, USA (1999)
24. Li, T.R., Pan, W.M.: Intrusion Detection System Based on New Association Rule Mining Model. In: Proceedings of the International Conference on Granular Computing, Beijing, China, pp. 512–515 (2005)
25. Lui, C.-L., Fu, T.-C., Cheung, T.-Y.: Agent-Based Network Intrusion Detection System Using Data Mining Approaches. In: Proceedings of the 3rd International Conference on Information Technology and Applications, Sydney, Australia, pp. 131–136 (2005)
26. MacQueen, J.B.: Some Methods for Classification and Analysis of Multivariate Observations. In: Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability, Berkeley, pp. 281–297 (1967)
27. Maxion, R.A., Roberts, R.R.: Proper Use of ROC Curves in Intrusion/Anomaly Detection. Technical report series cs-tr-871, School of Computing Science, University of Newcastle upon Tyne (2004)
28. Mohammed, R.G., Awadelkarim, A.M.: Design and Implementation of a Data Mining-Based Network Intrusion Detection Scheme. *Asian Journal of Information Technology* 10(4), 136–141 (2011)
29. Mosqueira-Rey, E., Alonso-Betanzos, A., Guijarro-Berdiñas, B., Alonso-Ríos, D., Lago-Píñeiro, J.: A Snort-based Agent for a JADE Multi-agent Intrusion Detection System. *International Journal of Intelligent Information and Database Systems* 3(1), 107–121 (2009)
30. Palomo, E.J., Domínguez, E., Luque, R.M., Muñoz, J.: A Self-Organized Multiagent System for Intrusion Detection. In: Proceedings of the 4th International Workshop on Agents and Data Mining Interaction, Budapest, Hungary, pp. 84–94 (2009)
31. Pasquier, N., Bastide, Y., Taouil, R., Lakhal, L.: Efficient Mining of Association Rules Using Closed Itemset Lattices. *Journal of Information Systems* 24(1), 25–46 (1999)
32. Patcha, A., Park, J.M.: An Overview of Anomaly Detection Techniques: Existing Solutions and Latest Technological Trends. *Computer Networks* 51, 3448–3470 (2007)
33. Peddabachigari, S., Abraham, A., Grosan, C., Thomas, J.: Modeling Intrusion Detection System Using Hybrid Intelligent Systems. *Journal of Network Computer Applications* 30, 114–132 (2007)
34. Portnoy, L., Eskin, E., Stolfo, W.S.J.: Intrusion Detection with Unlabeled Data using Clustering. In: Proceedings of ACM CSS Workshop on Data Mining Applied to Security (DMSA-2001), Philadelphia, PA (2001)
35. Rehák, M., Pechoucek, M., Celeda, P., Novotny, J., Minarik, P.: CAMNEP: Agent-Based Network Intrusion Detection System. In: Proceedings of the 7th International Conference on Autonomous Agents and Multiagent Systems, Estoril, Portugal, pp. 133–136 (2008)

36. Roesch, M.: SNORT - Lightweight Intrusion Detection System for Networks. In: Proceedings of the 13th USENIX Conference on System Administration (LISA 1999), Seattle, Washington, pp. 229–238 (1999)
37. Shun, J., Malki, H.A.: Network Intrusion Detection System Using Neural Networks. In: Proceedings of the 4th International Conference on Natural Computation (ICNC 2008), Jinan, China, pp. 242–246 (2008)
38. Shyu, M.-L., Sainani, V.: A Multiagent-based Intrusion Detection System with the Support of Multi-Class Supervised Classification. In: Data Mining and Multi-Agent Integration, pp. 127–142. Springer, Heidelberg (2009)
39. Spafford, E.H., Zamboni, D.: Intrusion Detection Using Autonomous Agents. *The International Journal of Computer and Telecommunications Networking* 34(4), 547–570 (2000)
40. Stolfo, S., Prodromidis, A.L., Tselepis, S., Lee, W., Fan, D.W., Chan, P.K.: JAM: Java Agents for Meta-Learning over Distributed Databases. In: Proceedings of the 3rd International Conference on Knowledge Discovery and Data Mining, Newport Beach, California, pp. 74–81 (1997)
41. Tsai, F.: Network Intrusion Detection Using Association Rules. *International Journal of Recent Trends in Engineering* 2(2), 202–204 (2009)
42. Wooldridge, M.: An Introduction to MultiAgent Systems, 2nd edn. John Wiley and Sons (2009)
43. Xuren, W., Famei, H., Rongsheng, X.: Modeling Intrusion Detection System by Discovering Association Rule in Rough Set Theory Framework. In: Proceedings of the International Conference on Computational Intelligence for Modelling Control and Automation, Sydney, Australia, pp. 24–29 (2006)
44. Zhang, Y., Xiong, Z., Wang, X.: Distributed Intrusion Detection Based on Clustering. In: Yeung, D.S., Liu, Z.-Q., Wang, X.-Z., Yan, H. (eds.) ICMLC 2005. LNCS (LNAI), vol. 3930, pp. 2379–2383. Springer, Heidelberg (2006)
45. Zhao, Z., Guo, S., Xu, Q., Ban, T.: G-MEANS: A Clustering Algorithm for Intrusion Detection. In: Processing of the 15th International Conference on Advances in Neuro-Information, Auckland, New Zealand, pp. 563–570 (2008)

Change Point Analysis for Intelligent Agents in City Traffic

Maksims Fiosins*, Jelena Fiosina**, and Jörg P. Müller

Clausthal University of Technology, Clausthal-Zellerfeld, Germany

{Maksims.Fiosins,Jelena.Fiosina}@gmail.com,

Joerg.Mueller@tu-clausthal.de

Abstract. Change point (CP) detection is an important problem in data mining (DM) applications. We consider this problem solving in multi-agent systems (MAS) domains. Change point testing allows agents to recognize changes in the environment, to detect more accurately current state information and provide more appropriate information for decision-making. Standard statistical procedures for change point detection, based on maximum likelihood estimators, are complex and require construction of parametrical models of data. In methods of computational statistics, such as bootstrapping or resampling, complex statistical inference is replaced by a large computation volumes. However, these methods require accurate analysis of their precision. In this paper, we apply and analyze a bootstrap-based CUSUM test for change point detection, as well as propose a pairwise resampling CP test. We derive some useful properties of the tests and demonstrate their application in the decentralized decision-making of vehicle agents in city traffic.

Keywords: Multiagent decision-making, data mining, change point detection, resampling, variance, bootstrapping CUSUM test, traffic control.

1 Introduction

Change point (CP) analysis is an important problem in data mining (DM), the purpose of which is to determine if and when a change in a data set has occurred. In multiagent systems (MAS) research, methods of CP analysis are applied, but not widely. One of the most popular agent-related areas of CP analysis application is web mining. Here, agents deal with automatic knowledge discovery [5] from web documents and services, including social networks. CP detection in values of different parameters, such as number of messages, frequency of certain actions, or number of active users in some blogs are relevant for a wide number of applications, such as marketing, production, security. For example, Lu et. al.

* Maksims Fiosins is supported by the Lower Saxony Technical University (NTH) project "Planning and Decision Making for Autonomous Actors in Traffic".

** Jelena Fiosina is supported within the FP7 under IEF grant agreement no. FP7-IEF-274881 "Agent-Oriented Distributed Data Mining using Computational Statistics".

[11] applied the CUSUM algorithm in combination with shared beliefs for agent-oriented detection of network attacks, McCulloh [10] for detection of changes in social networks.

A traditional statistical approach to the problem of CP detection is maximum likelihood estimation (MLE) [6],[9]. In this approach, an appropriate data model is constructed, which includes a model of CP. Then a likelihood function for model parameters (including CP) is designed and an estimator of for CP is obtained as a result of the likelihood function minimization. Such an approach requires assumptions about the data model and underlying distributions as well as complex analytical or numerical manipulations with a likelihood function.

As an alternative to the classical approach, methods of computational statistics, like bootstrap or resampling, are widely used [8]. In such methods, complex analytical procedures are replaced by intensive computation, which is effective with modern computers. However, these methods only provide approximate solutions, and the analysis of their accuracy and convergence rate is very important for their correct application.

One of the most popular methods of computational statistics, used for CP analysis, is a cumulative sum bootstrapping test (CUSUM bootstrapping test) [4]. This method does not require assumptions about data models and corresponding distributions; the idea of the test is to construct so-called CUSUM plots: one on the initial sample and one on each of a large number on permuted (bootstrapped) samples. The difference between the initial plot and bootstrapped plots aids to spread regarding the CP existence (see Figure 3).

In previous publications, we applied the resampling method for analysis and DM in different kinds of systems, including information, reliability, transportation logistics, software systems, including MAS [1],[3].

The purpose of this paper is twofold. First, we explain how non-parametrical CP detection methods may be integrated into agent decision support by illustrating it on decentralized traffic routing scenario. Second, we demonstrate how to analyze a precision of the considered tests, taking expectation and variance of resulting estimators as an efficiency criteria.

We discuss two CP tests based on methods of computational statistics: a bootstrap-based CUSUM test, and a novel test, called pairwise resampling test. We analyze the efficiency of both tests as well as show how these tests are applied in MAS systems, focusing on the traffic applications. Case studies are presented to demonstrate how CP analysis is incorporated into agents decision-making processes to verify the potential effect of the proposed approach.

The paper is organized as follows. In Section 2, we describe how CP analysis is incorporated in the decision module of agents. In Section 3, we formulate the CP problem and explain a standard CUSUM bootstrapping approach. In Section 4, we present proposed CP detection algorithms. In Section 5, we provide the most important aspects of the tests efficiency analysis. Section 6 demonstrates a case study, where the proposed methods are used in decentralized traffic scenario. Section 6 contains final remarks, conclusions and outlook.

2 CP Based Decision Making for Agents in Traffic

Appropriate DM tools are very important for a class of MAS where the individual agents use DM technologies to construct and improve a basis for local decision-making and use communication, coordination and cooperation mechanisms in order to improve local decision-making models and to provide a basis for joint decision-making [12]. In order to make appropriate decisions, agents analyze incoming data flows, construct relevant models and estimate their parameters.

The environment and behavior of other agents are subject to changes. Suppose, some other agent decides to change its plans and start acting in a new manner, some part of the environment may become unavailable for agents, new agents may appear etc. So, the old behavior of the agent becomes inappropriate.

Let us consider a simple example from a traffic domain. Let a vehicle agent plans its route through a street network of a city. The agent is equipped with a receiver device, which allows obtaining an information about times needed to travel through the streets. Based on this information, the vehicle agent makes strategic decisions regarding its route. The vehicle does not use only messages from TMC: rather it builds some model based on historical information (such a model is considered by Fiosins et. al. [7]). In the simplest case, the vehicle agent just calculates an average of some set of historical observations.

Now suppose that some change occurs (traffic accident, overloading of some street etc.). The purpose of the agent is to detect this change and make appropriate re-routing decisions.

Let us describe an architecture of an intelligent agent from this point of view. It receives observations (percepts) from the environment as well as communication from other agents (Fig. 1). Communication subsystem is responsible for receiving input data, which then is pre-processed by an initial data processing module. The information, necessary for decision-making, is obtained from initial data by constructing corresponding data models (regression, time series, rule-based etc.). A data models estimation/learning module is responsible for the estimation of the model parameters or iterative estimation ("learning"), which provides an information base for an agent. Mentioned blocks represent a DM module of the agent. The information then is transformed to an internal state of the agent, which is agent's model of the real world. It represents the agent knowledge about the environment/other agents. Based on the internal state, the agent performs its decision-making. The efficiency (utility) functions measure an accordance of the internal state as a model of the external (environment state) with goals of the agent. Based on it, the agent produces (updates) its plan to reach its goal; this process may include the internal state change. As well, the efficiency functions themselves (or their parameters) can change under learning process.

The decision-making process includes strategic decisions of the agents [7], which define plans of general resource distribution as well as tactical decisions, including operative decisions regarding resource usage. For example, strategic decisions of a vehicle agent may include the route choice, but tactical decisions may include include speed/lane selection. As well, the agent plans its social

behavior, i.e. its interactions with other agents. The result of this process is a construction of a plan (policy). The plan is given to an action module for the actual action selection and execution.

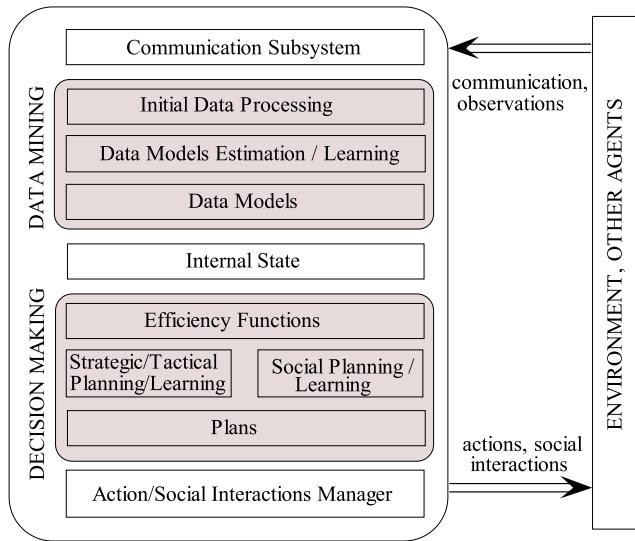


Fig. 1. An agent architecture including DM and decision making modules

Consider an example of DM module of an autonomous vehicle agent (Fig. 2).

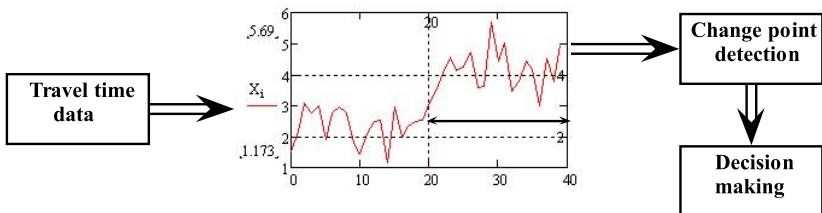


Fig. 2. DM module for strategic (routing) information processing

The vehicle receives travel time data, which are pre-processed by the initial data processing module. Then data is tested on the existence of CP. If it is detected, only data after the last CP are used in future analysis, where the current state (travel time) through a given street is estimated. In the simplest case an average of travel times after the last CP is used.

In the next Section we present a CP detection problem as well as describe a standard bootstrap-based CUSUM test for CP detection.

3 CP Problem and CUSUM Test

Let us formulate a CP detection problem. Let $\mathbf{X} = \{x_1, x_2, \dots, x_n\}$ be a random sample. Let us divide this sample into two parts as $\mathbf{X} = \{\mathbf{X}^B, \mathbf{X}^A\}$, where $\mathbf{X}^B = \{x_1^B, x_2^B, \dots, x_k^B\}$, $\mathbf{X}^A = \{x_1^A, x_2^A, \dots, x_{n-k}^A\}$. We say that there is a CP at position k in this sample, if elements \mathbf{X}^B are distributed according to a distribution function (cdf) $F_B(x)$, but elements \mathbf{X}^A according to cdf $F_A(x) \neq F_B(x)$. The aim of a CP detection test is to estimate the value of k (clear that in the case of $k = n$ there is no CP). We are interested in a case when $F_B(x)$ and $F_A(x)$ differ by a mean value.

Note that a CP is not always visually detectable. Figure 3 represents two samples: left has exponential distribution, right has normal one. Both have a CP at $k = 10$: for the exponential distribution its parameter λ changes from $1/10$ to $1/20$; for the normal distribution its parameter μ changes from 5 to 7 . One should have an experience to see these CP visually (see Fig. 3).

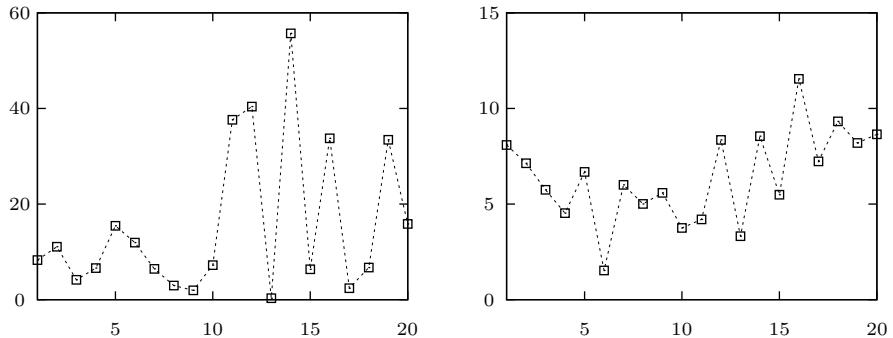


Fig. 3. A sample of 20 observations from exponential (left) and normal (right) distributions with CP at $k = 10$

A popular non-parametric approach for CP analysis is bootstrapping CUSUM test. CUSUM presents the cumulative sum of the differences between individual data values and the mean. If there is no shift in the mean of the data, the CUSUM chart will be relatively flat with no pronounced changes in slope. Also, the range (the difference between the highest and lowest data points) will be small. A data set with a shift in the mean will have a slope change at the data point where the change occurred, and the range will be relatively large.

The cumulative sum S_i at each point i is calculated by the sample \mathbf{X} by adding the difference between a current value and sample mean to the previous sum as

$$S_i = \sum_{j=1}^i (x_j - \bar{X}), \quad (1)$$

where \bar{X} is the mean of the sample \mathbf{X} , $i = 1, 2, \dots, n$.

A CUSUM chart starting at zero will always end with zero as well: $S_n = 0$. If a CUSUM chart slopes down, it indicates that most of the data are below the mean. A change in the direction of a CUSUM indicates a shift in the average. At the CP, the slope changes direction and increases, indicating that most of the data points are now greater than the average.

In order to make a CP detection procedure more formal, a measure of the initial CUSUM line divergence from "normal" lines for given data is calculated. It is calculated using a technique known as bootstrapping, whereby N random permutations \mathbf{X}^{*j} , $j = 1, 2, \dots, N$ of \mathbf{X} are generated and corresponding CUSUMS S_k^{*j} are calculated by formula (1).

For a fixed point k the percentage of times where the cumulative sum for the original data exceeds the cumulative sum for the randomized bootstrap data is calculated as $p_k^* = \#\{j : S_k^{*j} \leq S_k\}/N$.

For values of p_k^* near 0 or near 1 we can say that the CP in k occurs. The idea behind this is that values S_k^{*j} approximate the distribution of CUSUMS constructed till k under assumption that data is mixed (values may be taken both before the CP and after the CP).

An example of a CUSUM test is presented on Figure 4.

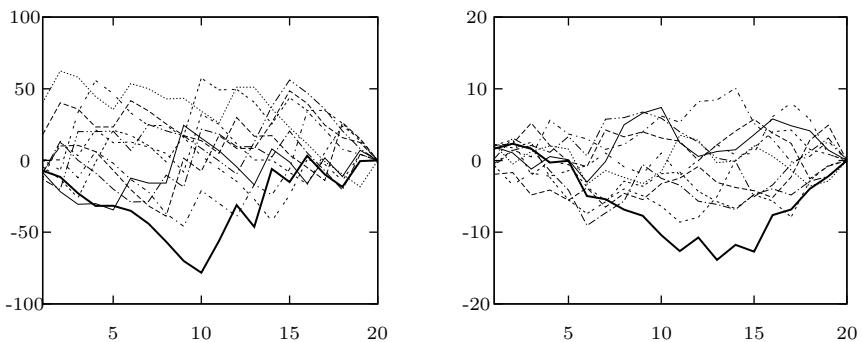


Fig. 4. CUSUM test examples with CP at $k = 10$

Here we can see a minimum of original CUSUM line at point $k = 10$; for other bootstrapped lines there is no minimum.

However, is this test reliable? Will the original CUSUM line be always outside of bootstrapped CUSUM line range? If not, then what is an expected value of the percentage of bootstrapped CUSUMS, which the original CUSUM exceeds?

How this percentage differs from its expected value? All these questions should be answered during the method analysis phase; without the accurate analysis the method cannot be correctly applied.

4 Resampling Based CP Tests

4.1 CUSUM Based Test

Let us describe an algorithm for a CUSUM-based test for a CP at the point k . On r -th resampling step, we extract, without replacement, k elements from a sample \mathbf{X} of size n , forming the resample $\mathbf{X}_k^{*r} = \{x_1^{*r}, x_2^{*r}, \dots, x_k^{*r}\}$. Then we construct r -th resampling CUSUM

$$S^{*r} = \sum_{i=1}^k (x_i^{*r} - \bar{X}) = \sum_{i=1}^k x_i^{*r} - k\bar{X}, \quad (2)$$

where \bar{X} is an average over the initial sample \mathbf{X} .

Each CUSUM is compared with a pre-defined value x , obtaining an indicator value $\zeta^{*r} = 1_{S^{*r} \leq x}$.

We make N such realizations, obtaining a sequence of indicators $\zeta^{*1}, \zeta^{*2}, \dots, \zeta^{*N}$. These values in fact approximate a cdf of CUSUMS. We estimate it as

$$F^*(x) = P\{S^{*r} \leq x\} = \frac{1}{N} \sum_{r=1}^N \zeta^{*r}. \quad (3)$$

As a value of x we take a value of a CUSUM S , calculated by initial data. So the probability of interest is $F^*(S)$.

Low or high value of this probability allows to spread about CP existence. In principle, we can find maximal (close to 1) or minimal (close to 0) value of this probability on all k and consider this point as a CP.

A corresponding calculation of the probability $F^*(S)$ is presented in Algorithm 1.

Algorithm 1. Function CUSUM_TEST

```

1: function CUSUM_TEST( $\mathbf{X}, k, N$ )
2:    $S = \sum_{i=1}^k (x_i - \bar{X})$ 
3:   for  $r = 1 \dots N$  do
4:      $\mathbf{X}_k^{*r} \leftarrow \text{resample}(\mathbf{X}, k)$ 
5:      $S^{*r} = \sum_{i=1}^k (x_i^{*r} - \bar{X})$ 
6:     if  $S^{*r} < S$  then  $\zeta^{*r} = 1$ 
7:     else  $\zeta^{*r} = 0$ 
8:   end for
9:   return  $1/N \sum_{r=1}^N \zeta^{*r}$ 
10: end function

```

However, as the same elements can be used for calculation of ζ^{*r} on different realizations r , it leads to a complex structure of dependency between ζ^{*r} . So we should be accurate in results interpretation here. In Section 5 we provide the most important aspects of this test analysis.

4.2 Pairwise Resampling CP Test

We propose an alternative resampling-based CP test; we call it pairwise resampling test. It is based on calculation of the probability $P\{Y \leq Z\}$ that one random variable (r.v.) Y is less than another r.v. Z [2]. Suppose that the sample \mathbf{X}^B contains realizations of some r.v. Y , but the sample \mathbf{X}^A realizations of some r.v. Z . Our characteristic of interest is the probability $\Theta = P\{Y \leq Z\}$.

On r -th resampling step we extract one value y^{*r} and z^{*r} from the samples \mathbf{X}^B and \mathbf{X}^A correspondingly and calculate an indicator value $\zeta^{*r} = 1_{y^{*r} \leq z^{*r}}$.

We make N such realizations, obtaining a sequence of indicators $\zeta^{*1}, \zeta^{*2}, \dots, \zeta^{*N}$. The resampling estimator of Θ is

$$\Theta^* = \frac{1}{N} \sum_{r=1}^N \zeta^{*r}. \quad (4)$$

In order to check if there is a CP, we construct a confidence interval for Θ . We produce v such estimators, denote them $\Theta_1^*, \Theta_2^*, \dots, \Theta_v^*$. Let us order them, producing an ordered sequence $\Theta_{(1)}^* \leq \Theta_{(2)}^* \leq \dots \leq \Theta_{(v)}^*$.

Let us select a confidence probability γ for this interval (γ is usually selected 0.95 or 0.99). We accept $[\Theta_{(\lfloor \frac{1-\gamma}{2} v \rfloor)}^*; \Theta_{(\lfloor \frac{\gamma}{2} v \rfloor)}^*]$ as a γ confidence interval for Θ .

Note that in the case of CP absence the probability Θ will be equal to 0.5, and the estimators Θ^* will be close, but different from 0.5. However, is this difference significant? In order to answer we check if value 0.5 traps into the constructed confidence interval (Algorithm 2).

Algorithm 2. Function PAIRWISE_CONFIDENCE

```

1: function PAIRWISE_CONFIDENCE( $\mathbf{X}, k, N, v, \gamma$ )
2:    $\mathbf{X}^B = \text{subsample}(\mathbf{X}, 1, k)$ ,  $\mathbf{X}^A = \text{subsample}(\mathbf{X}, k + 1, n)$ 
3:   for  $j = 1 \dots v$  do
4:     for  $r = 1 \dots N$  do
5:        $y^{*r} \leftarrow \text{resample}(\mathbf{X}^B, 1)$ ,  $z^{*r} \leftarrow \text{resample}(\mathbf{X}^A, 1)$ 
6:       if  $y^{*r} < z^{*r}$  then  $\zeta^{*r} = 1$ 
7:       else  $\zeta^{*r} = 0$ 
8:     end for
9:      $\Theta_j^* = \sum_{r=1}^N \zeta^{*r}$ 
10:   end for
11:   sort $\Theta^*$ 
12:   return  $[\Theta_{(\lfloor \frac{1-\gamma}{2} v \rfloor)}^*; \Theta_{(\lfloor \frac{\gamma}{2} v \rfloor)}^*]$ 
13: end function

```

There is again a complex dependence structure between ζ^{*r} , and so between Θ^* , because the same elements may be used in comparisons on different realizations. So true coverage probability of constructed interval will differ from γ . The goal of the algorithm analysis is to calculate the true coverage probability of this interval; then we can correctly apply the method.

5 Analysis of the CP Tests Accuracy

In this Section, we shortly highlight the most important aspects of the methods efficiency analysis. The complete analysis can be found in our articles [1],[2],[3].

5.1 CUSUM Based Test

In order to analyze the accuracy of CUSUM based test, we are going to calculate an expectation and variance of the estimator (3). This means that we calculate theoretically an average of the estimator and spread of the percentage of cases, when the CUSUM constructed on the original data exceeds CUSUMS constructed on the bootstrapped data.

Remind that the Algorithm 1 for each resampling realization r forms a resample \mathbf{X}_k^{*r} , which consists of k elements. Between these k elements there are some elements from \mathbf{X}^B and some from \mathbf{X}^A . We fix this number: let y_r be a number of elements, extracted from \mathbf{X}^B ; then from \mathbf{X}^A we extract $k - y_r$ elements; let $q(y_r)$ be a probability of this event, which can be calculated using hypergeometrical distribution:

$$q(y_r) = \frac{\binom{k}{y_r} \binom{n-k}{k-y_r}}{\binom{n}{k}}. \quad (5)$$

Then the expectation of (3) can be expressed using common formula for the sum of random variables for the fixed y_r , taking then all possible values of y_r with their probabilities (5):

$$E[F^*(x)] = P\{S^{*r} \leq x\} = \sum_{y_r=0}^k \int_{-\infty}^{\infty} F_B^{(y_r)}(x-u) dF_A^{(k-y_r)}(u) \cdot q(y_r), \quad (6)$$

where $F^{(k)}(x)$ is a convolution of the cdf $F(x)$ with itself.

Calculation of variance of (3) is a bit more complex. Using standard formula for variance of a sum, we have:

$$Var[F^*(x)] = \frac{1}{N} Var[1_{\{S^{*r} \leq x\}}] + \frac{(N-1)}{N} Cov[1_{\{S^{*r} \leq x\}}, 1_{\{S^{*p} \leq x\}}], \quad (7)$$

for $r \neq p$.

In formula (7) only the covariance term depends on the resampling procedure; the variance depends on the structure of the distributions $F_A(x)$ and $F_B(x)$ and can be calculated by analog with (6). The covariance term may be expressed by the standard formula for covariance

$$\begin{aligned} \text{Cov} & \left[1_{\{S_k^*(r) \leq x\}}, 1_{\{S_k^*(p) \leq x\}} \right] = \\ & = E \left[1_{\{S_k^*(r) \leq x\}} \cdot 1_{\{S_k^*(p) \leq x\}} \right] - E \left[1_{\{S_k^*(r) \leq x\}} \right] E \left[1_{\{S_k^*(p) \leq x\}} \right] = \\ & = \mu_{11} - (E[F^*(x)])^2, \end{aligned} \quad (8)$$

where $\mu_{11} = [1_{\{S_k^*(r) \leq x\}} \cdot 1_{\{S_k^*(p) \leq x\}}]$ is the second mixed moment of $1_{\{S_k^*(r) \leq x\}}$.

In order to calculate μ_{11} , we use the notation of α -pair [1],[3]. Let $\alpha = (\alpha_B, \alpha_A)$, where α_B and α_A are the number of common elements extracted from \mathbf{X}^B and \mathbf{X}^A correspondingly on two different resampling realizations.

Let us demonstrate a small example. Let the sample \mathbf{X}^B contains 5 elements, name them $b1, b2, \dots, b5$, and the sample \mathbf{X}^A contains 7 elements, name them $a1, a2, \dots, a7$. Let $k = 6$, which means that we extract 6 elements from these samples. Let for some resampling realization r we formed a resample $\mathbf{X}_6^{*r} = \{b4, a1, a6, b2, a3, a7\}$, but on another resampling realization p we formed a resample $\mathbf{X}_6^{*p} = \{a4, b2, a3, a7, b1, a1\}$. These two resampling realizations form an $\alpha = (1, 3)$ -pair.

Than μ_{11} can be expressed by fixing all possible values of α :

$$\mu_{11} = \sum_{\alpha} \mu_{11}(\alpha) P(\alpha), \quad (9)$$

where $\mu_{11}(\alpha)$ is a value of the mixed moment μ_{11} for the fixed α -pair, $P(\alpha)$ is a probability of the α -pair.

The calculation of $\mu_{11}(\alpha)$ may be done by fixing the values y_r and y_p of a number of common elements, extracted from \mathbf{X}^B on r -th and p -th resampling realizations correspondingly. The probability $P(\alpha)$ of the α -pair can be calculated using hypergeometrical distribution:

$$P(\alpha) = \frac{\binom{y_r}{\alpha_B} \binom{k - y_r}{y_p - \alpha_B}}{\binom{k}{y_p}} \cdot \frac{\binom{k - y_r}{\alpha_A} \binom{n - 2k + y_r}{k - y_p - \alpha_A}}{\binom{n - k}{k - y_p}}. \quad (10)$$

For the case of exponential and normal distributions we can obtain explicit formulas for the previous expressions.

Mentioned approach allows obtaining explicit expressions for the expectation and variance of the estimator (3), which allows to analyze, how effective is this test in different situations. We demonstrate numerical examples in Section 6.

5.2 Pairwise Resampling CP Test

In order to analyze properties of the pairwise test confidence interval, which is constructed using (4) and Algorithm 2, we introduce a protocol notation [2]. The

protocol fixes the relative position of the elements in the initial samples, which allows analyzing the test properties by fixing all possible protocols.

Let us order a sample \mathbf{X}^B , obtaining an ordered sequence $\{x_{(1)}^B, x_{(2)}^B, \dots, x_{(k)}^B\}$. Let $c_i = \#\{x_j^A \in \mathbf{X}^A : x_{(i-1)}^B \leq x_j^A \leq x_{(i)}^B\}$, $x_{(0)}^B = -\infty$, $x_{(k+1)}^B = \infty$. So c_i denotes a number of the sample \mathbf{X}^A elements, which trap between elements $i-1$ and i of the ordered sample \mathbf{X}^B . We call $k+1$ -dimensional vector $C = \{c_1, c_2, \dots, c_{k+1}\}$ as a protocol.

For example, let $\mathbf{X}^B = \{2.1, 9.2, 5.2, 5.0, 2.4\}$, $\mathbf{X}^A = \{7.3, 8.5, 6.3, 2.0, 10.4, 3.2, 1.2\}$. Let us order a sample $\mathbf{X}^B = \{2.1, 2.4, 5.0, 5.2, 9.2\}$ and calculate how much elements from \mathbf{X}^A trap to each interval. So we get a protocol $C = \{2, 0, 1, 0, 3, 1\}$.

The probability calculation of the protocol is complicated and may be done iteratively [2]. However, for an exponential distribution it is possible to get an explicit formula using its properties. Let Y_i have exponential distribution with parameter λ and Z_j have exponential distribution with parameter the probability of the C -th protocol is

$$P_C = \prod_{i=1}^n \frac{\lambda(n-i+1)}{\lambda(n-i+1) + \mu(m - \sum_{\nu=0}^{i-1} c_{\nu})} \cdot \prod_{j=0}^{c_{i-1}-1} \frac{\mu(m-j - \sum_{\nu=0}^{i-2} c_{\nu})}{\lambda(n-i+1) + \mu(m-j - \sum_{\nu=0}^{i-2} c_{\nu})} \quad (11)$$

For a fixed protocol C the conditional probability of the event $\{Y \leq Z\}$ can be calculated as

$$q_C = P\{Y \leq Z|C\} = \frac{1}{k(n-k)} \sum_{i=1}^k \sum_{j=i}^k c_j. \quad (12)$$

The probability that one resampling estimator Θ_j^* will be less than Θ is given by the binomial distribution with a probability of success (12):

$$\rho_C = P\{\Theta_j^* \leq \Theta|C\} = \sum_{\zeta=0}^{\Theta r-1} \binom{r}{\zeta} q_C^\zeta (1-q_C)^{r-\zeta}. \quad (13)$$

Finally the unconditional probability of coverage is calculated as $\sum_C P_C \cdot R_C$. Note that this probability is usually less than γ .

6 Case Study

Vehicle Agent. We consider a vehicle routing problem in a street network, where vehicles receive data about travel times and are applying the shortest path algorithm looking for a fastest path to their destination. As travel times are subject to change, that's why CP analysis is performed. If CP is detected, only the data part after the last CP is taken into account.

We suppose, that travel times through the streets are normally distributed and are subject to changes in the mean. An example of input data is shown in Figure 5 (left). The vehicle analyses CPs in such data for all streets and selects an appropriate fastest route; the route selection process is presented in Figure 5 (right).

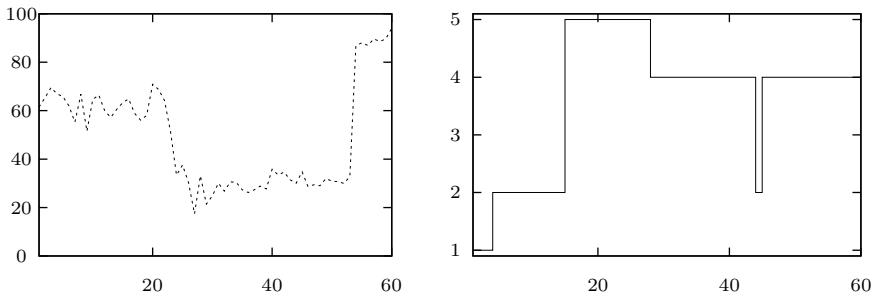


Fig. 5. Travel times on one street with 60 observations (left) and resulting route selection from 5 routes (right)

Now consider the behavior of CP estimators. In Figure 6 (left) the CUSUM test presented in the case of CP absence. In this case, we can see a big variance of the probability $F^*(x)$ of interest (standard deviation = 0.29). This means, that there exist a big risk of considering some point as a CP, if it is not one. Figure 6 (right) demonstrates the CUSUM test in the case of CP existence. Here we see very good CP detection with practically zero variance at the CP.

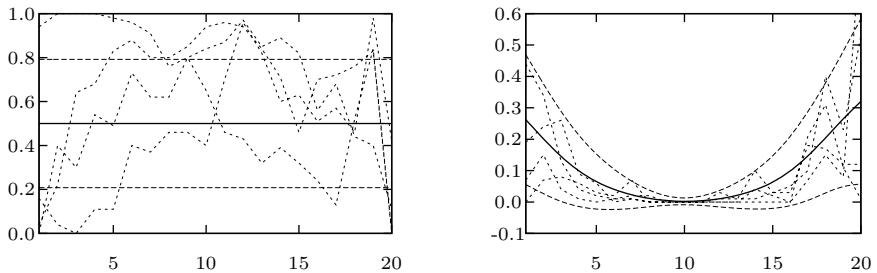


Fig. 6. Results of the CUSUM test without CP (left) and with CP at $k = 10$ (right) for a fragment of 20 observations of a vehicle agent. Straight line shows the expected value $E[F^*(x)]$ of the estimator $F^*(x)$, dashed lines show the difference between the expected value and standard deviation $E[F^*(x)] - Var[F^*(x)]^{1/2}$ of the estimator $F^*(x)$, dotted lines show several realizations of $F^*(x)$

Now let us consider the pairwise test. In Figure 7 (left) we see this test in the case of CP absence. Here we see smaller variance of the probability Θ^* of interest (standard deviation = 0.14 on the most of the interval). This means, that a risk of considering some point as a CP, if it is not one, is lower than for the CUSUM test. Figure 7 (right) demonstrates this test in the case of CP. Here detection is not so bad as well, however the variance of the estimator is bigger, so there is a risk to miss this CP.

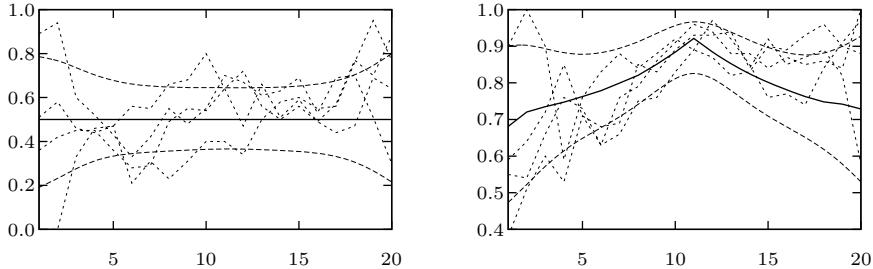


Fig. 7. Results of the pairwise test without CP (left) and with CP at $k = 10$ (right) for a fragment of 20 observations of a vehicle agent. Straight line shows the expected value $E[\Theta^*]$ of the estimator Θ^* , dashed lines show the difference between the expected value and standard deviation $E[\Theta^*] - \text{Var}[\Theta^*]^{1/2}$ of the estimator Θ^* , dotted lines show several realizations of Θ^* .

Traffic Light Agent. We consider a traffic control device agent (TCD) (traffic lights or matrix signs), which regulate the vehicle flows in the street network. TCD receives data about the intervals between vehicles intensity arriving to this TCD. The working regime of TCD depends on the state of traffic. When TCD noticed some CP in traffic flow characteristics it changes its working regime.

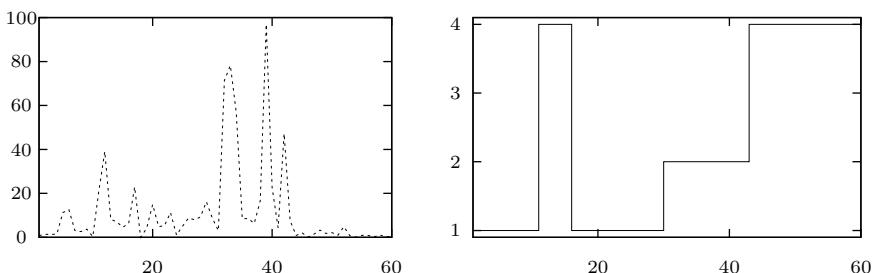


Fig. 8. Interval between vehicle arrivals with 60 observations (left) and resulting traffic light program selection from 5 programs (right)

That's why data mining using CP analysis is of great importance. Suppose for our investigation the initial parameters of intensity of traffic flow are $\lambda_B = 0.1$ before the CP and $\lambda_A = 0.05$ after the CP. Following the previous example we change the initial parameters to verify the correct behavior of our tests.

We show the behavior of our tests' estimators depending on different parameters of initial data (Fig. 8 - 10). For this case, we can see a bit worse performance of CP tests. In Fig. 9 and 10 we see the similar behavior of CUSUM and pairwise test: the first better detects existing CP, the second is more reliable in the case of CP absence.

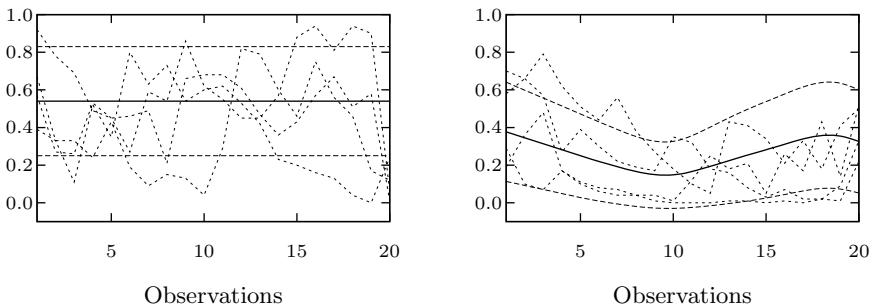


Fig. 9. Results of the CUSUM test without CP (left) and with CP at $k = 10$ (right) for a fragment of 20 observations of a traffic light agent. Straight line shows the expected value $E[F^*(x)]$ of the estimator $F^*(x)$, dashed lines show the difference between the expected value and standard deviation $E[F^*(x)] - Var[F^*(x)]^{1/2}$ of the estimator $F^*(x)$, dotted lines show several realizations of $F^*(x)$

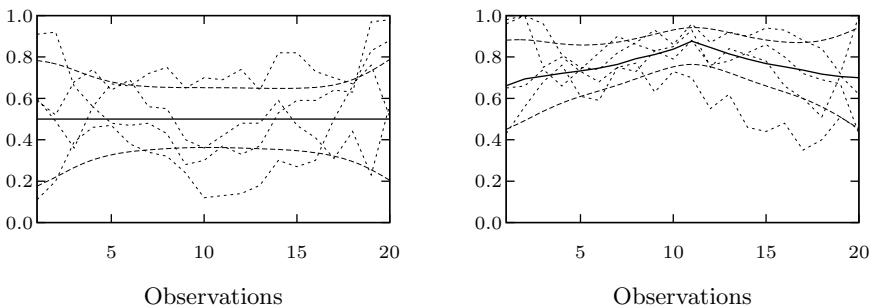


Fig. 10. Results of the pairwise test without CP (left) and with CP at $k = 10$ (right) for a fragment of 20 observations of a traffic light agent. Straight line shows the expected value $E[\Theta^*]$ of Θ^* , dashed show the difference between the exp. value and std. deviation $E[\Theta^*] - Var[\Theta^*]^{1/2}$ of the estimator Θ^* , dotted show several realizations of Θ^* .

We can conclude that the CUSUM test detects CP very well; however, it may consider as CP some point, which is not one. In opposite, the pairwise test is more reliable in the case of a CP absence; however, it can miss some CPs.

So for streets where CPs are rare, it is better to use the pairwise test; the CUSUM test is better for streets with often occurred CPs in travel times.

7 Conclusions

CP detection is very important task of DM for MAS, because it allows agents to estimate more accurate the environment state and prepare more relevant information for decentralized planning and decision-making. As classical statistical methods for CP estimation are relatively complex, it is better to apply methods of computational statistics for this problem.

In this paper, we considered an application of CP detection as a part of DM module of an intelligent agent. We considered two resampling-based CP detection tests: CUSUM-based bootstrapping test and pairwise resampling test. We described algorithms of their application as well as highlighted the most important aspects of their efficiency analysis, taking expectation and variance of the estimators as the efficiency criteria.

We demonstrated an application of CP detection for vehicle agents in city traffic. This allows vehicles to detect CPs in street travel times and select more appropriate path in a street network.

The first test demonstrated good detection of CPs, however has a big variance in the case of CPs absence. The second test has smaller variance in this case, however worse detects existing CPs.

First experiments show that the demonstrated approach allows reducing the travel time of vehicles. In the future we will work on an application of computational statistics methods for different DM procedures in MAS. Another important direction is an application of our approach for different domains.

References

1. Afanasyeva, H.: Resampling-approach to a task of comparison of two renewal processes. In: Proc. of the 12th International Conference on Analytical and Stochastic Modelling Techniques and Applications, Riga, pp. 94–100 (2005)
2. Andronov, A.: On resampling approach to a construction of approximate confidence intervals for system reliability. In: Proceedings of 3rd International Conference on Mathematical Methods in Reliability, Trondheim, Norway, pp. 34–42 (2002)
3. Andronov, A., Fioshina, H., Fioshin, M.: Statistical estimation for a failure model with damage accumulation in a case of small samples. Journal of Statistical Planning and Inference 139(5), 1685–1692 (2009)
4. Antoch, J., Hušková, M., Veraverbeke, N.: Change-point problem and bootstrap. Journal of Nonparametric Statistics 5, 123–144 (1995)
5. Cao, L., Gorodetsky, V., Mitkas, P.: Agent mining: The synergy of agents and data mining. IEEE Intelligent Systems 24(3), 64–72 (2009)

6. Ferger, D.: Analysis of change-point estimators under the null-hypothesis. *Bernoulli* 7(3), 487–506 (2001)
7. Fiosins, M., Fiosina, J., Müller, J., Görmer, J.: Agent-based integrated decision making for autonomous vehicles in urban traffic. In: Proceedings of 9th International Conference on Practical Applications of Agents and MAS (2011)
8. Gentle, J.E.: Elements of Computational Statistics. Springer, Heidelberg (2002)
9. Hinkley, D.V.: Inference about the change-point from cumulative sum tests. *Biometrika* 58(3), 509–523 (1971)
10. McCulloh, I., Lospinoso, J., Carley, K.: Social network probability mechanics. In: Proceedings of the World Scientific Engineering Academy and Society 12th International Conference on Applied Mathematics, Cairo, Egypt, pp. 319–325 (2007)
11. Peng, T., Leckie, C., Ramamohanarao, K.: Detecting reflector attacks by sharing beliefs. In: Proceedings of the IEEE GLOBECOM, pp. 1358–1362 (2003)
12. Symeonidis, A., Mitkas, P.: Agent Intelligence Through Data Mining (Multiagent Systems, Artificial Societies, and Simulated Organizations). Springer, Heidelberg (2005)

Mining Frequent Agent Action Patterns for Effective Multi-agent-Based Web Service Composition

Xiaofeng Wang¹, Wenjia Niu², Gang Li^{3,*}, Xinghua Yang², and Zhongzhi Shi¹

¹ Institute of Computing Technology, Chinese Academy of Sciences,
Beijing 100190, P.R. China

wangxiaofeng@ict.ac.cn, shizz@ics.ict.ac.cn

² Institute of Acoustics, Chinese Academy of Sciences,
Beijing 100190, P.R. China
niuwj.yangxh@ict.ac.cn

³ School of Information Technology
Deakin University, Vic 3125, Australia
gang.li@deakin.edu.au

Abstract. The dynamic description logic (*DDL*) is utilized as one emerging *AI planning*-related solution for automatic Web service composition. However, reasoning utilization when facing the real world service applications in such *DDL*-related solutions is still an open problem. In this paper, we propose the cooperative reasoning-based multi-agent model (*CREMA*) which can systematically incorporate *DDL* action reasoning with data mining, together with a *support*-based planning method for task decomposition in order to improve the overall throughput of the Web service execution. The case study and experimental analysis demonstrates the capability of the proposed approach.

Keywords: Multi-Agent, Action Mining, *DDL* Reasoning, Planning, Service Composition.

1 Introduction

In the past decade, substantial research efforts have been devoted to *automated Web services composition* [17, 23]. When the user-required service is associated with explicit goal definition, and each atomic service can be specified by its *pre-conditions* and *effects*, *AI planning* approach can provides an automatic synthesis of the required Web services through logical reasoning (e.g. *OWL* [19], *OWL-S* [17], *SHOP2* [23]). Recently, the *Dynamic Description Logic (DDL)* [10] [20] has been successfully adopted to facilitate the flexible dynamic *AI planning* for *Web service composition*, mainly because of its well support in the representation and the reasoning of static as well as dynamic Web services information.

* Corresponding author.

Unfortunately, practitioners who want to use the *DDL*-based *AI* planning approach in industrial service computing areas such as *travel planning*, *book publishing* or *distance education*, usually face two significant barriers:

Dynamically Changing Service Resources. In real world applications, the service resources (such as the service *input* and *output* data) may change dynamically. For example, one service may not replace the *flight* querying with a new kind of *train* querying, and also change the corresponding querying interface. In the *DDL*-based *AI* planning approach, a *DDL* knowledge base is used to store, track and manage available service resources, inspired by the idea of *service registry* named *UDDI* [11]. However, when the number of service resources is getting bigger, the updating and maintenance of the *DDL* knowledge base will be time-consuming, and this calls for more efficient method in this *AI* planning approach.

Reliable Task Decomposition. new services may be released in real world applications, while existing services may become unavailable. If a decomposed task can only be executed on a small number of services, the reliability will be suffered when some of these services become busy or unavailable, and the overall throughput of the system will also be affected. Existing *DDL* planning doesn't take the "*support*" of decomposed actions into consideration, even though that some actions could be available on many services, while other actions may only be available on a small number of services.

Recent advances in the *agent technology*, especially the *multi-agent technology*, have provided methods of building agents capable of making autonomous decisions and also enabling the cooperation between agents. Moreover, with the support from *data mining* technique, it is possible to alleviate the above mentioned barriers for the following reasons.

1. Agents can proactively sense the dynamic changes in service resources. When a new service request arrives, agents can provide the latest service resources through their inner autonomous computing. In this sense, agents can make the dynamic resource management process transparent.
2. Agents can actively cooperate with each other on given tasks. With the *multi-agent technology*, agents can collectively share the whole computing in the *DDL* knowledge base, with each agent realizes the autonomous dynamic management on a small scale of service resources. This provides the potentials for effective resource updating and maintenance. Moreover, the central *DDL* planning architecture can also be designed with a relative flexible and redundant mechanism, so that the reliability and executability can be improved.
3. Lots of services stored as *DDL* actions in the *DDL* knowledge base may possess similar behavior. By *data mining* techniques, these common behavior pattern can be identified as a concise summary of all those involved similar actions, and it implies the potentials for improving the task decomposition. In addition, the "*support*" of each discovered behavior pattern could also be utilized for reliable service by increasing the overall throughput of the service composition system.

1.1 Related Work

Since 2002, the agent technology has been incorporated into many *Web service* composition systems. Existing methods either extend the agent model to accommodate service management operations (e.g. *query*, *invoke*) or use agent as a protocol translation middle-ware for services: along the first line of research, the *SEAGENT* system [13] integrates the agent and *OWL-QL* [14], and extends the Semantic Web query language to support agent and related services. In the *NUIN* framework [12], the agent technology was extended by *client action* so that agents can wrap and invoke services. In the second line of research, systems like the *WSIG* [4] develops the *Gateway Agent* and combines the agent platform with the Web service to facilitate better interaction. The *WSIG* develops services provided by agents and published in the *JADE* [2] as Web services with minimal additional effort.

On the other hand, *data mining* can be also utilized by agents to facilitate agent-based services [16] [6] [7] [5]. Krishnaswamy et al. [16] presents a hybrid architectural model for *Distributed Data Mining* (DDM), which is customized for e-businesses applications such as service providers sell *DDM* services to *e-commerce* users and systems. Lately, Cao et al. [6] [7] [5] contributes a series of great efforts on agent-mining based services. In their work, the interaction of agent technology and data mining presents prominent benefits to solve challenging issues in different areas. For instance, data mining can enhance agent learning, while agent can benefit data mining with distributed pattern discovery. They also summarize the main functionalities and features of an agent service and data mining symbiont – *F-Trade*. Furthermore, data mining can be used to strengthen agents for flexible and efficient services, optimization and discovery, as well as plug and play of algorithms. All these agent-mining interaction are mainly driven by mutual needs from agent and mining areas. Therefore, the interaction between agent and data mining provides the potential to complement each other for effective intelligent systems.

1.2 Research Issues

In this paper, we aim to fill the void by proposing a multi-agent model together with the associated pattern discovery and utilization algorithms to strengthen the flexibility and effectiveness of Web service composition. In particular, one key issue is to be addressed in this paper, i.e. *cooperative reasoning-based multi-agent modeling*. As mentioned above, the *multi-agent technology* provides the potentials for effective resource updating and maintenance. How to effectively integrate the *action reasoning* into agents, how to effectively control and manage these agents and how to realize the *support-based* task decomposition mechanism for service composition through multi-agent cooperation, will be supported by the multi-agent model.

The rest of this article is organized as follows. Section 2 briefly introduces the basic concepts and definitions of *DDL*-related work for Web Services. Section 3 presents the cooperative reasoning-based multi-agent model, Section 4 proposes

the frequent agent action patterns mining approach for task decomposition, and section 5 provides case studies and experiments together with the analysis of the proposed approach. Finally conclusions and future work are presented in section 6.

2 Preliminaries

In this section, we will give a brief introduction to the *Dynamic Description Logic (DDL)*.

Dynamic Description Logic (DDL) is a kind of dynamic logics which integrate the *action theory* with description logic ([21], [9]). By incorporating *actions* into conventional description logics, *DDL* supports both the reasoning on actions and the representation of dynamic domain knowledge. The *action* in *DDL* can be defined as follows.

Definition 1 (Atomic Action). *An atomic action in DDL is a tuple $\alpha \equiv <\mathcal{P}, \mathcal{E}>$, where*

1. $\alpha \in \mathcal{N}_A$ represents the name of the atomic action;
2. \mathcal{P} is a set of clauses which conjunctively specify the precondition that should be satisfied to execute the action;
3. \mathcal{E} is a set of clauses that conjunctively specify the effect of when the action has been executed.

Intuitively, an atomic action encodes the changes of the domain caused by the action. Besides the atomic action, *DDL* also allows compound actions to be constructed from atomic actions using a set of rules.

Definition 2 (Compound Action). Compound Actions in DDL can be built up with the following rule:

$$\pi, \pi' \rightarrow \alpha|\phi?|\pi \cup \pi'|\pi; \pi'|\pi^*$$

where α is an atomic action and ϕ is a logic formula in DDL. The action in the form of “ $\phi?$ ” is called test action which tests whether the formula ϕ is hold or not. The action in the form of “ $\pi \cup \pi'$ ” is called select action which represents a selection between π and π' . Action in the form of “ $\pi; \pi'$ ” is called sequence action which represents an action that sequentially executes action π and action π' . The action in the form of “ π^* ” is referred to as iteration action which represents an action that iteratively executes action π .

The compound action can be used to represent program control flows. For example, the program fragment: “if ϕ then π else π' ” can be translated into a compound action: $(\phi?; \pi) \cup ((\neg\phi)?; \pi')$.

In order to reuse the action definition, variables can also be used into action definition. Then, an action α can be represented in *DDL* as a tuple $\alpha(x_1, x_2, \dots, x_k) \equiv (\mathcal{P}_\alpha, \mathcal{E}_\alpha)$, where x_1, x_2, \dots, x_k are action variables that can be assigned by the individuals in the *domain of discourse*, which refers to sets of individuals which can be represented with logic formulas. For simplicity, we will use α to represent $\alpha(x_1, x_2, \dots, x_k) \equiv (\mathcal{P}_\alpha, \mathcal{E}_\alpha)$ in this paper.

3 Cooperative Reasoning-Based Multi-agent Model

From the previous section, it is obvious that multi-agent techniques together with data mining provide a feasible extension for improving *DDL-based AI planning* approach in the Web service composition. In this section, we will propose a *Cooperative REasoning-based Multi-Agent (CREMA)* model for intelligent Web services composition to systematically incorporate the *DDL reasoning* and *data mining* into agent.

3.1 Roles of Agents in Web Service Composition

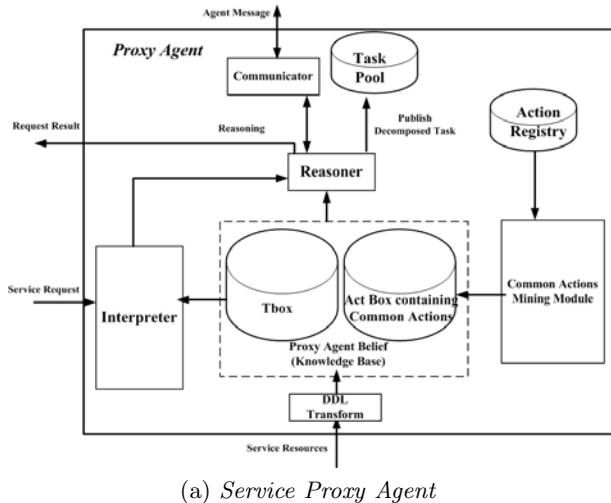
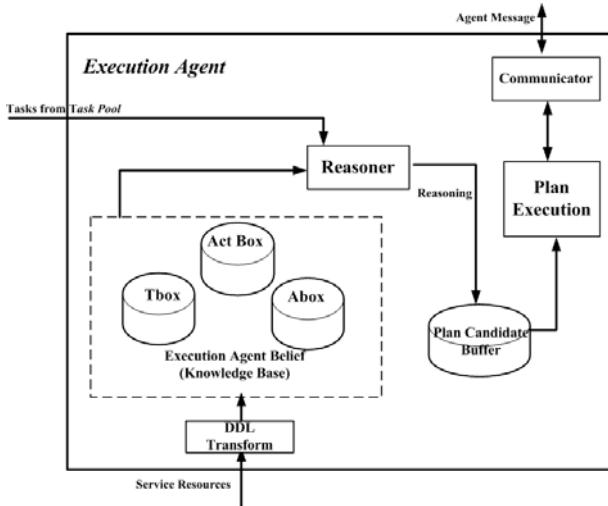
As a software model developed for programming intelligent agents, *BDI* model [3] is characterized by an agent's *beliefs*, *desires* and *intentions*. In general, *beliefs* represent the informational state of the agent, *desires* represent the motivational state of the agent, and *intentions* represent the deliberative state of the agent.

Considering the fact that in a real application scenario, the *provider agent* can send requests to *requester agents* for the third-party service evaluation, while the *requester agent* can also provide its discovered service to the *provider agents*, these two agents are unified in real world applications. Hence in this paper we call this unified agent as the *service execution agent*. While for the *broker agent*, we can refer to it as *service proxy agent* because of its sophisticated capability in *service discovery* and *composition*.

Service Proxy Agent. As shown in Fig. 1(a), the *service proxy agent* contains eight main modules: *Proxy Agent Belief*, *Action Registry*, *Common Action Mining Module*, *Interpreter*, *Reasoner*, *Task Pool*, *DDL Transform* and *Communicator*.

The *Service Interpreter* will translate the service request into a representation acceptable by the *Reasoner*. The *Proxy Agent Belief* (Knowledge Base) contains the *DDL Tbox* and *DDL ActBox*. The *Action Registry* collects all the registered actions of the managed *service execution agents*. The approach to mining these common actions from the *Action Registry* will be explained in detail in Sec. 4. The *Reasoner* provides two interfaces for corresponding functions. First, the *Reasoner* runs the *DDL* reasoning to decompose a main task into several subtasks. Second, it can be also used to compose the final execution action sequence. The *DDL Transform* module will transform the service resources into the *DDL-based knowledge* stored in the *Knowledge Base*. The *Communicator* in *service execution agent* will be utilized to communication with *service execution agents* to exchange agent messages including the request and response results.

Service Execution Agent. As shown in Fig. 1(b), the *service execution agent* contain six main modules: *Execution Agent Belief*, *Reasoner*, *Plan Candidate Buffer*, *Plan Execution*, *DDL Transform* and *Communicator*.

(a) *Service Proxy Agent*(b) *Service Execution Agent***Fig. 1.** Agent Architectures

The *Execution Agent Belief* (Knowledge Base) contains the *DDL Tbox*, *DDL Abox* and *DDL ActBox*. The *ActBox* in *service execution agent* stores the *DDL*-based agent action. The *Reasoner* in *service execution agent* is mainly responsible for running *DDL* reasoning to generate candidate plans for a given decomposed subtask. The *Plan Candidate Buffer* is responsible for storing the generated plans, while the *Plan Execution* module executes the corresponding plan. The *DDL Transform* will also transform the service resources into *DDL*-based knowledge stored in the *Knowledge Base*. The *Communicator* in *service execution agent* will be used to communication with the *service proxy agent* to exchange agent messages including the request and response results.

3.2 Multi-agent Cooperation for Web Services

Based on the architectures of *service proxy agent* and *service execution agent*, we propose a *Cooperative REasoning-based Multi-Agent (CREMA)* model for Web services (See Fig. ??). In this model, each *service proxy agent* is in charge of a set of *service execution agent*, while each *service execution agent* is responsible for realizing a specific decomposed subtask by agent action-based reasoning. Through communication management and task decomposition, the *service proxy agent* can enables several *service execution agents* cooperatively to accomplish a specific main task. Multi-agent based Web service management consists of the following five processes:

- The *service execution agent* should register itself into the *service proxy agent*.
- The *service proxy agent* will translate the *service request* into acceptable *DDL* input, and form a main *task*.
- The *service execution agent* actively checks the *Task Pool* for those newly added subtasks it can solve.

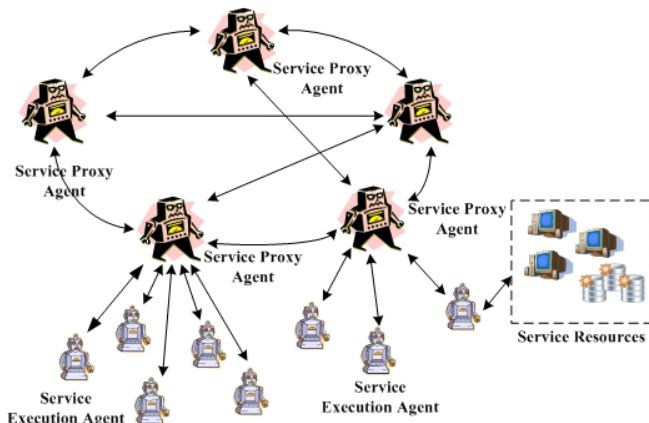


Fig. 2. Cooperate-based Multi-Agent System for Web Services

- If the subtasks decomposed by *service proxy agent* can be accomplished by corresponding *service execution agents*, then the service request will be fulfilled successfully. If in a fixed time, there are still some subtasks which can not be matched to any *service execution agent*, then the *service proxy agent* will take two measures: first, it will forward the request to other *service proxy agent* for help; second, it need generate another possible task decomposition again until each subtask can be achieved by some *service execution agent*.
- For those successful responded subtask, the *service proxy agent* will compose *service execution agent*'s actions to form the final execution action sequence.

In above process, the task decomposition and subtasks execution are essential for the Web service composition. Therefore, such a cooperative reasoning-based multi-agent (*CREMA*) model can support Web service composition well.

4 Mining the Frequent Common Actions

Considering the fact that many actions stored in *Action Registry* possess similar behavior, their common behaviors can be adopted as a concise summary of all those involved similar actions. As the common behavior is concise and is supported by a large amount of *service execution agents*, it is relatively easier to assign the decomposed task to available agents. In this section, we will exploit the similarity between agent actions, with an aim of *identifying the common behaviors to support efficient task decomposition*

4.1 Fundamental Concepts

In order to utilize the common behavior for task decomposition, an approach to automatically discover the common behavior from *Action Registry* is needed.

It is interesting to notice that the common behavior can be regarded as a kind of frequent patterns in actions. Accordingly, the frequent pattern mining method could be adopted in this scenario.

Definition 3 (Imply). For any two actions α and β , α implies β (denoted as $\alpha \Rightarrow \beta$) if and only if $\mathcal{P}_\beta \subseteq \mathcal{P}_\alpha$ and $\mathcal{E}_\beta \subseteq \mathcal{E}_\alpha$.

Informally, $\alpha \Rightarrow \beta$ means that action α supersedes action β .

Definition 4 (Common Action). For a set of actions $\text{ActSet} = \{\alpha_1, \alpha_2, \dots, \alpha_n\}$, the action $\gamma \equiv (P_\gamma, E_\gamma)$ is the common action of ActSet if the support of γ satisfies $\text{sup}(\gamma) = \frac{|\{\alpha_i | \alpha_i \in \text{ActSet} \wedge \delta(\alpha_i) \Rightarrow \gamma\}|}{|\text{ActSet}|} \geq \theta$, where $\theta(0 \leq \theta \leq 1)$ is the threshold.

One basic characteristic of an action is the changes it made to the *Abox* after its execution. Thus we can define the concept of *Action Change Set*:

Definition 5 (Action Change Set). For an action $\alpha \equiv (P_\alpha, E_\alpha)$, its action change set $\text{ChangeSet}(\alpha)$ is defined as $\text{ChangeSet}(\alpha) = \{\neg\phi | \phi \in E_\alpha\}$.

4.2 Action Change Graph

The formulas in an action's pre-condition set \mathcal{P} and the effect set \mathcal{E} could be associated by common variables. For example, in the action *TravelByTrain*, formula $InCity(x_1, y_1)$ and $Source(y_1, w_1)$ are associated by variable y_1 .

To discover common actions, we need to consider the formula association formed by common variables. In this paper, we employ the graph to represent the *Action Change Set* and adopt the frequent pattern mining method [15] to discover common actions. Here we give the definition of *Action Change Graph* which represents the *Action Change Set* of actions.

Definition 6 (Action Change Graph). For an action $\alpha \equiv (\mathcal{P}_\alpha, \mathcal{E}_\alpha)$ and its change set $ChangeSet(\alpha)$, a directed graph $G_\alpha = (V, E)$ is the Action Change Graph of α if and only if there exists an one to one mapping f such that:

- For each variable x in $ChangeSet(\alpha)$, there exists a node $n(x) \in V$ that satisfies $f(x) = n(x)$;
- For each concept C in $ChangeSet(\alpha)$, there exists a node $n(C) \in V$ that satisfies $f(C) = n(C)$;
- For each concept assertion $C(x_i) \in ChangeSet(\alpha)$, there exists an edge $\langle n(C), n(x_i) \rangle \in E$ that satisfies $f(C(x_i)) = \langle n(C), n(x_i) \rangle$;
- For each role assertion $R(x, y) \in ChangeSet(\alpha)$, there exists an edge $\langle n(x), n(y) \rangle$ that satisfies $f(R(x, y)) = \langle n(x), n(y) \rangle$;

According to the definition 6, for every action α , its *Action Change Graph* $G_\alpha = (V_\alpha, E_\alpha)$ can be constructed using the following heuristic rules:

1. For each variable x in $ChangeSet(\alpha)$, add an node labeled with *var* into V_α ;
2. For each concept assertion $C(x) \in ChangeSet(\alpha)$, if there does not exist node $n \in V_\alpha$ that labeled with C , then add a node labeled with C into V_α , add a directed edge $\langle x, C \rangle$ between nodes corresponding to variable x and y , and label the edge with *instanceOf* ;
3. For each role assertion $R(x, y) \in ChangeSet(\alpha)$, add a directed edge $\langle x, y \rangle$ between nodes corresponding to variable x and y , and label it with R ;
4. For each role assertion $\neg R(x, y) \in changeSet(\alpha)$, add a directed edge $\langle x, y \rangle$ between nodes corresponding to variable x and y , and label it with $\neg R$.

4.3 Identifying the Common Actions

According to the definition 6, any nontrivial subgraph of the *Action Change Graph* corresponds to an non-empty action change set, which can be further mapped to an action.

The task of identifying the common actions can be transformed into a graph mining problem of identifying frequent sub-graphs of *Action Change Graphs*. In this subsection, we propose a Apriori-style [1] algorithm to achieve this, and the pseudocode is shown in Alg. 1.

The algorithm firstly transforms each action into the corresponding *action change graph*, and the mining procedure *AprioriGraph* is called to discover frequent *sub-graphs*, finally the discovered frequent *sub-change graphs* are transformed back into the frequent actions via a reverse procedure that is opposite to the procedure which transforms actions into graphs.

The *AprioriGraph* use the frequent edges in change graph that are above the threshold θ as the initial frequent patterns. By combining two size k frequent patterns sharing $k - 1$ common edges, a new pattern of size k is produced. If the produced size k pattern is frequent according to the threshold θ , then the produced pattern is also a frequent pattern. The size k frequent patterns is then used to further discover the size $k + 1$ patterns according to Apriori rules.

Algorithm 1. Mining frequent common actions

Require: An action set $ActSet = \{\alpha_1, \alpha_2, \dots, \alpha_n\}$, support threshold θ ;
Ensure: Return frequent action set $ActSet_{freq}$;

```

1: changeSet  $\leftarrow \{\}$ ;
2:  $L_0 \leftarrow \{\}$ ;
3: for  $\alpha_i \in ActSet$  do
4:   Build change graph  $g_i$  according to  $\alpha_i$ ;
5:   Add  $g_i$  to changeGraphSet;
6:   for edge  $e_i \in g_i$  do
7:     if no edge  $e_j \in L_0$  can match with  $e_i$  then
8:       if  $e_i$  is frequent according to  $\theta$  then
9:         Add  $e_i$  to  $L_0$ ;
10:      end if
11:    end if
12:   end for
13: end for
14:  $freqGraphSet \leftarrow AprioriGraph(changeSet, \theta, L_0)$ ;
15: for  $g \in freqGraphSet$  do
16:    $\alpha_c \leftarrow$  transform  $g$  into action;
17:   Add  $\alpha_c$  to freqActSet;
18: end for
19: Return freqActSet;
```

4.4 Common Action Based Task Decomposition

The *service proxy agent* can decompose the task based on the common action. The task is represented as a tuple $T = (S_{init}, S_{final})$, where S_{init} is a set of formulas describing the initial state, S_{final} is a set of formulas describing the final state. A solution to the task $T = (S_{init}, S_{final})$ is an action sequence $Seq_\alpha = (\alpha_1, \alpha_2, \dots, \alpha_n)$. By sequentially executing the action sequence Seq_α , the state can be transformed from S_{init} to S_{final} . According to the definition 2,

Procedure 2 AprioriGraph(GraphSet $gSet$, support threshold θ , candidates S_k)

```

1:  $S_{k+1} \leftarrow \emptyset;$ 
2: for  $g_i \in S_k$  do
3:   for  $g_j \in S_k$  do
4:     if  $g_i$  and  $g_j$  contains  $k$  common edges then
5:       Combine  $g_i$  with  $g_j$  to a new graph  $g_{new}$ ;
6:       if  $g_{new}$  is frequent in  $gSet$  according to  $\theta$  then
7:         Add  $g_{new}$  to  $S_{k+1}$ ;
8:       end if
9:     end if
10:   end for
11: end for
12: if  $S_{k+1} \neq \emptyset$  then
13:   Return  $S_{k+1} \cup \text{AprioriGraph}(gSet, \theta, S_{k+1})$ ;
14: end if

```

the Seq_α is equivalent to the compound action $\alpha_1; \alpha_2; \dots; \alpha_n$. In order to automatically build a feasible solution to the task T based on common actions, we proposed a forward planning method for task decomposition in Alg. 3.

The function $merit(\cdot)$ in Alg. 3 evaluates the action's fitness as the decomposition cue:

$$merit(\alpha) = \lambda sup(\alpha) + (1 - \lambda) Sim(S_i, S_{final})$$

where S_i is an intermedia state, S_{final} is the goal state, and $\lambda (0 \leq \lambda \leq 1)$ is the linear coefficient, $Sim(\cdot)$ is the function to measure the similarity between two states, and it is defined as

$$Sim(S_i, S_{final}) = \frac{|\{\psi | \psi \in S_{final} \wedge \neg(Conj(S_i) \rightarrow \psi) \text{ is valid}\}|}{|S_{final}|}$$

which is based on the number of formulas in S_{final} that are implied by state S_i .

In Alg. 3, executable actions in Set_A are chosen at first. Then selected actions are sorted according to function $merit(\cdot)$. The action with the highest merit value is adopted as the basis for further task decomposition. If no agent responses to the *service proxy agent*, then current adopted frequent action is discarded and the frequent action with the second highest merit value will be considered.

5 Experiment and Analysis

We implemented the experiments based on our developed multi-agent platform named *Multi-AGent Environment (MAGE)* [22]. To simplify the service resource transforming, we directly operate the *OWL* ontology files for service information input and editing, before storing them in the agent *Knowledge Base*; on the other hand, we let the *Tbox* and *Abox* have the unified global knowledge, so that most attention can put on the agent *ActBox* for action-based reasoning. The *JENA* Tool [18] is adopted for service resource transforming.

Algorithm 3. Task Decomposition based Action Planning

Require: Task $T = (S_{init}, S_{final})$, common action set $Set_A = \{\alpha_1, \alpha_2, \dots, \alpha_n\}$ and action support $Set_{sup} = \{sup(\alpha_1), sup(\alpha_2), \dots, sup(\alpha_n)\}$.

Ensure: Return Plan P;

```

 $S_{current} \leftarrow S_{init};$ 
 $initFlag \leftarrow true;$ 
repeat
  if  $initActFlag$  then
    for each  $\alpha_k = (P_{\alpha_k}, E_{\alpha_k}) \in Set_A$  do
       $candidateActions[i] \leftarrow \{\};$ 
      if  $\neg(Conj(S_{current}) \rightarrow Conj(P_{\alpha_k}))$  unsatisfiable then
        Add  $\alpha_k$  to  $candidateActions[i]$ ;
      end if
    end for
    Add  $\alpha \in candidateActions[i]$  to a priority queue  $Q(i)$  according to  $merit(\alpha)$ ;
  end if
  Select the first action  $\alpha$  from  $Q(i)$ ;
  Remove  $\alpha$  from  $Q(i)$ ;
   $S_{temp} \leftarrow E_{\alpha_k}$ ;
  Put the task  $T = (S_{current}, S_{temp})$  into service pool and wait for response from service execution agent;
  if Receive Response then
    Choose the agent, denoted as  $A$ , that provide best QoS service;
     $\beta = (P_\beta, E_\beta) \leftarrow$  action to accomplish  $T$ ;
     $S_{next} = \{\phi | \phi \in S_{current} \wedge \neg\phi \notin E_\beta\} \cup S_{temp}$ ;
    Put the tuple  $(S_{current}, S_{next}, A, \beta)$  into stack planStack;
     $S_{current} \leftarrow S_{next}$ ,  $i \leftarrow i + 1$ ,  $initFlag \leftarrow true$ ;
    Continue loop;
  end if
  if No response after a time period then
    if  $Q(i)$  is empty then
      Popup top element  $(S_k, S_{k+1}, A, \beta)$  from planStack;
       $i \leftarrow i - 1$ ;
      if  $i < 0$  then
        Exit loop;
      else
         $i \leftarrow i + 1$ ,  $initFlag \leftarrow true$ ,  $S_{current} \leftarrow S_k$ ;
      end if
    else
       $initFlag \leftarrow false$ ;
    end if
  end if
  until  $\neg(Conj(S_{current}) \rightarrow Conj(S_{final}))$  unsatisfiable
  Return planStack;

```

All the experiments are done on a 2.0GHZ Intel Pentium-4 PC with 2GB main memory, running Windows XP.

5.1 Experiment Data

As this is the first attempt in *DDL reasoning*-based multi-agent cooperation for Web service composition, there is no existing work for comparison. In the experiment we focus primarily on the performance evaluation with the aim to evaluate how the proposed methods can improve the capacity and efficiency of Web service composition.

We use the *OWL-S-TC* data set [8] as our testing data. The *OWL-S-TC* data set includes 406 *OWL-S* services which are related to six different areas (*communication, economics, education, food, medicine* and *tourism*). In these services, we select 68 typical services as our action templates for agent action generation. In other words, each service execution agent should have the ability to realize one or more services among these 68 services.

5.2 Evaluation on Mining Frequent Action Patterns

For the 68 action templates, we initialize 200 *execution agents*. Then firstly for each *execution agent* we randomly select 2 action templates from the 68 templates, so that each agent will have its special behavior. In this case, the total action number comes to $200 * 2 = 400$. Similarly, we will increase the number of selected action templates and the selected action templates number could become 5, 8, 10 and 15 respectively.

Next, we construct one *service proxy agent* to do the frequent actions patterns mining and the threshold of *support* will be set to be 0.05, 0.10 and 0.20 respectively.

In this experiment, the efficiency of mining performance will be evaluated using time (in *second*). The results of the experiment are shown in Fig. 3, in which the *X-axis* represents the number of *execution agent* actions, while the *Y-axis* represents the mining run time (in *second*).

One observation is that the bigger *support* threshold will lead to less common actions to be mined and accordingly cost less runtime. When the *support* threshold is more than 0.25, it is difficult to mining any common action, so the mining algorithm runs in a very short time. Another is obvious to see the observation that with the *execution agent* action increment, the mining time will increase accordingly. There are two main factors to affect the mining runtime. One is that the action should be firstly constructed into the *action change graph* and the mining process is essentially to do graph-based matching between *graph pattern* and *action change graph*. Another factor is the number of actions. Through analysis we find that, each node in the *action change graph* corresponds to a variable, while in the real applications, an action with ten variables is rare. So the *action change graph* always contains at most ten more or less nodes and the graph-based matching will not put great effect on mining efficiency. As a result, only the number of *execution agent* actions contributes to the mining efficiency. Just as we see, the curves in Fig. 3 climbs in a relative smooth curve. And the mining results are shown in Tab. 1.

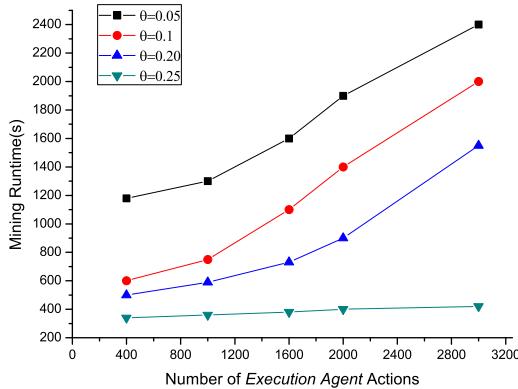


Fig. 3. Efficiency of Mining Frequent Action Patterns

Table 1. Frequent Action Patterns Mining Results

Action Number	Support Threshold	Common Action Number
(400,1000,1600,2000,3000)	0.05	(119,226,324,411,632)
(400,1000,1600,2000,3000)	0.10	(89,188,308,399,581)
(400,1000,1600,2000,3000)	0.20	(65,154,282,368,547)
(400,1000,1600,2000,3000)	0.25	(0,0,0,0)

5.3 Evaluation on Task Decomposition

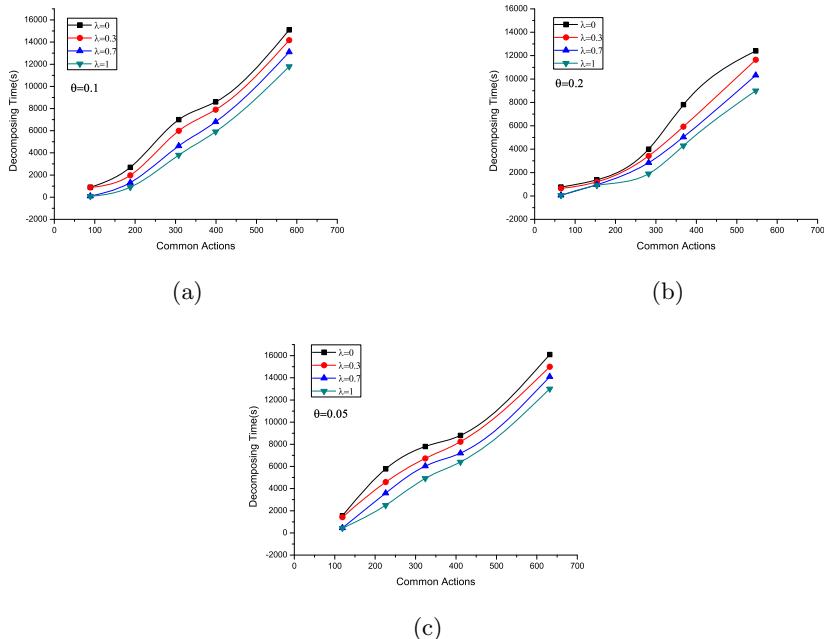
In the task decomposition, there are two key parameters: the coefficient λ which controls the strength of *support*-based merit evaluation; and the θ which represents the *support* threshold for the mining common action. So in this experiment, we will evaluate how these two parameters will affect the decomposing time. Here, we construct totally five main tasks (See Tab. 2) for decompositon. In the next experiment on decomposing time, the decompositon approach will decompose these five main tasks respectively and the time cost is averaged for comparison.

In this experiment, corresponding to the common actions in Tab. 1, for each λ , the five main tasks will be decomposed respectively and computing the average decomposing time (in *second*). The results of the experiment are shown in Fig. 4, in which the *X*-axis represents the number of common actions in under a specific *support* threshold, while the *Y*-axis represents the decomposing time (in *second*).

Table 2. Five Main Tasks in Logic Form

Label	Name	<i>Task S_{init}</i>	<i>Task S_{final}</i>
a	<i>Travel</i>	<i>Person(x), InCity(x, y), ¬InCity(x, z)</i>	<i>¬InCity(x, y), InCity(x, z)</i>
b	<i>BookHotel</i>	<i>Hotel(h), ¬bookHotel(x, h), Person(x)</i>	<i>bookHotel(x, h)</i>
c	<i>BuyBook</i>	<i>Person(x), Book(y), ¬ownBook(x, y)</i>	<i>ownBook(x, y)</i>
d	<i>Visit</i>	<i>Scene(y), Person(x), ¬visited(x, y)</i>	<i>visited(x, y)</i>
e	<i>Eat</i>	<i>Restaurant(r), Person(x)</i>	<i>eatAt(x, r)</i>

As we can see in Fig. 4, with the increment of the number of common actions, the *DDL* reasoning-based task decomposition will present the exponential time consuming, just as any *AI planning*-based method (as discussed in [20]). Moreover, no matter what is the threshold of *support*, a higher λ will lead to a more efficient decomposition. This actually indicates that *the proposed common actions based method could improve the efficiency of the task decomposition*.

**Fig. 4.** Decomposing Time Analysis

6 Conclusions

Multi-agent based Web service management is a new paradigm for Web service. By introducing the multi-agent into Web service, the service resources can be automatically and dynamically managed by agents. Furthermore, the cooperative characteristics enable the agents to collaborate with each other for the accomplishment of complex service requests.

In this paper, a new multi-agent based Web service model named *CREMA* is proposed in order to facilitate the Web service management using agent technologies. The model employs *DDL*-based planning method to enable the interaction between different type of agents. Additionally, frequent pattern mining method is further introduced to identify common actions of the agents. By exploiting those common actions, more efficient and reliable Web services are possible. The empirical studies indicate that the proposed model can improve the efficiency of the Web service composition, and can also improve the throughput of the system under heavy system load.

Acknowledgements. This work was partially supported by the *National Natural Science Foundation of China* (No. 61103158 and 60802066), the *National S&T Major Special Project* (No. 2009ZX03004-001) and the *Securing CyberSpaces Research Cluster* of Deakin University.

References

1. Agrawal, R., Imielinski, T., Swami, A.N.: Minig association rule between sets of items in large databases. In: Proceedings of ACM SIGMOD International Conference on Management of Data, pp. 207–216 (1993)
2. Bellifemine, F., Poggi, A., Rimassa, G.: JADE: a FIPA 2000 compliant agent development environment. In: Proceedings of the Fifth International Conference on Autonomous Agents, pp. 216–217. ACM (2001)
3. Braubach, L., Pokahr, A., Moldt, D., Lamersdorf, W.: Goal Representation for BDI Agent Systems. In: Bordini, R.H., Dastani, M.M., Dix, J., El Fallah Seghrouchni, A. (eds.) PROMAS 2004. LNCS (LNAI), vol. 3346, pp. 44–65. Springer, Heidelberg (2005)
4. Caire, G., Gotta, D., Banzi, M.: Wade: a software platform to develop mission critical applications exploiting agents and workflows. In: Proceedings of the 7th International Joint Conference on Autonomous Agents and Multiagent Systems: Industrial Track, pp. 29–36 (2008)
5. Cao, L., Gorodetsky, V., Mitkas, P.A.: Agent mining: The synergy of agents and data mining. IEEE Intelligent Systems 24(3), 64–72 (2009)
6. Cao, L., Luo, C., Zhang, C.: Agent-Mining Interaction: An Emerging Area. In: Gorodetsky, V., Zhang, C., Skormin, V.A., Cao, L. (eds.) AIS-ADM 2007. LNCS (LNAI), vol. 4476, pp. 60–73. Springer, Heidelberg (2007)
7. Cao, L., Zhang, C.: F-trade: an agent-mining symbiont for financial services. In: Proceedings of the 6th International Joint Conference on Autonomous Agents and Multiagent Systems, p. 262. ACM (2007)

8. Central, S.: Owls-tc: An OWL-S service retrieval test collection (May 22, 2008), <http://projects.semwebcentral.org/projects/owlstcV2>
9. Chang, L., Shi, Z., Qiu, L.R., Lin, F.: A tableau decision algorithm for dynamic description logic. *Jisuanji Xuebao/Chinese Journal of Computers* 31(6), 896–909 (2008)
10. Chang, L., Shi, Z., Gu, T., Zhao, L.: A family of dynamic description logics for representing and reasoning about actions. *Journal of Automatic Reasoning* (2011)
11. Curbera, F., Duftler, M., Khalaf, R., Nagy, W., Mukhi, N., Weerawarana, S.: Unraveling the Web services web: an introduction to SOAP, WSDL, and UDDI. *IEEE Internet Computing* 6(2), 86–93 (2002)
12. Dickinson, I., Wooldridge, M.: Agents are not (just) web services: considering bdi agents and web services. In: Proceedings of the 2005 Workshop on Service-Oriented Computing and Agent-Based Engineering (SOCABE 2005), citeseer, Utrecht, The Netherlands (2005)
13. Dikenelli, O., Erdur, R.C., Gumus, O.: Seagent: a platform for developing semantic web based multi agent systems. In: Proceedings of the Fourth International Joint Conference on Autonomous Agents and Multiagent Systems, p. 1272. ACM (2005)
14. Fikes, R., Hayes, P., Horrocks, I.: Owl-qla language for deductive query answering on the semantic web. *Web Semantics: Science, Services and Agents on the World Wide Web* 2(1), 19–29 (2004)
15. Han, J., Kamber, M.: Data Mining: Concepts and Techniques. Morgan Kaufmann (2005)
16. Krishnaswamy, S., Zaslavsky, A., Loke, S.: An architecture to support distributed data mining services in e-commerce environments. In: WECWIS, p. 239. IEEE Computer Society (2000)
17. Martin, D., Burstein, M., Hobbs, J., Lassila, O., McDermott, D., McIlraith, S., Narayanan, S., Paolucci, M., Parsia, B., Payne, T., et al.: OWL-S: Semantic markup for web services. W3C Member Submission 22, 2007–04 (2004)
18. McBride, B.: Jena: A semantic web toolkit. *IEEE Internet Computing* 6(6), 55–59 (2002)
19. McGuinness, D., Van Harmelen, F., et al.: OWL web ontology language overview. W3C Recommendation 10, 2004–03 (2004)
20. Niu, W., Li, G., et al.: Multi-granularity context model for dynamic Web service composition. *Journal of Network Computer Applications* 34, 312–326 (2011)
21. Shi, Z., Dong, M., Jiang, Y., Zhang, H.: A logical foundation for the semantic Web. *Science in China Series F: Information Sciences* 48(2), 161–178 (2005)
22. Shi, Z., Zhang, H., Cheng, Y., Jiang, Y., Sheng, Q., Zhao, Z.: Mage: An agent-oriented programming environment. In: Proceedings of the Third IEEE International Conference on Cognitive Informatics, pp. 250–257. IEEE Computer Society (2004)
23. Wu, D., Sirin, E., Hendler, J., Nau, D., Parsia, B.: Automatic web services composition using shop2. In: Proceedings of the World Wide Web Conference (2003)

Enhancing Agent Intelligence through Evolving Reservoir Networks for Predictions in Power Stock Markets

Kyriakos C. Chatzidimitriou^{1,2}, Antonios C. Chrysopoulos¹,
Andreas L. Symeonidis^{1,2}, and Pericles A. Mitkas^{1,2}

¹ Electrical and Computer Engineering Department

Aristotle University of Thessaloniki

GR 54124, Thessaloniki, Greece

² Informatics and Telematics Institute

Centre for Research and Technology Hellas

GR 57001, Thermi, Thessaloniki, Greece

{kyrcha,achryso}@issel.ee.auth.gr,

{asymeon,mitkas}@eng.auth.gr

<http://issel.ee.auth.gr>

Abstract. In recent years, *Time Series Prediction* and clustering have been employed in hyperactive and evolving environments –where temporal data play an important role– as a result of the need for reliable methods to estimate and predict the pattern or behavior of events and systems. *Power Stock Markets* are such highly dynamic and competitive auction environments, additionally perplexed by constrained power laws in the various stages, from production to transmission and consumption. As with all real-time auctioning environments, the limited time available for decision making provides an ideal testbed for autonomous agents to develop bidding strategies that exploit time series prediction. Within the context of this paper, we present *Cassandra*, a dynamic platform that fosters the development of Data-Mining enhanced Multi-agent systems. Special attention was given on the efficiency and reusability of *Cassandra*, which provides Plug-n-Play capabilities, so that users may adapt their solution to the problem at hand. *Cassandra*'s functionality is demonstrated through a pilot case, where autonomously adaptive *Recurrent Neural Networks* in the form of *Echo State Networks* are encapsulated into *Cassandra* agents, in order to generate power load and settlement price prediction models in typical *Day-ahead Power Markets*. The system has been tested in a real-world scenario, that of the Greek Energy Stock Market.

Keywords: Data Mining, Power Stock Markets, Reservoir Computing, Multi-Agent System, Neuroevolution.

1 Introduction

Agent Technology (AT) has been successfully employed in various aspects of real-world trading and electronic markets [12]. In highly dynamic, uncertain and

multi-player markets, decisions have to be made continuously, within limited time, while data are generated at extreme rates. Thus, agents are considered a suitable candidate for building efficient trading mechanisms, where their intelligence is based on algorithms that are able to adapt, requiring no or limited user input.

Based on authors' prior work [24], we argue that software agents can be employed with Data Mining (DM) capabilities [6], embed useful nuggets of knowledge, and gain a predictive advantage over their competitors. Performing DM on the (residing or streaming) data pool, one may model the problem at hand, validate a hypothesis, or even predict trends and behaviors. Once the pattern is established, one can then interpret and integrate it with other data (i.e., use it in the theory of the investigated phenomenon, e.g., seasonal commodity prices). Regardless of the depth of the understanding and the validity of the interpretation (theory) of the phenomenon, one can extrapolate the identified pattern to predict future events.

To this end, we have fused AT and DM and developed *Cassandra*, an Agent-Based Framework that fully supports Data Mining capabilities for Prediction and Rapid Problem Solving. *Cassandra* adopts a modular, loosely-coupled, 4-layer Service Oriented Architecture (SOA) [4], with *Plug-n-Play capabilities*. Interconnections between layers, as well as the communication interfaces with users or other applications / Web Services, are properly defined. *Cassandra* provides a step-by-step, easy-to-use guide, as well as a respective Web Services Description Language (WSDL¹) specification and data structure, in order to encapsulate prediction models (PMML²). Through *Cassandra*, the user may select the appropriate dataset(s), preprocess it, and select a Data Mining API to build his/her models (WEKA and R are fully supported, other APIs are partially supported also). Results are displayed and stored, while action can be taken (in an autonomous or semi-autonomous manner), when deemed appropriate.

We prove the feasibility of our approach through a demonstrator case for the Greek Power Stock Market, which is a dynamic, partially observable environment, giving room for applying a wide variety of strategic approaches. In order to capture the temporal and non-linear dynamics of the Power Stock Market signals, we employ *Echo State Networks* (ESNs), a state-of-the-art neural network function approximator. In order to promote the autonomy of the multi-agent system deployed, networks are self-adaptive through neuroevolution [22]. Short-term predictions are made for load and price time-series and are tested against standard regression techniques, previously employed in the *Cassandra* system. Results with respect to load forecasting were excellent, while for price forecasting, which is a much more complex time-series, promising.

The rest of the paper is organized as follows: Section 2 provides the background theory with respect to Echo State Networks and Power Market auction environments. Section 3 discusses related work on platforms enhanced with DM capabilities, as well as AT and DM approaches for solving problems related to

¹ <http://www.w3.org/TR/wsdl20/>

² <http://www.dmg.org/v4-0/GeneralStructure.html>

Power Markets. Section 4 presents the *Cassandra* architecture and functionality, while Section 5 discusses the problem domain, the proposed solution, the experiments conducted and the results derived. Finally, Section 6 summarizes work and concludes the paper.

2 Background Theory

2.1 Echo State Networks

The idea behind *reservoir computing* (RC) and in particular Echo State Networks (ESNs) [13] is that a random *recurrent neural network* (RNN), created under certain algebraic constraints, could be driven by an input signal to create a rich set of dynamics in its reservoir of neurons, forming non-linear response signals. These signals, along with the input signals, could be combined to form the so-called *read-out function*, a linear combination of features, $y = \mathbf{w}^T \cdot \phi(\mathbf{x})$, which constitutes the prediction of the desired output signal, given that the weights, w , are trained accordingly.

The basic structure of an ESN is depicted in Figure 1. The reservoir consists of a layer of K input units, connected to N reservoir units through an $N \times K$ weighted connection matrix W^{in} . The connection matrix of the reservoir, W , is an $N \times N$ matrix. Optionally, an $N \times L$ backprojection matrix W^{back} could be employed, where L is the number of output units, connecting the outputs back to the reservoir neurons. The weights from input units (linear features) and reservoir units (non-linear features) to the output are collected into an $L \times (K + N)$ matrix, W^{out} . For this, the reservoir units use $f(x) = \tanh(x)$ as an activation function, while the output units use either $g(x) = \tanh(x)$ or the identity function, $g(x) = x$.

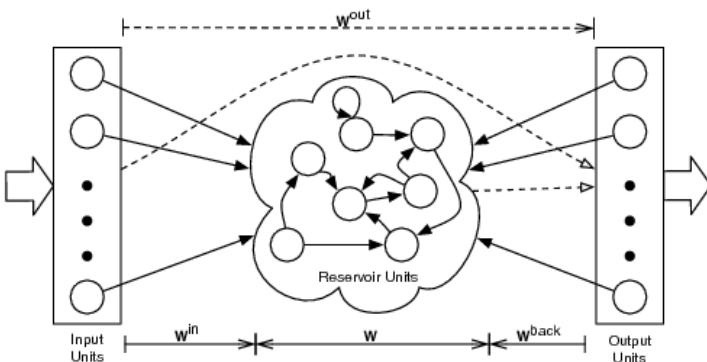


Fig. 1. A basic form of an ESN. Solid arrows represent fixed weights and dashed arrows adaptable weights.

One may refer to [13,14] for best practices for generating ESNs, in the sense of procedures for generating the random connection matrices W^{in} , W and W^{back} . These could be briefly summarized in the following: (i) W should be sparse, (ii) the mean value of weights should be around zero, (iii) N should be large enough to introduce more features for better prediction performance, (iv) the spectral radius, ρ , of W should be less than 1 to practically (and not theoretically) ensure that the network will be able to function as an ESN. Finally, a weak uniform white noise term can be added to the features for stability reasons.

In current work, we consider discrete time models and ESNs without backprojection connections. As a first step, we scale and shift the input signal, $\mathbf{u} \in \mathbb{R}^K$, depending on whether we want the network to work in the linear or the non-linear part of the sigmoid function. The reservoir feature vector, $\mathbf{x} \in \mathbb{R}^N$, is given by Equation 1:

$$\mathbf{x}(t+1) = \mathbf{f}(\mathbf{W}^{in}\mathbf{u}(t+1) + \mathbf{W}\mathbf{x}(t) + \mathbf{v}(t+1)) \quad (1)$$

where \mathbf{f} is the element-wise application of the reservoir activation function and \mathbf{v} is a uniform white noise vector. The output, $\mathbf{y} \in \mathbb{R}^L$, is then given by Equation 2:

$$\mathbf{y}(t+1) = \mathbf{g}(\mathbf{W}^{out}[\mathbf{u}(t+1)|\mathbf{x}(t+1)]) \quad (2)$$

with \mathbf{g} , the element-wise application of the output activation function.

2.2 NeuroEvolution of Augmented Topologies

NeuroEvolution of Augmented Topologies (NEAT) [21] is a topology and weight evolution algorithm of artificial neural networks, founded upon four principles that have established it as a reference algorithm in the area of NeuroEvolution. First of all, the network, i.e. the phenotype, is encoded as a linear genome (genotype), making it memory efficient with respect to algorithms that work with full weight connection matrices. Second, using the concept of *historical markings*, newly created connections are annotated with innovation numbers. During crossover, NEAT aligns parent genomes by matching the innovation numbers and performs crossover on these matching genes (connections). The third principle is to protect innovation through *speciation*, by clustering organisms into species in order for them to have time to optimize by competing only in their own niche. Last but not least, NEAT initiates with minimal networks, i.e. networks with no hidden units, in order to: (a) initially start with a minimal search space and, (b) justify every complexification made in terms of fitness. NEAT complexifies networks through the application of structural mutations, by adding nodes and connections, and further adapts the networks through weight mutation by perturbing or restarting weight values. The above successful ideas could be used in other NE settings in the form of a meta-search evolutionary procedure. Within the context of our work, we adopt the NEAT model in order to achieve an efficient search in the space of ESNs.

2.3 NeuroEvolution of Augmented Reservoirs

NeuroEvolution of Augmented Reservoirs (NEAR) utilizes NEAT as a meta-search algorithm and adapts its four principles to the ESN model of neural networks. The structure of the evolutionary search algorithm is the same like in NEAT, with adaptations made mainly with respect to gene representation, crossover with historical markings and clustering, thus including some additional evolutionary operators related to ESNs. A major differentiation from NEAT is that both evolution and learning are used in order to adapt networks to the problem at hand. A complete description of NEAR can be found in [7].

2.4 Power Market Auctions

The deregulation of the Power Market has given room for the development of open markets, where participants are able to choose between different energy products in different periods of time and may negotiate on their “product portfolio”. These portfolios pertain to three different market regimes:

- *The Long-term Market*, where participants come to direct agreements in form of long-term contracts.
- *The Day-ahead Market*, where buyers place their bids in 24 hourly auctions, in order to establish a contract for the next day.
- *The Real-time Market*, where buyers place their bids in order to establish a contract for the next hour.

In Power Market Auction environments, two are the most important entities:

1. *The Market participants* (or Players)
2. *The Independent System Administrator* (ISA)

A *Player* is defined as a financial entity that accesses the Power Market [25]. In general, this entity may represent a group of Production Units and/or a group of Consumers. Players participating in the Power Market as *Producers*, submit their power supply offers in pre-specified time intervals. Each offer contains the amount of supplying Power, as well as the minimum price one is willing to accept. On the other hand, Players participating in the Power Market as *Consumers* submit their power demands within the same time intervals, along with the maximum price they are willing to pay for it.

The *ISA* is the administrator of the Power System, and also the Administrator of the Power Stock Market (more entities may exist, but they are merged into one for simplicity reasons). Among others, ISA is responsible for the *Settlement Price* of the Power Market, taking into consideration the limitations of the power system. ISA collects bids for each auction and calculates two curves: the *aggregate Supply Curve* (ascending) and the *aggregate Demand Curve* (descending). In the simplified case, where no transmission limitations exist, Settlement of the Market is achieved at the intersection of the two curves (Figure 2). The intersection point specifies the Settlement Price of the Market (SPM), as well as the load to be provided/offered.

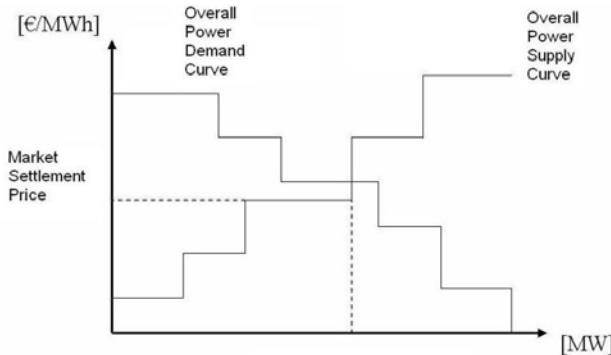


Fig. 2. The aggregate power supply and demand curves

Though demand is mostly covered by Long-term and Day-ahead market transactions, the fact that power cannot be stored for long periods of time dictates the development of a mechanism that can efficiently balance supply and demand of power. Such a balance between production and consumption is ensured through the utilization of a Real-Time Market model. This model bears the highest risk for participants, since the malfunction of an electric node or the shutting down of a power line can bring exaggerated rising or falling of loads and prices. Nevertheless, higher risks also imply profit maximization potential for players willing to participate in this market.

3 Related Work

3.1 DM-Enhanced Software Platforms

There have been several attempts to provide semi-autonomous platforms that fuse Agent Technologies, Data Mining and later on, Web Services.

In the turn of the new millennium, Web Service (WS) technology came to the foreground, resulting to a bloom of platforms based on this new technological achievement. One of the first noticeable implementations is SWORD [18], a set of tools for the composition of a class of web services including “information-providing” services. Nevertheless, the data integration capabilities of the platform did not support standards like WSDL and SOAP, weakening openness and interoperability characteristics of the system.

Following, an agent-based architecture for autonomic web service selection was proposed [15]. This approach selected the appropriate web service from the service directory, employing an aggregated system based on other users’ feedback, trusted sites’ opinion and previous results of the same web service usage. Nevertheless, the platform was lacking composition capabilities, still leaving no room for extensibility.

Working the other way around, some research efforts exist, attempting to build Agent-based Web Services. The Personal Agent Framework for Web Services and Commercial Systems [19] should also be referred to at this line of

work. It is a platform-independent, hybrid artificial intelligent agent framework. Its purpose is to assist a user with processing and retrieving of emails, files, data (from databases), and recreational requests such as searching for an appropriate restaurant or making show reservations, all through a single user interface (portal). The module-based approach is not only appropriate from a software engineering point of view, but it also allows proven artificial intelligence technologies to be incorporated, for example, neural network and automatic speech recognition models.

Finally, special note must be made to the work of Diosteanu and Coltfas [10] for implementing an Agent Based Knowledge Management Solution using Ontology, Semantic Web Services and GIS. Within the context of their work, they developed an agent based knowledge management application framework using a specific type of ontology that is able to facilitate semantic web service search and automatic composition. This framework was later on used to develop complex solutions for location based services, supply chain management, etc, though its main orientation was economic applications.

3.2 Power Market Analysis through AT and DM

Various approaches have been employed for analyzing the behavior of Power Markets, some of which employ AT and DM primitives.

In the early 2000s, during the bloom of MAS utilization, the Electric Power Research Institute (EPRI) developed SEPIA (Simulator for Electrical Power Industry Agents), a multi-agent platform capable of running a plethora of computing experiments for many different market scenarios [2,1]. The Argonne National Laboratory, on the other hand, developed EMCAS (Electricity Market Complex Adaptive System) [8], an efficient implementation for handling the Electric Energy Market. Through EMCAS, one may study the complex interactions between the physical entities of the market, in order to analyze the participants and their strategies. Players' learning is based on genetic algorithms, while EMCAS supports stock market transactions, as well as bipartite contracts. Finally, Petrov and Sheble [17] introduced genetic programming in their simulation and tried to model the bipartite Auctions for the Electric Energy Market, by the use of agents. One of the players incorporates knowledge represented as a Decision-Making Tree, which is developed by genetic programming. The rest of the agent-players incorporate ruled-defined behaviors.

Special attention should be drawn to work by Melzian [16], who developed EMSIM (Energy Market SIMulator), in an effort to derive a deeper understanding of the bidding behavior at the EEX (European Energy Exchange), the impact of changes in market design and individual strategies of the participants. Additionally, work by Bremer et al. [5] should also be noted. They built an agent-based environment for modeling and simulating adaptive household consumers responding to dynamic electricity pricing. Focus was given on the analysis of household load shifting potential under different tariffs and negotiation strategies.

Although efficient, the framework approaches mentioned in the state-of-the-art chapter are either too focused on a specific application domain or a specific learning mechanism. From a software engineering perspective, one may identify that they cannot adapt to new specifications, they are not expandable and modular, and what is most important, cannot reuse the existing infrastructures. From a learning/adaptation perspective, one may identify that they cannot dictate change in the course of action (unless the user directly asks them to). Instead of focusing on the exploitation of Data Mining extracted knowledge in order to augment agent intelligence and autonomy, existing approaches merely focus on solving a very specific problem or selecting the right web service for the given application.

This is the main reason why *Cassandra* is built as a general-purpose prediction tool. Different prediction types and different DM algorithms are supported, while *Cassandra* can manipulate various data types and various datasets. We demonstrate the power of *Cassandra* by focusing on the newly added algorithms, ESNs and NEAR. With respect to power load forecasting, ESNs have been studied in [20,3], where data were drawn from the “World-wide competition within the EUNITE network”³. Neither of the approaches optimize topology, reservoir properties and weights at the same time, in order to adapt the function approximator with minimum human intervention. Additionally, besides load forecasting, we deal with settlement price forecasting as well, which is a much more demanding problem.

4 The Cassandra MAS

4.1 Cassandra Scope of Use

Cassandra follows a hybrid Agent Oriented Software Engineering approach, integrating AT, DM and WS technologies, in order to provide a powerful schema for Prediction and Rapid Problem Solving. Combining the AT approach with WS leads to a uniform representation of services and offer a greater degree of interoperability when using heterogeneous agents and service providers. On the other hand, DM is employed in order to generate knowledge models from data, which can then be dynamically embedded into agents. As new data accumulate, the process can be repeated and the decision structures can be updated, effectively retraining the agents and improving system efficiency.

Looking at the bigger picture, the fusion of these three leading technologies can lead to a wide spectrum of autonomous data mining-enhanced tools covering most business and personal needs. On this basis, *Cassandra* can reach its full potential on environments with sufficient amount of data available (real time data, historical data, correlated data storages etc.) utilized for DM purposes. Based on the processed data, autonomous models are developed, making *Cassandra* an invaluable tool in use cases where such Business Intelligence implementations are needed.

³ <http://neuron.tuke.sk/competition/index.php>

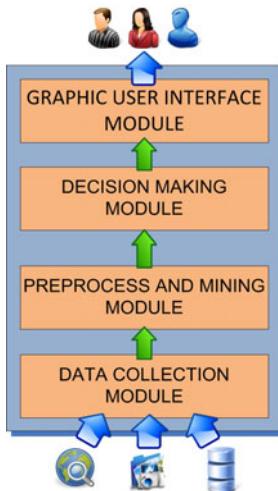


Fig. 3. Cassandra 4-Layer Architecture

4.2 Architecture

Cassandra follows the IRF Architecture Model (Intelligent Recommendation Framework) [23], which defines a 4-layer functional model for the agent system. IRF is usually employed in enterprises for the optimization of the administration mechanism, since it can automate and integrate all the data producing or data demanding facets of a company. Taking a closer look of the Power Market through the IRF prism, one may identify several tasks that have to be tackled:

- Collection of the historical data from previous auctions and processing of the data.
- Application of the suitable DM algorithms in order to build the necessary forecasting models for the values in question.
- Integration of generated models in the Business Intelligence of the System and evaluation of the results.
- Continuous monitoring of the power stock market in order to validate the efficiency of the platform.

As expected, *Cassandra* employs a modular architecture (Figure 3), where each module is responsible for one of the aforementioned tasks. The platform also provides a wrapper around all modules and ensures communication with the system users. The modules comprising *Cassandra* are:

1. Data Collection Module (DCM): Collecting historical data, either from files provided by the user, or directly via an API or the web.
2. Data Processing and Mining Module (DPMM): Processing datasets, preparing training sets and applying DM algorithms.

3. Decision Making Module (DMM): Aggregating all raw data and derived knowledge models in order to make the optimal decision in any given occasion.
4. Graphic User Interface Module (GUIM): Interacting with the System Users. It must be user-friendly, and easily comprehensive.

The most important feature of *Cassandra*, though, is its Plug-N-Play capability; it has the ability to replace any of the four basic modules easily, thus allowing for full expansion of the tool.

4.3 Cassandra User Roles

Cassandra identifies three types of user roles, each accommodated through a respective system view. Table 1 summarizes the functionality provided to each role, enabled upon user authentication.

Table 1. Cassandra User Roles and their respective functionality

User Type	Views	Privileges	Functionality
Operator	- Managing Only	None	<ul style="list-style-type: none"> - Account Checking - Interface Monitoring - WWW / RSS Browsing - Statistic / Chart Viewing
System Analyst	<ul style="list-style-type: none"> - Monitoring - Manager 	Managing only	<i>All of the above plus:</i> <ul style="list-style-type: none"> - Manual Decision Making - Logger Handling
System Architect	<ul style="list-style-type: none"> - Monitoring - Manager - Analyst 	All	<i>All of the above plus:</i> <ul style="list-style-type: none"> - Agent Overview - Model Retraining - Model Configuration - Model Statistics - Cost Evaluator - Data File Import

- **System Architect.** The *System Architect (SAr)* is an expert user. SAr is the developer of the system, so he/she has absolute knowledge over it. SAr is responsible for the smooth operation (unimpeded operation of the system, satisfaction of the software requirements of the users), as well as its efficiency. SAr also provides domain knowledge, while he/she is responsible for generating the DM models, in order to decide on the best-performing one(s). Finally, SAr checks the decision routines, in order to trace any errors that may occur.
- **System Analyst.** The *System Analyst (SAn)* is a domain expert, not necessarily on the involved technologies, but on the application at hand. SAn cannot interfere directly with the system, but using his/her experience SAn

can easily assess *Cassandra* actions and decide whether the system operates efficiently enough or not. SAn has the authority to manually override *Cassandra* decisions, but he/she cannot change the models used by the system for prediction (that is under the Architect's jurisdiction).

- **Operator.** The *Operator* is the simple user of the system. Operator only overviews and logs actions and decisions made by *Cassandra*. In case the Operator notices something “out of the ordinary” (actions that do not have the expected results), he/she notifies the SAn. SAn must then double check Operators observations and, in case of error, notify the SAr, who will react accordingly in order to optimize system operation.

In order to ensure robustness and reusability, *Cassandra* employs a hybrid architecture comprising a set of auxiliary modules ensuring untroubled integration and easy module replacement, as well as a MAS configuration for building intelligent applications, incorporating domain intelligence and performing DM. The *Cassandra* abstract level Z-Notation (Figure 4) depicts the entities involved and the main *Cassandra* goals.

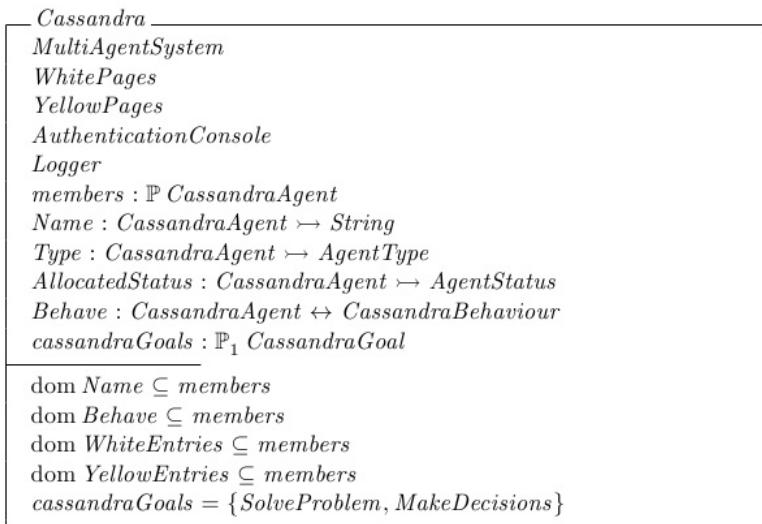


Fig. 4. Schema of the Cassandra platform in Z-Notation

4.4 Cassandra Agent Types and Auxiliary Modules

Cassandra comprises four subsystems, each containing a number of agents (members) with different goals and behaviors. Agents are built upon four basic concepts, well-defined in the schema proposed by *d' Inverno and Luck* [9]:

1. Attributes: a set of perceivable features, varying for every type of agent.
2. Actions: a set of discrete events that can change the state of the environment when performed
3. Goals: a set of states of affairs to be achieved in the environment.
4. Motivations: a set of desires or preferences that may lead to the generation and adoption of goals and that affect the outcome of the reasoning or behavioral task intended to satisfy those goals.

The following attributes are common for all *Cassandra* agents:

- *Name*: Distinctive and unique in order to characterize each agent.
- *Type*: There are many different types of agents developed within the context of *Cassandra*. These types are generic and can be adapted or even replaced, given the user needs. Table 2 lists the supported types and their main functionality.
- *Status*: Agents can be *Ready*, waiting to be assigned with a task to perform, *Busy*, i.e. agents already assigned with a task, *Suspended*, which have postponed operation for the time being, and *Disabled* agents, which are unable to operate.

Table 2. Cassandra Agent Types

Agent Type	Description
Data Collector	Collecting Data From Internet, Databases or Files.
Data Transmitter	Converting files into different extensions and transmitting data to the appropriate agent.
Data Cleaner	
Data Transformer	
Data Integrator	Filtering datasets usign various filtering methods when needed.
Data Reducer	
Data Discretizer	
Curves Agent	Simulating prediction curves including noise and outliers.
Models Agent	Creating and encapsulating prediction models to agents.
Cost Agent	Estimating prediction costs and efficiency.
Launching Agent	Responsible for the initialization of all the other <i>Cassandra</i> Agents.
GUI Agent	Responsible for the Graphic User Interface of <i>Cassandra</i> .
Decision Making Agent	Taking decisions after considering every available parameter of the system.

Cassandra implements four auxiliary modules in order to accommodate the basic operations of the system and agents are successfully integrated with them. These objects are the *WhitePages*, *YellowPages*, *Authentication Console* and the *Logger*.

- The *White Pages* register each new member-agent upon instantiation, while also monitor agents' *Status*. *Enter* and *Exit* functionality defined in the *White Pages* interface correspond to the respective actions.

- *Yellow Pages* are operate in a similar to real-life manner, categorizing agents by type. Whenever a user is in search of a certain type of agent, all he/she has to do is query the *Yellow Pages*. A list of the available agents of the type specified will appear.

- The *Authentication Console* provides the gateway for users to connect and defines the privileges for the different platform roles. The lists of logins and roles can either be hard coded or dynamically defined by the System Architect.

- Finally, the *Logger* is a mechanism that provides a detailed overview of the platform's predictions and actions, in order to monitor and evaluate the efficiency of the system. It also provides an override mechanism for manually bypassing the automated decision making process, in case the System Analyst considers that the prediction made by the *Cassandra* models is not correct.

5 Experiments

To demonstrate the capabilities of the *Cassandra* platform, we have set up a real-life scenario, based on historical data from the Greek Energy Stock Market (<http://www.desmie.gr/>). Figure 5 displays the two time series under considerations, power load and settlement prices, for one week period. One may easily observe how irregularities are more intense in Figure 5b than in Figure 5a, indicating a much more difficult task.

Cassandra employs DM algorithms in order build models capable of forecasting the settlement price, as well as the power load for the day-ahead market. The implemented system may even function in a fully autonomous manner, if granted permission, and may proceed with the necessary actions for establishing a contract. The efficiency of the system is highly dependent on the models generated from historical data, as well as a set of the fail-safe rules specified by the Power Market expert (*Cassandra* System Analyst). Through the *Cassandra* interface (Figure 6), DM models produced are rebuilt dynamically, in order to study, evaluate and compare the results and finally choose the one that optimally projects running market trends (*Cassandra* System Architect).

The collected data span from the 1st of March 2003 to the 30th of September 2010. We have devised two tasks, one for each of the two time series. The tasks are to make daily predictions, for a period of one month. This type of task is used to evaluate the *Cassandra* framework as it would have performed in real life, during one month of simulated operation.

Bids by market participants are sent per hour, bundled over a 24-hour period from time 00 : 00 to 23 : 00 of a single calendar day. Time-series data are available daily, on day d , for that day, after the market clears. They include hourly power load and settlement price values, from time 0h to 23h. Each day the Data Collector agent retrieves the newly available data and stores them to

the repository. The goal is to predict all the 24 hourly values for the next day, $d + 1$, knowing only data available up to date d . Figure 7 presents *Cassandra's* daily cycle.

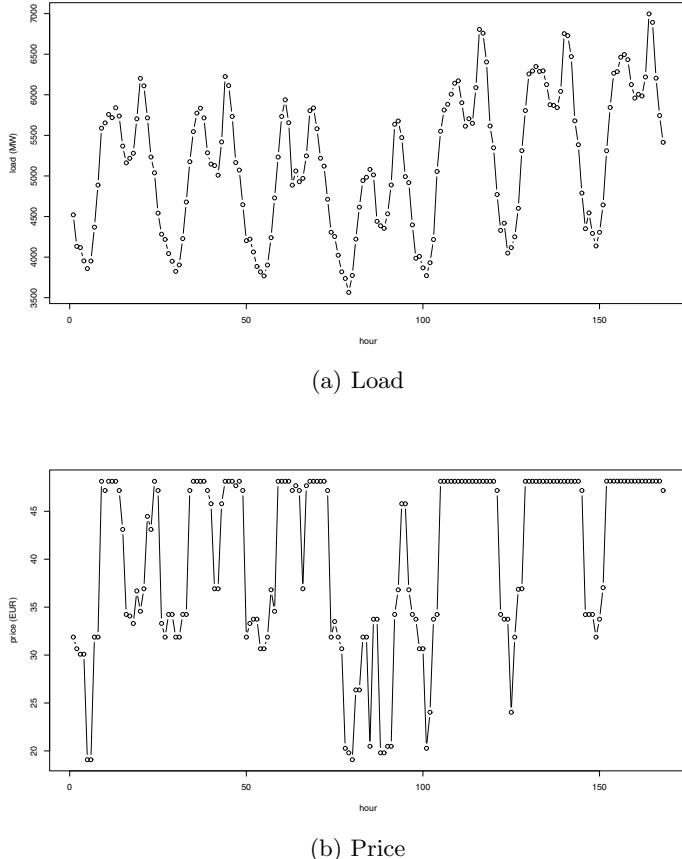


Fig. 5. Power load and settlement price time series for an indicative one week period

We tested the reservoir computing approaches versus benchmark regression algorithm that exist in the WEKA platform [11]: a) linear regression and b) M5' regression trees. We formulated each task as to build models that would predict the value at hour i given all hour values of the previous day:

$$\hat{x}_i^d = f(x_0^{d-1}, x_1^{d-1}, \dots, x_{23}^{d-1}), i = 0 \dots 23$$

where f is the function derived by each algorithm approximating the requested value.

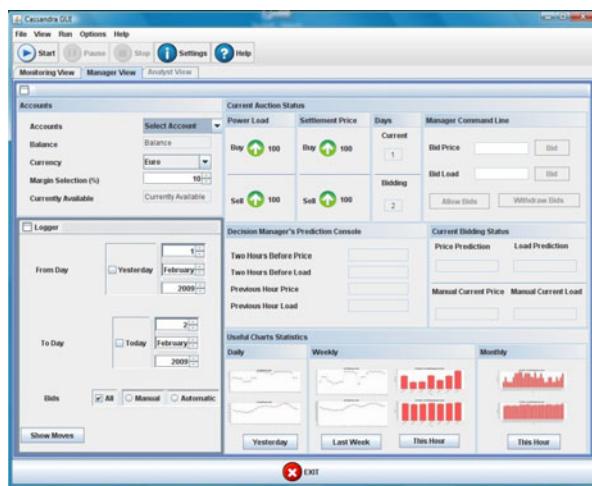


Fig. 6. System Analyst View

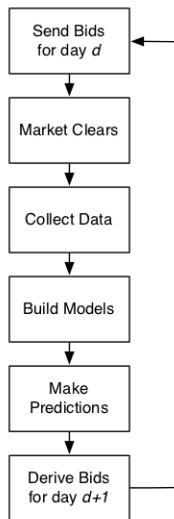


Fig. 7. *Cassandra's* daily cycle

Performance was measures using the Mean Absolute Percentage Error (MAPE) and the maximal error (MAXIMAL) metrics, which are given by equations:

$$MAPE = 100 \cdot \frac{\sum_{i=1}^N \sum_{j=1}^H \frac{|L_{Rij} - L_{Pij}|}{L_{Rij}}}{N \cdot H}$$

and

$$MAXIMAL = \max(|L_{Rij} - L_{Pij}|)$$

where L_{Rij} is the real value at day i and at hour j and L_{Pij} the predicted value.

For the reservoir computing related algorithms (ESN and NEAR), we have used one year (365 samples) as a washout sequence in order to initialize the networks and minimize initial transient phenomena and two years (730 samples) as training data. MAPE and MAXIMAL errors were calculated for each day. Errors are reported over one month period, that of September 2010 (30 samples). In the NEAR case, fitness was calculated for the month prior to the prediction month (October 2010). The fitness function was defined as $1/MAPE$. A population of 50 networks evolved over 100 generations. The same number of initial reservoir neurons were used as in the ESN case. For the ESN approach each network has 25 inputs, one for each of the 24 hour values of the previous day plus a bias input, and 24 outputs stemming out of the reservoir each one estimating the power load of the next day for a predefined hour. ESN parameters used can be found in Table 5. For the standard supervised learning methods (linear regression and M5') 24 datasets are created and 24 models are derived using them, one for each output. The M5' and linear regression implementations are the ones provided by the WEKA API [11].

Table 3 presents average (AVG) and standard deviations (STD) of the test errors over the period of one month, September 2010, for all the algorithms. The MAPE error is low and indicates a quite precise prediction of the power load time series. The reservoir computing paradigm is superior to the other two algorithms due to its ability to maintain a memory and thus calculate temporal features, important to time series predictions. The NEAR methodology was able to optimize the reservoirs and their properties and as a consequence drive the errors even lower.

Table 4, displays the MAPE and MAXIMAL errors for settlement price forecasting. Again NEAR is providing the *Cassandra* platform with the lower errors, but in general MAPE and MAXIMAL error rates are high. This is due to the highly non-linear and dynamic case of settlement price time series. The market clearing protocol is based on vast volumes of laws that make its prediction a very difficult task. On the other hand, load forecasting is based on the behavior of consumers which is much more predictable.

In Figure 8, we present the forecasts made for both power load and market price on the 24th to the 30th of September 2010 period, using the NEAR methodology. The actual and predicted values in Figure 8a are indicative of the

Table 3. MAPE and MAXIMAL error averages (AVG) and standard deviations (STD) over the 30 days period for all the algorithms evaluated for power load forecasting

Method	MAPE AVG (%) 30 Days	MAXIMAL (MW) 30 Days
ESN	2.14	316.13
NEAR	1.91	280.14
Lin. Regression	3.08	436.32
M5'	3.03	570.97
Method	MAPE STD (%) 30 Days	MAXIMAL STD (MW) 30 Days
ESN	1.20	155.16
NEAR	0.87	114.66
Lin. Regression	1.85	163.57
M5'	1.16	178.49

Table 4. MAPE and MAXIMAL error values for ESN and NEAR algorithms with respect to settlement price forecasting

Method	MAPE AVG (%) 30 Days	MAXIMAL STD (Euro) 30 Days
ESN	18.40	28.00
NEAR	16.92	27.22
Lin. Regression	17.58	27.56
M5'	17.00	27.54
Method	MAPE STD (%) 30 Days	MAXIMAL STD (Euro) 30 Days
ESN	6.99	7.82
NEAR	6.27	8.41
Lin. Regression	8.25	8.01
M5'	7.67	7.93

Table 5. ESN parameters initially used (Init.) and ESN parameters derived through NEAR (Found)

ESN Parameter	Load: Init. (Found)	Price: Init. (Found)
Spectral Radius	0.85 (0.79)	0.85 (0.12)
Density	25% (11%)	25% (9%)
Reservoir Activation Function (f)	$tanh$	$tanh$
Output Activation Function (g)	$identity$	$identity$
Reservoir Size	200 (232)	50 (57)
Noise level	10^{-7}	10^{-5}

prediction abilities of ESNs. In the second case, that of Figure 8b, one may identify that the prediction model can follow highly non-linear time series, nevertheless cannot capture unexpected events. In order to do so, more information on the physical constraints of the power system are needed as well as knowledge on the market clearing protocol.

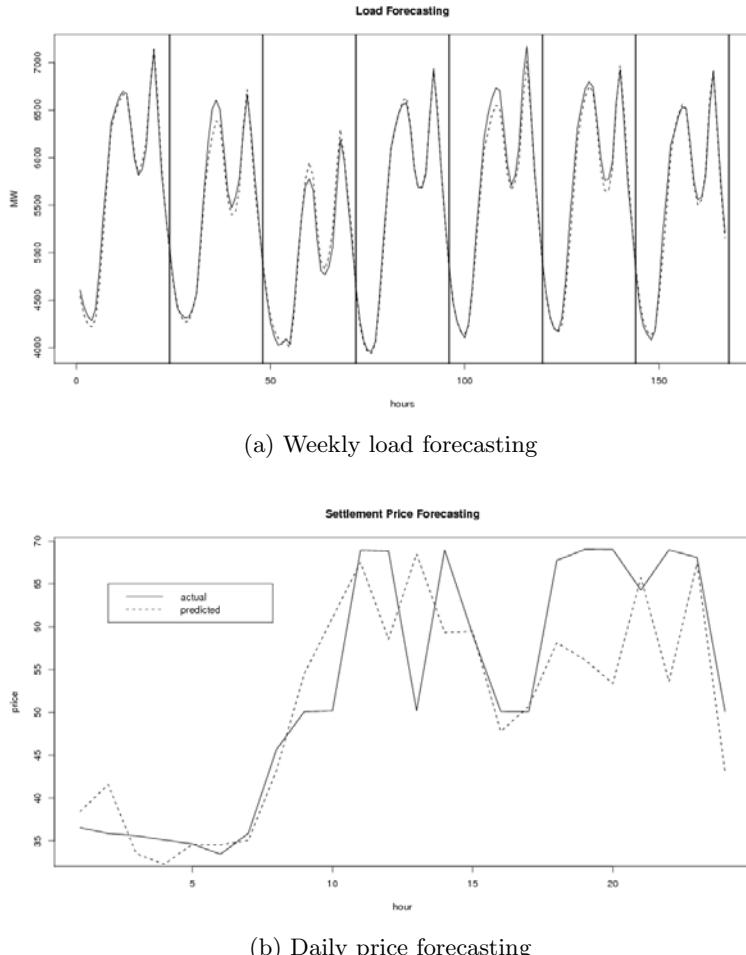


Fig. 8. Load forecasting between the 24th and 30th of September 2010 and price forecasting for the 30th of September 2010

6 Conclusions and Future Work

In this paper, we have discussed framework *Cassandra* as a tool for building MAS enhanced with DM capabilities. Through *Cassandra*, the user may select the ap-

properiate dataset(s), preprocess it, and select a DM API to build his/her models (WEKA and R are fully supported, other APIs are partially supported also). Results are displayed and stored, while action can be taken (in an autonomous or semi-autonomous manner), when deemed appropriate.

We explored the use of the ESN and NEAR methodologies as autonomous adaptation methods of reservoir computing topologies, and applied them in order to enhance the prediction ability of our MAS system with respect to short term power load and market price values in the Greek power stock market. In many tasks, non-standard, specially constructed algorithms that usually don't exist in standard DM packages are required. Using the Cassandra system a large and heterogeneous set of DM algorithms could be easily tested in a semi-automated way or be made to autonomously adapt to the problem at hand.

Prediction of load and prices are very important to both the power system administrators, in terms of the economy and system security, as well as the players involved, giving them strategic advantage over their competitors. Our methodology results in very small error for day-ahead power load forecasts, while for price forecasting it outperforms standard data mining algorithms. We believe that with the incorporation of exogenous factors that interfere with market prices, the prediction capability of the system will further improve.

References

1. Amin, M.: Restructuring the Electric Enterprise: Simulating the Evolution of the Electric Power Industry with Intelligent Agents. In: Electricity Pricing in Transition, pp. 27–50. Kluwer Academic Publishers (2002)
2. Amin, M., Ballard, D.: Defining new markets for intelligent agents. IEEE IT Professional 2(4), 29–35 (2000)
3. Babinec, Š., Pospíchal, J.: Optimization of echo state neural networks for electric load forecasting. Neural Network World 2(7), 133–152 (2007)
4. Bell, M.: Service-Oriented Modeling: Service Analysis, Design, and Architecture. John Wiley and Sons, Inc. (2008)
5. Bremer, J., Andressen, S., Rapp, B., Sonnenschein, M., Stadler, M.: A modelling tool for interaction and correlation in demand-side market behavior. In: First European Workshop on Energy Market Modeling Using Agent-Based Computational Economics, Karlsruhe, pp. 77–91 (March 2008)
6. Cao, L., Gorodetsky, V., Mitkas, P.: Agent mining: The synergy of agents and data mining. IEEE Intelligent Systems 24(3), 64–72 (2009)
7. Chatzidimitriou, K.C., Mitkas, P.A.: A NEAT way for evolving echo state networks. In: European Conference on Artificial Intelligence. IOS Press (August 2010)
8. Conzelmann, G., Boyd, G., Koritarov, V., Veselka, T.: Multi-agent power market simulation using emcas. In: IEEE 2005 Power Engineering Society General Meeting, vol. 3, pp. 2829–2834 (2005)
9. D'Inverno, M., Luck, M.: Understanding Agent Systems. Series on Agent Technology. Springer, Heidelberg (2003)
10. Diosteanu, A., Cotfas, L.: Agent based knowledge management solution using ontology, semantic web services and gis. Informatica Economică 13(4), 90–98 (2009)
11. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The weka data mining software: An update. SIGKDD Explorations 11(1), 10–18 (2009)

12. He, M., Jennings, N.R., Leung, H.: On agent-mediated electronic commerce. *IEEE Transactions on Knowledge and Data Engineering* 15(4), 985–1003 (2003)
13. Jaeger, H.: Tutorial on training recurrent neural networks, covering BPTT, RTRL, EKF and the “echo state network” approach. *Tech. Rep. GMD Report 159*, German National Research Center for Information Technology (2002), <http://www.faculty.iu-bremen.de/hjaeger/pubs/ESNTutorialRev.pdf>
14. Lukosevicius, M., Jaeger, H.: Reservoir computing approaches to recurrent neural network training. *Computer Science Review* 3, 127–149 (2009)
15. Maximilien, M., Singh, M.: Agent-based architecture for autonomic web service selection. In: 1st International Workshop on Web Services and Agent Based Engineering, WSABE 2003 (2003)
16. Melzian, R.: Bidding and pricing in electricity markets - agent-based modelling using emsim. In: First European Workshop on Energy Market Modeling using Agent-Based Computational Economics, Karlsruhe, pp. 49–61 (March 2008)
17. Petrov, V., Shebl, G.: Power auctions bid generation with adaptive agents using genetic programming. In: 2000 North American Power Symposium, Waterloo-Ontario, Canada (October 2000)
18. Ponnekanti, S.R., Fox, A.: Sword: A developer toolkit for web service composition. In: 11th International WWW Conference (WWW 2002), Honolulu, HI, USA (2002)
19. Shia, A.: A novel personal agent framework for web services and commercial systems. In: 2006 International Conference on Semantic Web & Web Services, SWWS 2006. CSREA Press, Las Vegas (2006)
20. Showkati, H., Hejazi, A., Elyasi, S.: Short term load forecasting using echo state networks. In: The 2010 International Joint Conference on Neural Networks (IJCNN), Barcelona, pp. 1–5 (July 2010)
21. Stanley, K.O., Miikkulainen, R.: Evolving neural networks through augmenting topologies. *Evolutionary Computation* 10(2), 99–127 (2002)
22. Stone, P.: Learning and multiagent reasoning for autonomous agents. In: Proceedings of the 20th International Joint Conference on Artificial Intelligence, pp. 13–30 (January 2007), <http://www.ijcai-07.org/>
23. Symeonidis, A.L., Mitkas, P.: Agent Intelligence through Data Mining. Springer, USA (2005)
24. Symeonidis, A.L., Chatzidimitriou, K.C., Athanasiadis, I.N., Mitkas, P.A.: Data mining for agent reasoning: A synergy for training intelligent agents. *Engineering Applications of Artificial Intelligence* 20(8), 1097–1111 (2007)
25. Tellidou, A., Bakirtzis, A.: Multi-agent reinforcement learning for strategic bidding in power markets. In: 3rd IEEE International Conference on Intelligent Systems, London, UK, pp. 408–413 (September 2006)

Pricing Analysis in Online Auctions Using Clustering and Regression Tree Approach

Preetinder Kaur, Madhu Goyal, and Jie Lu

School of Software, University of Technology, Australia
preetinder.kaur@student.uts.edu.au, {madhu,jielu}@it.uts.edu.au

Abstract. Auctions can be characterized by distinct nature of their feature space. This feature space may include opening price, closing price, average bid rate, bid history, seller and buyer reputation, number of bids and many more. In this paper, a price forecasting agent (PFA) is proposed using data mining techniques to forecast the end-price of an online auction for autonomous agent based system. In the proposed model, the input auction space is partitioned into groups of similar auctions by k-means clustering algorithm. The recurrent problem of finding the value of k in k-means algorithm is solved by employing elbow method using one way analysis of variance (ANOVA). Based on the transformed data after clustering, bid selector nominates the cluster for the current auction whose price is to be forecasted. Regression trees are employed to predict the end-price and designing the optimal bidding strategies for the current auction. Our results show the improvements in the end price prediction using clustering and regression tree approach.

Keywords: Online auctions, Price forecasting, Bidding strategies, Data mining, Clustering, Regression trees.

1 Introduction

Predicting the end price of an auction has become an increasingly important area of research because buyers and sellers can be offered a great benefit by using the above predicted prices [1][2][5]. The online auctions are exchange mechanisms which produce a large amount of transaction data. This data can be exploited to forecast the final prices of the auction items, if utilized properly. Researchers' efforts can be noticed in the area of forecasting end-price of an auction using machine learning techniques, functional data analysis, time series analysis [1][2][3][4][5][6][7][8]

Software agent technology is one of the most popular mechanisms used in on-line auctions for buying and selling the goods. Software agent is a software component that can execute autonomously, communicates with other agents or the user and monitors the state of its execution environment effectively [9][10][11][13]. Agents can use different auction mechanisms (e.g. English, Dutch, Vickery etc.) for procurement of goods or reaching agreement between agents. Agents make decisions on behalf of consumer and endeavor to guarantee the delivery

of item according to the buyers preferences. These are better negotiators than human being in terms of monitoring, remembering and emotional influence. In these auctions buyers are faced with difficult task of deciding amount to bid in order to get the desired item matching their preferences. This bid amount can be forecasted effectively by analyzing the data produced as an auction progresses (historical data). This forecasted bid can be exploited by the bidding agents to improve their behaviors. Also the analysis of plethora of data produced in the online auction environment can be done by using DM techniques [1,12][4][7][8][14].

Predicting the end price depends on many factors, such as item type, type of auction, quantity available, opening price, number of bidders, average bid amount and many more. Price dynamics of the online auctions can be different even when dealing with auctions for similar items. Functional data analytical tools have been used to characterize different type of auctions that exhibit different price dynamics in [8]. In this paper, a price forecasting agent (PFA) is proposed using data mining techniques to forecast the end-price of an online auction for autonomous agent based system. A clustering based approach is used to characterize different type of auctions. In the proposed model, the input auctions are clustered into groups of similar auctions based on their characteristics using k-means algorithm. To decide the value of k in k-means algorithm is a recurrent problem in clustering and is a distinct issue from the process of actually solving the clustering problem. The optimal choice of k is often ambiguous, increasing the value of k always reduce the error and increases the computation speed. In this paper, we are exploring Elbow approach using one way analysis of variance (ANOVA) to estimate the value of k. Based on the transformed data after clustering and the characteristics of the current auction whose price is to be forecasted, bid selector nominates the cluster. Regression trees are employed to the corresponding cluster for forecasting the end-price and to design the bidding strategies for the current auction.

The rest of the paper is organized as follows. In section 2 we discuss the related work to the topic. Section 3 illustrates the design of PFA- Price Forecasting Agent describing the data mining mechanism developed for forecasting the end-price and optimizing the bidding strategies for online auctions. Section 4 depicts the experimental results. Section 5 discusses the conclusions of the paper and presents directions for the future work.

2 Related Work

In the recent literature, different approaches have been presented for end price prediction in the online auctions environment. A data mining based multi-agent system has been designed in favor of a multiple on-line auctions environment for selecting the auction, in which the traded item will be sold at the lowest price [4]. The K-means clustering technique has been used to classify auctions into discrete clusters. Clustering operates dynamically on multiple auctions as bid price changes in running auctions. The results of the dynamic clustering

are fed into the agents, and by employing probability-based decision making processes, agents deduce the auction that is most likely to close at the lowest price. Experimental results have demonstrated the robustness of the designed system for multiple on-line auctions with little or no available information.

Forecasting [3] has been proposed as a time series problem and has been solved using moving averages and exponential smoothing models. Authors in this paper also emphasized that there is no one best forecasting technique for a particular set of data. Authors used sequence mining to improve the decision mechanism of an agent for predicting bidding strategy of a set of trading agents [7]. Prediction are mostly based on the continuously growing high dimensional bids history, so authors identified that sequence mining technique can be employed to classify the high dimensional frequent patterns in the bids history. Also the sliding window concept has been used for feeding the continuous classifier. Classification and meta-classification are applied to predict the final bid.

Predicting end price of an online auction has been stated as a machine learning problem and has been solved using regression trees, multi-class classification and multiple binary classification [2]. Among these machine learning techniques, posing the price prediction as a series of binary classification has been proved to be the best suited method for this task. In the literature, along with the machine learning techniques, traditional statistical methods have also been used to forecast the final prices of the auction item, but the experimental results demonstrated that the machine-learning algorithms such as BP networks outperform traditional statistical models [5]. Seeking the effect of clustering of data was also the concern of the authors [5]. Clustering has increased the accuracy of the results in case of BP networks but decreased the accuracy in logistic regression.

A support system for predicting the end-price of an online auction based on the item description using text mining and boosting algorithm has been proposed [6]. Emphasis is given by the authors on capturing the relevant information for incorporating into the price forecasting models for ongoing online auction [1]. A novel functional K-nearest neighbor (fKNN) forecaster based on functional and non-functional distance has been proposed and has been proved to be effective particularly for heterogeneous auction populations.

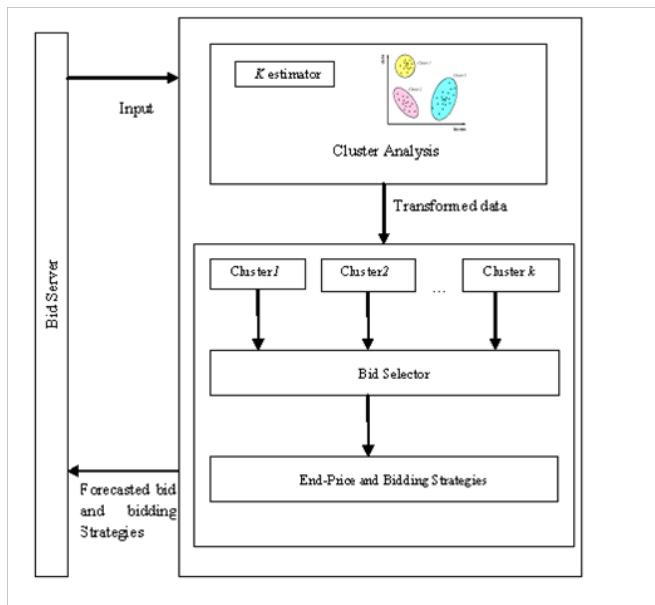
LS-SVM (Least Square Support Vector Machine) algorithm has been introduced by Zhou and Wang for forecasting in online electronic commerce [14]. Authors first improved the SVM to solve the sparsity and time lag problems existing in the traditional method and then they established the LS-SVM online forecast model based on the time factor elimination. Experimental results demonstrated almost same values for the forecasted and the actual price.

3 PFA-Price Forecasting Agent

A clustering based method is used to forecast the end-price of an online auction for autonomous agent based system. In the proposed methodology the input auctions are partitioned into groups of similar auctions depending on their different characteristics. This partitioning has been done by using k-means clustering

algorithm. The value of k in k-means algorithm is determined by employing elbow method using one way analysis of variance (ANOVA). Based on the transformed data after clustering and the characteristics of the current auction, bid selector nominates the cluster for price forecasting. End-price is predicted and bidding strategies are designed by using regression trees for the nominated cluster.

The price forecasting and bidding agent is represented in Fig. 1. Formally our approach consists of four steps. First, data is extracted from the bid server as per the requirements to form the agents knowledge base for online auctions. Let A be the set of the attributes collected for each auction then $A = \{a_1, a_2, \dots, a_j\}$, where j is the total number of attributes. Then based on the auctions characteristics, similar auctions are clustered together. Secondly, k -estimator agent determines the best number of partitions for the overall auction data and then the set of similar auctions are clustered together in k groups. Let C be the set of clusters then $C = \{c_1, c_2, \dots, c_k\}$, where k is the total number of clusters. Thirdly, based on the transformed data after clustering and the characteristics of the current auction, bid selector nominates the cluster for price forecasting. Finally, regression tree is employed to forecast the end price and to design the bidding strategies for the selected cluster.



3.1 K-means Cluster Analysis

The main idea of k-means clustering is to define k centroids, one for each cluster. Initially k data are randomly chosen to represent the centroids. Each data point is assigned to the group that has the closest centroid. After assignment of each data point, positions of the k centroids are re-calculated as the average value of every cluster. These steps repeat until the centroids no longer move or minimizing the objective function J.

$$j = \sum_{j=1}^k \sum_{i=1}^n \|x_i^{(j)} - c_j\|^2 \quad (1)$$

Where $\|x_i^{(j)} - c_j\|^2$ is a chosen distance measure between a data point $x_i^{(j)}$ and the cluster centre. It indicates the distance of the n data points from their respective cluster centers.

K-means clustering technique is used here to categorize different types of auctions based on some predefined attributes from the vast feature space of online auctions. The feature space may include average bid amount, average bid rate, no. of bids, item type, seller reputation, opening bid, closing bid, quantity available, type of auction, duration of the auction, buyer reputation and many more. In this paper, to classify different types of auctions, we focus on only a set of attributes; opening bid, closing price, number of bids, average bid amount and average bid rate. Now A={*OpenBi*, *ClosePi*, *NUMi*, *AvgBi*, *AvgBRi*}

Where A is the set of attributes for an auction.

OpenBi is the starting price of *ith* auction

ClosePi is the end price of *ith* auction

NUMi is the total number of bids placed in *ith* auction

AvgBi is the average bid amount of *ith* auction and can be calculated as *Avg(B₁, B₂, ..., B_l)* where *B₁* is the 1st bid amount, *B₂* is the second bid amount and *B_l* is the last bid amount for *ith* auction.

AvgBRi is the average bid rate of *ith* auction and can be calculated as $\frac{1}{n} \frac{\sum_{i=1}^n B_{i+1} - B_i}{t_{i+1} - t_i}$

where *B_{i+1}* is the amount of (*i*+1)th bid, *B_i* is the amount of *ith* bid, *t_{i+1}* is the time at which (*i*+1)th bid is placed and *t_i* is the time at which *ith* bid is placed.

3.2 K-estimator

To decide the value of k in k-means algorithm is a recurrent problem in clustering and is a distinct issue from the process of actually solving the clustering problem. The optimal choice of k is often ambiguous, increasing the value of k always reduce the error and increases the computation speed. The most favorable method to find k adopts a strategy which balances between maximum compression of the data using a single cluster, and maximum accuracy by assigning each data point to its own cluster. There are several approaches to decide the value of k: rule of thumb, the elbow method, information criterion approach, an information theoretic approach, choosing k using the Silhouette and cross-validation.

In this paper, we are exploring Elbow approach using one way analysis of variance (ANOVA) to estimate the value of k. Number of clusters are chosen so that adding another cluster doesn't give much better modeling of the data. A graph has been plotted for percent of variance against the number of clusters. At some point the marginal gain will drop, giving an angle in the graph Fig. 2. The number of clusters is chosen at this point.

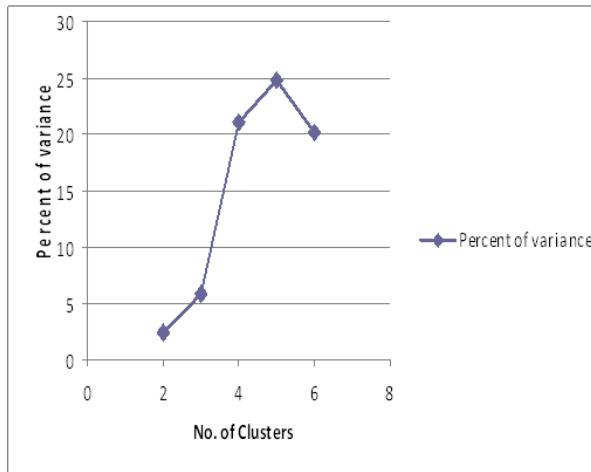


Fig. 2. Choosing the value of k

Below is the mathematical model explained for the value of k.

$$\text{percent of variance} = \frac{\sum_{n=1}^k \sum_{i=1}^{N_n} (X_{in} - \bar{X})^2 - \sum_{n=1}^k \sum_{i=1}^{N_n} (X_{in} - \bar{X}_n)^2}{\sum_{n=1}^k \sum_{i=1}^{N_n} (X_{ij} - \bar{X})^2} \quad (2)$$

Where k is the total no. of clusters N_n is total no. of elements in the n^{th} cluster \bar{X}_n is the mean of distances of auctions from the cluster center in n^{th} cluster X_{in} is the distance of i^{th} auction in n^{th} cluster from its cluster center.

3.3 Bid Selector

In order to decide that the current auction belongs to which cluster, the bid selector is activated. Based on the transformed data after clustering and the characteristics of the current auction, bid selector nominates the cluster for the current auction whose price is to be forecasted.

3.4 Extracting Bidding Strategies

Regression Tree is an analytic procedure for predicting the values of a continuous response variable from continuous predictors. We can derive simple if-then rules

from these predictions. In this paper, we are employing regression tree to predict the end price and to design the optimal bidding strategies for the targeted cluster. This targeted cluster is the cluster which bid selector nominates for the current auction whose end price is to be forecasted. Regression trees are built in two phases. The first phase is growing phase and the second phase is pruning phase. In the growing phase, the tree is grown until no error reduction on the training possible or a pre-defined threshold has been reached. The resulting model usually over-fits the data. This is overcome by the pruning the tree. The best tree is chosen when both the output variable variance and the cost-complexity factor are minimized. The trade-off between these two criterions should be considered. There are two ways to accomplish this. One is test-sample cross-validation and the other is v-fold cross-validation. In this paper, test-sample cross-validation method is used to determine the best pruned tree for designing the optimal bidding strategies of the online auction.

4 Experimentation

In the proposed approach end-price is predicted and the bidding strategies are designed for an online auction by exploiting regressions trees on each cluster which are generated by applying k-means algorithm on the input auctions dataset. The outcome of the proposed model with clustering is compared with the classic (without clustering) model for price prediction. The improvement in the error measure for each cluster for a set of attributes gives support in favor of the proposed model using clustering. Optimal bidding strategies are designed by employing regression trees on each cluster and the model is evaluated by test-sample cross validation approach. Our dataset includes the complete bidding records for 149 auctions for new Palm M515 PDAs. All are 7-day auctions that were transacted on eBay between March and June, 2003 (a sample is available online at http://www.rhsmith.umd.edu/digits/statistics/pdfs_docs/Palm_7day_149auctions.xls) Table 1 shows the statistical information of the data set.

Table 1. Description of the data

	Min	Max	Mean	Std. Deviation
Opening Bid(OpenB)	0.01	240	40.552	69.7142
Closing Price(CloseP)	177	280.65	228.245	16.098
# Bids(NUM)	2	51	21.358	10.155
Avg bid amt(AvgB)	77.473	243.75	148.912	33.133
Avg bid rate(AvgBR)	-2196.268	42679.939	11624.091	8143.416

OpenB: Starting price of an auction.

CloseP: End price of an auction.

NUM: Total number of bids placed in an auction.

AvgB: Average bid amount of each auction.

AvgBR: Average bid rate of each auction.

The value of k for k-means algorithm is estimated by Elbow approach using one way analysis of variance (ANOVA). Percent of variance is calculated after performing clustering for subsequent values of k using k-means algorithm for estimating the point where marginal gain drops. In our experiment, this point occurs after five numbers of clusters as shown in Table. 2. and Fig. 2. So we divide the input space in five clusters considering a set of attributes i.e. opening bid, closing price, no. of bids, average bid amount and average bid rate for a particular auction. These five clusters contain 20.27%, 34.46%, 18.92%, 20.95% and 5.41% of auctions data.

The prediction performance is evaluated using the root mean square error (RMSE) measure. This is used as a measure of the deviation between the predicted and the actual values. The smaller the value of RMSE, the closer is the predicted values to the actual values. Typically the RMSE is defined as

$$RMSE = \sqrt{\frac{1}{m} \sum_{i=1}^m (y_i - \bar{y}_i)^2} \quad (3)$$

Where \bar{y}_i is the forecasted bid for the i^{th} auction based on the estimated model, y_i is the actual bid amount for the i^{th} auction and m is the total no. of auctions in the cluster.

Table 2. Percent of variance after subsequent clustering

No. of Clusters	Percent of Variance
2	2.46
3	5.8
4	21.06
5	24.75
6	20.22

The improvement in root mean square error after clustering for each of the five clusters are 16.43%, 29.47%, 20.76%, 31.17% and 64.91% respectively. This indicated that the proposed price forecasting agent can improve the accuracy of the forecasted price for online auctions.

Optimal bidding strategies are designed by employing regression trees on each cluster. To find the best tree, we have used the test-sample cross validation approach. We randomly divide the data into 80% training and 20% validation set and applied the tree-growing algorithm to the training data and grows the largest tree which over-fits the data. Then the tree is pruned by calculating the cost complexity factor at each step during the growing of the tree and deciding the number of decision nodes for the pruned tree. The tree is pruned to minimize the sum of (1) the output variable variance in the validation data, taken a terminal node at a time, and (2) the product of the cost complexity factor and the number

of terminal nodes. In our experiments, we applied regression trees on the second cluster only and designed the bidding rules for the same. Regression tree for cluster 2 is shown in Fig. 3. The non-terminal nodes present test/decisions on one or more attributes and the terminal nodes reflect the decision outcomes. Fig. 4 gives the rule set that sum up this regression tree.

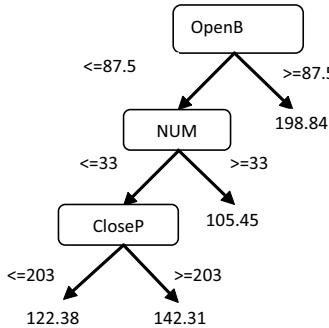


Fig. 3. Regression tree for cluster 2

1 If OpenB	And NUM	And CloseP	Then AvgB:
<=87.5	<=33	<=203	122.38
2 If OpenB	And NUM		Then AvgB:
<=87.5	>=33		105.45
3 If OpenB			Then AvgB:
>=87.5			198.84
4 If OpenB	And NUM	And CloseP	Then AvgB:
<=87.5	>=33	>=203	142.31

Fig. 4. Rule set for cluster 2

5 Conclusions

In this paper we presented a clustering and regression trees based approach to forecast the end-price and to find the optimal bidding strategies of an online auction for autonomous agent based system. In the proposed model, the input auctions are partitioned into groups of similar auctions by k-means clustering algorithm. The recurrent problem of finding the value of k in k-means algorithm is solved by employing elbow method using one way analysis of variance (ANOVA). Then k numbers of regression models are employed to estimate the forecasted price of the online auction. Based on the transformed data after clustering, bid selector nominates the cluster for the current auction whose price is to be forecasted. Regression trees are employed to the corresponding cluster for forecasting the end-price and to design the bidding strategies for the current

auction. The outcome of the proposed model with clustering is compared with the classic model for price prediction. The improvement in the error measure for each cluster for a set of attributes gives support in favor of the proposed model using clustering. Optimal bidding strategies are designed by employing regression trees on each cluster and evaluating the model by test-sample cross validation approach. Further work will be focused in two directions; first, to improve the prediction model by exploiting decision trees and classification methods and secondly, study the importance of each attribute on the end price prediction for online auctions.

References

1. Zhang, S., Jank, W., Shmueli, G.: Real-time forecasting of online auctions via functional K-nearest neighbors. *International Journal of Forecasting* (2010)
2. Ghani, R., Simmons, H.: Predicting the end-price of online auctions. In: *Proceedings of the International Workshop on Data Mining and Adaptive Modeling Methods for Economics and Management, Held in Conjunction with the 15th European Conference on Machine Learning* (2004)
3. Guo, W., Chen, D., Shih, T.: Automatic forecasting agent for e-commerce applications. In: *20th International Conference on Advanced Information Networking and Applications (AINA 2006)*, pp. 1–4 (2006)
4. Kehagias, D.D., Mitkas, P.A.: Efficient E-Commerce Agent Design Based on Clustering eBay Data. In: *IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology Workshops*, pp. 495–498 (2007)
5. Xuefeng, L., Lu, L., Lihua, W., Zhao, Z.: Predicting the final prices of online auction items. *Expert Systems with Applications* 31, 542–550 (2006)
6. Heijst, D., Potharst, R., Wezel, M.: A support system for predicting ebay end prices. *Econometric Institute Report* (2006)
7. Nikolaidou, V., Mitkas, P.A.: A Sequence Mining Method to Predict the Bidding Strategy of Trading Agents. In: Cao, L., Gorodetsky, V., Liu, J., Weiss, G., Yu, P.S. (eds.) *ADMI 2009. LNCS*, vol. 5680, pp. 139–151. Springer, Heidelberg (2009)
8. Jank, W., Shmueli, G.: Profiling price dynamics in online auctions using curve clustering. Technical report, Smith School of Business, University of Maryland, pp. 1–28 (2005)
9. Greenwald, A., Stone, P.: Autonomous bidding agents in the trading agent competition. *IEEE Internet Computing*, 52–60 (2001)
10. Byde, A., Preist, C., Jennings, N.: Decision procedures for multiple auctions. In: *ACM First International Joint Conf. on Autonomous Agents and Multi-Agent Systems*, pp. 613–620 (2002)
11. Anthony, P., Jennings, N.: Evolving bidding strategies for multiple auctions. In: *Proc. of 15th European Conf. Artificial Intelligence*, pp. 178–182 (2002)
12. Cao, L., Gorodetsky, V., Mitkas, P.A.: Agent Mining: The Synergy of Agents and Data Mining. *IEEE Intelligent Systems* 24(3), 64–72 (2009)
13. Goyal, M., Kaushik, S., Kaur, P.: Automated Fuzzy Bidding Strategy for Continuous Double Auctions Using Trading Agents Attitude and Market Competition. *International Journal of Agent Technologies and Systems* 2(4) (2010)
14. Min, Z., Qiwan, W.: The On-line Electronic Commerce Forecast Based on Least Square Support Vector Machine. In: *Second International Conference on Information and Computing Science, ICIC 2009*, vol. 2, pp. 75–78 (2009)

The Impact of Recommender Systems on Item-, User-, and Rating-Diversity

W. Kowalczyk, Z. Szlávik, and M.C. Schut

Department of Computer Science

VU University Amsterdam (NL)

{w.j.kowalczyk,z.szlavik,mc.schut}@vu.nl

Abstract. Recommender systems are increasingly used for personalised navigation through large amounts of information, especially in the e-commerce domain for product purchase advice. Whilst much research effort is spent on developing recommenders further, there is little to no effort spent on analysing the *impact* of them – neither on the supply (company) nor demand (consumer) side. In this article, we investigate the diversity impact of a movie recommender. We define diversity for different parts of the domain and measure it in different ways. The novelty of our work is the usage of real rating data (from Netflix) and a recommender system for investigating the (hypothetical) effects of various configurations of the system and users' choice models. We consider a number of different scenarios (which differ in agents' choice models), run extensive simulations, analyse various measurements regarding experimental validation and diversity, and report on selected findings. The scenarios are an essential part of our work, since these can be influenced by the owner of the recommender once deployed. This article contains an overview of related work on data-mining, multi-agent systems, movie recommendation and measurement of diversity; we explain different agents' choice models (which are used to simulate how users react to recommenders); and we report on selected findings from an extensive series of simulation experiments that we ran with real usage data (from Netflix).

1 Introduction

Recommender engines have made an overwhelming entrance in the world of digital information. Here, electronic commerce is a front-runner, hoping to sell more products with effective recommenders. Still, not only commerce, but many owners of other information systems also start to realise that recommenders may benefit the users in finding items of interest. Whether recommendations are generated by algorithms (e.g., Amazon, Netflix) or in a more social way (e.g., Facebook, Twitter), owners ultimately want to know if their users can better and faster navigate through the available information. On commercial sites, this means whether users buy more items.

Besides tackling technical questions, e.g., on deployment, up-time and responsiveness guarantees and algorithmic optimisation, owners of a recommender want

to know how users react to their engine, and, following from that, how they or the engine can react to these user reactions. This will enable them to dynamically adapt to the customer within the purchasing process – similar to how a human salesperson pitches in order to sell a product.

In this article, we look at the interaction effects between recommendation and user behaviour. We test a movie recommender in combination with a variety of scenarios (i.e., the way that users react to suggestions of the recommender engine) and report on the observed interaction effects. In particular, we look at *diversity* effects, i.e., the property of being *numerically distinct*. We calculate diversities for users, items and ratings. Later in this article, we define different diversity measurements and calculate user-, item-, and rating-diversities for our recommender.

We are the first to conduct such a study, to our best knowledge. Whilst other works have investigated the combination of diversity and recommenders, e.g., [14,10,11], we are the first ones to 1) do this with actual usage data (from Netflix), 2) perform a variety of diversity measurements for different parts of the recommender (item-, user-, and rating-based diversity), and 3) use a combination of data-mining and multi-agent simulation techniques. Also, we are not aware of other work that investigates this issue within the domain of movie recommendation, as we do.

Our work relies on two main technologies: *data-mining techniques* in the recommendation engine, and *agent-based simulation* for the user choice model in the scenarios. Also in this respect, this article can be considered to be breaking ground. Despite the fact that agents have earlier been used for recommendation purposes (as described extensively below), we use agents in a completely different way, namely in order to analyse possible user behaviours and to give engine owners information on how to manage and react to these behaviours. This approach is also novel to the field of recommenders that is traditionally more focused on design and development of optimisation algorithms. As such, our work gives rise to new approaches for simulation-based recommender design and deployment with the user-in-the-loop (by means of agents).

The objective of the study presented in this article is to systematically investigate item-, user-, and rating-diversity effects of a movie recommender system by means of a simulation. This simulation is based on real-world usage data. With this study, we aim to obtain an insight into the effects of recommenders from the owner's perspective (i.e., how can we set up the recommender such that it generates maximum amount of added value?) as well as the consumer's perspective (i.e., what are the consequences of particular ways of reacting to the predictions that recommenders give?). In particular, we aim to answer the following three questions:

1. *What happens if we force users to rate a certain number of items in a period of time (e.g., everyone rates 5 movies a week)?* Such a restriction is an example of how the owner of the recommender can ‘influence’ the behaviour of users.

2. *What is the effect of changing the type of information that a recommender gives to users?* For example, a recommender can show to the user either most popular movies, or movies that match best the user's preferences, or just a random selection of movies.
3. *What is the influence of mixed groups of users in terms of how they react to recommendations?* We may assume that not all users of a recommender react similarly to a recommender. We perform simulation experiments with both homogeneous (everyone reacts the same to recommendations) and heterogeneous user groups.

It is worthwhile to mention that our work is closely related to what is also researched in social and political sciences on group dynamics and political systems [4,9,12]. In these studies, it is also researched how individuals in groups react to each other, both in direct interactions as well through an intermediate entity (the government, or, in our case, the recommender system). The scenarios that we have used are based on general social models concerning group dynamics on peer influence, social networks and collective behaviour. Extensions of our study include the investigation of the particular role of intermediaries in this dynamic process (e.g., recommenders/governments).

This article is organised as follows. In Section 2 we describe related work on movie recommendation and multi-agent systems to position our work; we also present background work on which we have based our work (e.g., specific recommender algorithms). Following that, Section 3 contains the description of the experimental design and setup as well as the simulation scenarios. In Section 4, we elaborate and analyse obtained results from the simulation experiments. Finally, Section 5 concludes and provides pointers for future work.

2 Related Work

In this section, we overview the literature related to the study presented in this article. In general, we focus on agent-based models in combination with (data) mining techniques; and in particular with collaborative filtering engines for movie recommendation. We review literature on these topics among two main dimensions: recommenders and agents. For the first, we describe *design and development* – which is about actually making and designing a recommendation engine algorithm, software and/or framework; *effects of engines* – about the interaction effects between recommendation engines and consumers; and *users* – describing opinions about, the evaluation of and interfaces of recommenders. For the latter, we discuss *consumer behaviour* – on how to realistically simulate consumer behaviour; *communities* – about forming agent-user communities; and *clustering and mining* – about the use of these techniques for recommendation and collaborative filtering.

Our objective is to give a concise overview of existing work at the cross section of movie recommendation and multi-agent systems. This overview consists of a number of exemplar articles at this intercross. We finish this section with an

explanation on how our work relates to the described existing work. In the following section, we go into details of the use of data mining techniques for the Netflix movie recommendation dataset.

2.1 Recommenders

Design and Development. It is not surprising that much effort in the area of recommenders is spent on design and development, since most e-commerce shops have incorporated such technology on their websites. Later in this article, we look specifically if the recommender technology is worth investing in (intuition says it is). First, we give a brief overview of recommender design and development that we consider relevant within the context of this article, i.e., we concentrate on *multi-agent* recommender design specifically for *movies* or television.

In [8], Dehuri *et al.* propose a multi-agent system for the *multiple recommendation problem* (MRP). In an MRP, several personalised recommendations are conducted simultaneously, which can lead to *churning*: users being given many uninteresting recommendations. The proposed multi-agent system is a bio-inspired algorithm that mimics the behaviour of a wasp colony.

Another approach for sharing recommendations is proposed by Weng *et al.* [29], where multiple engines are connected to each other. Although this is not based on multi-agent technology, the developed system (the Ecommerce-oriented Distributed Recommender System, EDRS) has several agent-like properties. The EDRS consists of multiple recommenders from different organisations, in order to improve the service to users (i.e., compared to recommending in a single organisational basis). Among others, the EDRS tries to tackle the *cold-start problem*: instead of a new recommender starting from nothing, it can use collected information from other recommenders in order to form initial recommendations. The distributed variant of recommending can also be seen as an alternative for centralised recommenders that come with privacy and scalability issues, i.e., to use distributed ones to avoid central server failure and protect user privacy. The most significant contribution of the paper is a novel peer profiling and selection strategy.

Lee and Yang [19] propose another multi-agent framework, specifically suited for TV programme recommendations. The framework consists of parts for information gathering, user modelling, and system adaptation; it incorporates a decision tree-based approach (combined with an overfitting-reduction technique) to learn users' preferences. The proposed system contains different types of agents for different programme types (films and news), as well as for learning, collaboration and adaptation. The agents collect information through the set-top-box of the user, on-line (movie and news) sites, and agents of other users, in order to profile a user and offer recommendations. The collaboration with other agents is to overcome the problem of modelling users with only sparse samples.

Finally, Ono *et al.* [20] presents a recommender for constructing movie preference models. Here, data is explicitly acquired through a WWW questionnaire survey instead of collecting data implicitly (e.g., through weblog analysis).

In the work, a novel way is presented of constructing context-aware and multi-applicable preference models using Bayesian networks. Context-awareness relates to a specific situation at hand (for example in a mobile internet scenario); multi-applicability refers to item *promotion* in addition to recommendation. First experimental results show that the approach is promising, despite that the improvement of prediction accuracy is only slight (according to the authors because of the sparsity of the data used for model construction).

Algorithms. The main role of a recommender system is to provide its users with recommendations concerning possible products (e.g., movies, books or services) that would match their preferences or needs. This can be achieved with the help of various algorithms that analyse available data about products and users, such as their (product or user) characteristics, purchase history, explicit feedback provided by users in the form of ratings, product reviews. Within the last two decades many algorithms for this task have been invented. We will provide here a brief overview of the most significant types of these algorithms. A more extensive overview of recommendation algorithms can be found in survey articles of Adomavicius and Tuzhilin [1] or Su and Khoshgoftaar [27].

There are two main strategies for building recommendation engines. The first strategy, *content filtering*, is used in situations when some information about individual users or products is available. This information – attributes that describe various properties of users or products – is used to construct their profiles. These profiles are then used to assign products to users. Another, more general approach which does not rely on external information, *collaborative filtering*, uses information about the past user behaviour (such as users' purchase history) to group users that behave in a similar way and offer them similar products. In some cases both approaches can be combined, leading to *hybrid recommender systems*. There are several ways of combining content-based and collaborative methods; they are outlined in [1].

In some situations users evaluate products by assigning them a *rating* – usually an integer between 1 and 5 that expresses a user's opinion about the given product. When rating information is available, one can measure similarity between products or between users by comparing the row or column vectors of the 'user by product' matrix of ratings. In this way, two similarity measures can be defined: *item-to-item* and *user-to-user*, which can then be used for generating recommendations, Sarwar *et al.* [24].

In 2006 the DVD rental company, Netflix, announced a big competition (with a \$1 million prize) that was aimed at improving the accuracy of their movie recommender system, *Cinematch* [5]. This competition attracted the attention of thousands of researchers from all over the world, leading to the development of several new approaches. Eventually, these efforts led to the improvement of the accuracy of the original Netflix recommendation engine by more than 10%. The new approaches are based on the concept of matrix factorisation of a huge matrix that contains ratings given to items by users. The concept of matrix factorisation, also called low rank matrix approximation, Singular Value Decomposition, or

latent factor model, had already been known and used earlier, for example, in the context of information retrieval [7], but it was Simon Funk (real name Brandyn Webb) who described, on his blog [13], a very simple and efficient modification of this method in the context of the Netflix competition. His algorithm (just a few lines of code!) was good enough to beat the Cinematch baseline ratings by 5%. Not surprisingly, Funk's approach attracted the attention of other competitors, which resulted in numerous improvements of the original method, and finally, in reaching the main objective of the Netflix Challenge (improvement by 10%). Several modifications of Funk's method are described in [18,28].

Effects. In this article, we concentrate on the *effects* of recommenders, rather than design and development. We are particularly interested in the impact of recommenders on item-, user- and rating-diversity, but others have looked at other effects, mainly regarding added business value. We consider three such other works here.

Firstly, Garfinkel *et al.* [14] have developed an empirical method to evaluate the relationship between recommendations and sales. It incorporates 1) indirect impact of recommendations on sales through retailer pricing, 2) potential simultaneity between sales and recommendations, and 3) a comprehensive measure of the strength of recommendations. It was found that the recommendation strengths have a positive impact on sales; sales affect the recommendations; and recommendations have a positive impact on the prices. The authors emphasise and confirm that the research on the business value of recommenders to retailers is only nascent. In the article, a research model is constructed that consists of a set of equations with sales, prices, and recommendation strength as dependent variables. Based on an empirical investigation with panel data, the aforementioned conclusions are derived.

In [10], Benjamin Dias *et al.* look at the business value of a personal recommender in the area of consumer packaged goods. The study was conducted in collaboration with a Swiss e-grocer. The analysis involved shopper penetration, and the (in)direct extra generated revenue. From the analysis it could be concluded that the effect of a recommender is larger than only the direct extra generated revenue (e.g., introducing new categories of products to shoppers). It was also found that accurately updating the model was key to generating extra revenue.

Finally, the work of Fleder and Hosanagar [11] is very closely related to the topic of our paper: it concerns the effect of recommender systems on the diversity of sales. In their work, they investigate (by means of modelling and simulation) whether recommenders help consumers discover new products, or if they reinforce the popularity of already-popular products. The study involves two stages: firstly, path-dependent effects are explored by means of analytical modelling; secondly, by means of simulation, the results are validated with a combination of specific choice models and particular recommender implementations. Results of the study are threefold: firstly, recommenders can indeed create a rich-get-richer effect and reduce sales diversity; secondly, there is a difference between

individual (consumer) and aggregate (overall) diversity: while the former decreases, the latter can actually increase; finally, different recommender design choices affect sales diversity differently.

Users. Another type of recommender effects concerns the way that users interact with the system, e.g., how do users evaluate a particular recommender, does the interface affect a consumer in particular choices, does it influence the opinions of users? Cosley *et al.* [6] investigate how users' opinions are influenced by recommenders. Two features of interfaces are looked at: the rating scale and the display of predictions at the time that users rate items. The study shows that users do not react much to different rating scales, but that they can be manipulated to tend to rate toward the system's recommendations. But users know when the systems tries to manipulate them. An experimental study with 536 users was conducted to answer questions on the way that people respond to recommendations and to see if they noticed when being manipulated. For the first, it was found that users respond to the way the predictions are shown, but it is unknown how long this change in ratings last; for the second, although users would not consciously know they were manipulated, user satisfaction suffered in case the predictions were tinkered with. The study is a first step towards knowing how to deliver accurate predictions to users in a way that creates the best experience for them.

2.2 Agents

Simulation of Consumer Behaviour. Multi-agent models can be used to model consumer behaviour in an intuitive, straightforward way. In our work reported here, we employ a multi-agent system to model choices that users make for watching movies and how they respond to recommended predictions. In the literature, more simulations on consumer behaviour can be found, of which we describe 3 articles here.

Ahn [2] presents a way to use agent-based modelling to imitate user behaviour in internet stores regarding browsing and collecting information. Through this, the rational behaviour of each user is found, which is used to simulate shopping and evaluate different *target customer aid functions*, e.g., personalised pages. Results indicate that personalised pages might not be the best aid function compared to simpler ones.

Ben Said *et al.* [21,23,22] present the CUBES project (CUstomer BEhaviour Simulator) with which one can simulate a consumer population and investigate effects of marketing strategies in a competing market. The project is on the intersection of sociology, marketing and psychology. In CUBES, one can represent observed behaviours as well as simulate large consumer populations. It is possible to simulate behavioural attitudes, impacts of consumption acts, retroactive effects of these acts, and brand reactions. It is shown in [23] through a series of experiments that macro-emergent market phenomena can be explained by behavioural attitudes such as imitation and innovation.

Finally, Ishigaki [17] models consumer behaviour with the help of real-world datasets. The underlying model is a Bayesian network (thus not multi-agent, but because it still concerns simulation of consumer behaviour, we included it here). The model assumes a large-scale dataset (e.g., weblog data, questionnaire data) and tries to explain certain consumer-related factors (e.g., satisfaction level and product valuation). The paper distinguishes different types of models which are increasingly elaborative: process models (e.g., advertisement models), utility models (including data mining) and psychological models (e.g., Stimulus-Response models, information processing models). The proposed computational model fills gaps that these other models have: it is quantitative, it is constructed on actual data, and it deals with uncertainty of consumer behaviour. The model is applied in a retail service application and a content providing service (i.e., movie recommendation). For the latter, details are described in [20].

Communities. Agent communities are a big area of research in multi-agent systems. An agent community is *a stable, adaptive group of self-interested agents that share common resources and must coordinate their efforts to effectively develop, utilize and nurture group resources and organization* [26]. Sen *et al.* [26] have looked at various issues in such communities, e.g., trust-based computing, negotiation, and learning. In recent work [25,15,16], they introduce the Social Network-based Item Recommendation (SNIR) system that consists of agents that are situated in a social network to find items for their peers. The system was used to search and recommend photos on the Flickr photo-sharing social network based on the use of tag lists as search queries. In an experimental study, SNIR outperforms a content-based approach and other recommendation schemes.

Search, Clustering and Mining. The last category of related literature concerns clustering and mining techniques for web data (aka *web usage mining*) as well as search in social networks. This is a wide area of research, and we focus particularly on the combination of such techniques and multi-agent systems.

Alam *et al.* [3] present a web clustering technique for session data based on swarm intelligence, specifically particle swarm optimisation (PSO). The technique clusters on *time* and *browsing sequence* dimensions of the web usage data set. Sessions are modelled as particles for the PSO algorithm; the swarm of agents represents the complete clustering solution. It is demonstrated by means of a series of simulation experiments that the algorithm performs better than *k*-means clustering.

Following up on the previous category of work (agent communities), Gursel *et al.* [15] have developed an agent-based system (closely related to the before-mentioned SNIR) to mine social networks of users to improve search results. In this system, agents observe users' activities as well as the ratings and comments provided by the user to items retrieved from the social network. The process refines search results based on preferences (categories \times contacts), the learning of preferences with mining, category determination, and recommendation (ranking

the items). An experimental study investigates this system to see whether a user has different preferences for the same friend based on item topics. The study shows that the mechanism 1) filters search results by contacts, and 2) ranks results according to user preferences for the users who posted those items.

2.3 Summary

We overviewed related work here. Of these related works, we consider the work of Fleder *et al.* [11] most relevant. However, our conducted analysis is different from theirs in an important aspect: whilst their findings are based on theoretical conduct (still, with *actual* recommendation engines), we based ourselves on real usage data. Also, our scenarios are related to the choice models of Fleder *et al.* but are broader and more flexible: whilst the definition of Fleder *et al.* considers user behaviour (i.e., the way users react to recommendation), we also allow for the owner of the recommendation engine to change its configuration. Therefore, the ‘recommender part’ of a scenario concerns the *information* that is shown to the user (i.e., most popular items, random items, most relevant items). If we know how users react to specific information (e.g., buy more items if most popular items are shown), then the engine owner can anticipate on this.

Our extended notion of ‘choice model’ then makes the work of Cosley *et al.* [6] (described above) very relevant. Without going into the details of that study again, remember that it shows that there are specific effects of how users react to recommendations. This demonstrates that an engine owner has to put significant efforts into deciding what an engine should show to users in order to give them the best experience. For ‘creating such a best experience’, the specific type of *collaborative filtering* (explained below) that we use here (i.e., by means of ratings) is a complicated one. Ratings are a much-used way to elicit and show users’ *preferences* for items. But these preferences are complicated – people do not necessarily always desire to maximise these in the context of recommendation. A rating expresses how good a user finds a particular item – for example, a complicated art-house movie would, in general, be considered better than an average blockbuster. However, this does not mean that users always want to watch art-house movies, e.g., after a week of hard work, one might prefer to settle for a 3-stars blockbuster. Hence, users do not always maximise rating when choosing movies. In comparison with other types of collaborative filtering (e.g., those that show what other items users have seen or bought), this aspect of preferences must always be kept in mind.

3 Experiments

In this section, we describe the data used as the basis of our experiments (Section 3.1), the recommender algorithm we used (Section 3.2), the various simulation scenarios considered (Section 3.3), the setup of our simulations (Section 3.4), and finally, the various aspects of diversity we measured (Section 3.5).

3.1 The Netflix Data

In our experiments we used the original data from Netflix and the basic variant of Funk's algorithm. Now we will describe the data and the algorithm in more detail.

The dataset used in the Netflix Challenge [5] consists of about 100 million records which represent ratings given by about 500.000 users to 17.700 movies over a period of about 6 years. Each record consists of four items: `user_id`, `movie_id`, `date`, and `rating`. Ratings are integers between 1 and 5 that reflect users' satisfaction levels (the higher the better). This dataset served as a *training set*: a recommender algorithm was supposed to use this set to learn how to predict ratings that could be given by users to movies in the future. To test the accuracy of various algorithms Netflix also provided a *test set*: around 3 million records with actual ratings (known to Netflix, but not disclosed to the public) replaced by question marks. The quality of recommender algorithms was measured using Root-Mean-Squared-Error:

$$RMSE = \sqrt{\frac{1}{N} \sum_{i=1}^N (p_i - t_i)^2} \quad (1)$$

where the sum ranges over all N test records, p_i denotes the predicted rating (which is produced by the tested recommender algorithm) and t_i is the true rating that was given by the user to the movie.

3.2 The SVD Algorithm

The algorithm that we decided to use in our experiments is described in [13]. It is based on an assumption that the taste of a user can be captured by a vector of a few numbers called *user features*, while the corresponding characteristics of a movie can be expressed by another vector of the same length, *movie features*. More formally, we assume that for every user u and movie m , there are two vectors of length d : user features, f_u , and movie features, f_m . Additionally, we assume (and this is a very strong assumption!) that the preference (possible rating) of user u with respect to movie m , $p_{u,m}$, can be expressed by a dot product of both vectors:

$$p_{u,m} = \sum_{i=1}^d f_u(i) f_m(i) \quad (2)$$

The values of feature vectors are unknown, but they can be estimated from the training data. Let $r_{u,m}$ denote the actual rating given by user u to movie m (as recorded in the training set). Then the error made by our recommender on the training set is a function of feature vectors:

$$RMSE = \sqrt{\frac{1}{N} \sum_{u,m} \left(\sum_{i=1}^d f_u(i) f_m(i) - t_{u,m} \right)^2} \quad (3)$$

This function, although it involves a huge number of unknowns (d times the number of users plus d times the number of movies), is relatively simple (after ignoring the \sqrt symbol it becomes a polynomial), so its minimum can be found, for example, with the help of the gradient descent algorithm.

When optimal values of feature vectors of movies and users are found, Equation 2 can be used for calculating, for any user and movie, the expected rating the user could give to the movie.

In our simulations we will often have to generate predictions for users or movies that are new and do not have corresponding feature vectors. In such situations we use the following rules:

- if both the user and the movie are new then use the average rating of all available ratings,
- if only the user is new then use the average rating given to the movie by others,
- if only the movie is new then use the average rating given by the user to other movies.

3.3 Scenarios

In order to study how diversity changes under various conditions, it is important to understand the circular nature of a recommender. On one hand, the recommendations a system provides to its users shape the way users act (otherwise the system cannot be considered of good quality). On the other hand, the way users select and, in this case, rate items also affects the recommendations that the system gives in a later stage (assuming that the system is retrained regularly or adapts on-line).

This circular process, illustrated in Figure 1, can be broken down into ‘rounds’ where one round is defined, for our purpose, as a list of user-movie-rating tuples. A tuple indicates that a certain user gave a certain rating to a certain movie, and one round is made up by several people rating several movies. One round has a direct effect on the next round, and so on. Namely, after every round, new ratings are processed by the recommendation engine to update the rating model that will be used in the next round. In other words, the fact that the recommender system iteratively updates its model after every round has a cumulative effect on rounds over time, which is affected by user behaviour. As also shown in Figure 1, both recommender system designers and users can make several choices. For example, a system might offer the most popular movies to users, and users might either behave in an accepting manner, or watch and rate other movies.

In this paper, we select several such system-user choice pairs and investigate their impact of various aspects of the whole process. System-user choice pairs

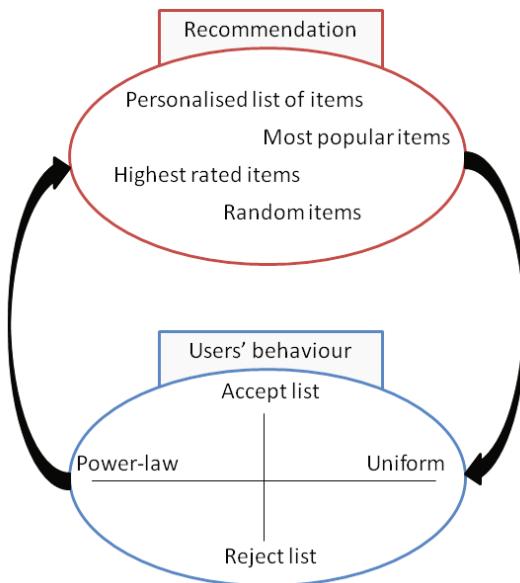


Fig. 1. The lifecycle of a recommender, and choices by system designers and users

define scenarios, and we run a simulation for each scenario. We have identified the following 10 scenarios.

- 1a. The first scenario is made up by the recommender providing personalised lists of items (i.e., the items that are supposed to receive the highest rating) and all users rate those, and only those, items that are recommended to them. The number of items that are recommended to a user is the same as the number of items the user rated in the true data. As the user is supposed to rate all the items offered by the recommender, we call this scenario a **Yes-men** scenario. It is well-known that in the Netflix data the number of items that are rated by individual users follows a power-law. Therefore, sometimes we will refer to this scheme as a power-law or a natural distribution.
- 1b. A very similar scenario is made up by, as above, providing users with personalised recommendations which are then accepted, but this time the number of movies rated in a round is distributed evenly among users. This change allows us to investigate the effect of ‘uniformisation’ in a certain sense. This scenario will be identified in this paper as **Uniform Yes-men**.
- 2a. As the first representative of another group of scenarios, we also define a scenario in which users are offered the most popular movies so far. Popularity is defined as the number of times a movie has been rated since the very beginning of the simulation. Note that this is sensitive to initialisation, but we make an attempt to make this less of an issue, as described in

the next subsection. With this scenario, as before, users accept what they are offered, and the number of movies users rate follow a power-law. This scenario is the **Popularists**.

- 2b. The counterpart of the above scenario is called **Uniform Popularists**, in which the power-law distribution of the previous scenario is replaced by a uniform distribution.
- 3a. We also define a scenario in which users accept what they are given, the number of movies rated per user is distributed according to a power-law, but this time it is the so far highest rated movies that get recommended to users. We identify this scenario as **Trend-followers**.
- 3b. The counterpart of Trend-followers are, obviously, the **Uniform Trend-followers**, where the power-law is replaced by a forced uniformity.
4. The next scenario is significantly different from those discussed so far. In this scenario, movies that are presented to users are selected at random. The number of items that are presented to a user is determined in the same way as in scenario 1a and follows a power-law. This scenario is called the **Randomisers**.
- 5-7. To also investigate mixtures of (power-law based) scenarios, we have made up three scenarios that present a transition from Yes-men to Randomisers. In these three scenarios, we define a probability with which a user either accepts what they are personally offered, or rates a random movie instead. Thus, between the 100% pure Yes-men and pure Randomisers (= 0% Yes-men), we define scenarios called **75% Yes-Randomisers**, **50% Yes-Randomisers**, and **25% Yes-Randomisers**, respectively. These scenarios are to allow us to investigate various mixtures of the two scenarios.

The following subsection describes the simulations in general, which is followed by the description of diversity measures used for the analysis of simulation results.

3.4 Simulations

The outline of a simulation is depicted in Figure 2: firstly, we initialise the recommender, then recommend movies to users, who select movies, then they rate them. Once movies in a round are rated, the recommender (user and movie profiles) is updated, and the next round of simulation starts.

A round, shown as a loop in Figure 2, represents a time period of one month, i.e. we simulate recommendations and movie ratings of one month before updating the recommender and moving on to the next one-month period. Given that we use the Netflix data for calibration (see the following paragraphs), we have 60 rounds available per scenario. The choice of one month was made so that it provided us with a sufficient number of rounds, a feasible time frame for running the simulations, and an intuitive unit of simulation rounds.

In order to keep control over the simulations and to keep them realistic, we ‘calibrated’ our simulations with real data. The data we used for calibration was the Netflix dataset (Section 3.1), to which we refer as the ‘*true dataset*’ in the remainder of this article. The calibration consisted of several components.

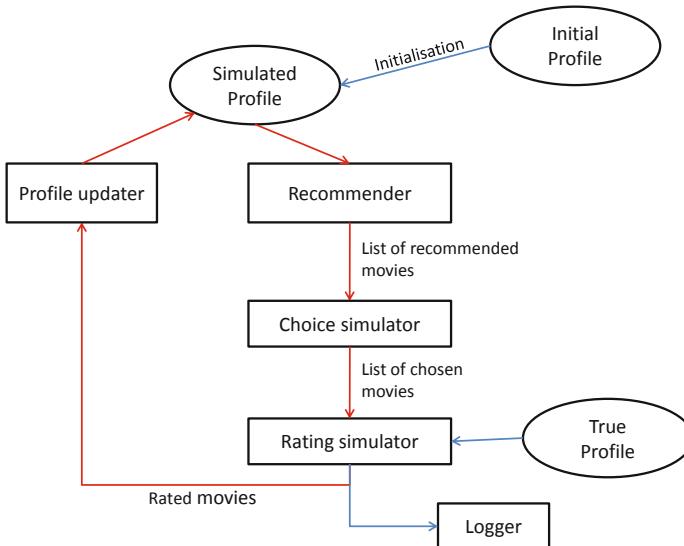


Fig. 2. Simulation outline

Firstly, in each round of the simulation, the system was allowed to recommend only those movies that appeared up to the end of that round in the true dataset. With this method, we aimed at controlling available movies, so the system would not recommend a movie that was only going to be released the following month.

Secondly, to avoid large variations caused by random initialisation, we initialised the recommender of the simulation using user-movie-rating triplets from the first one year of the Netflix data. This allowed us to make comparisons between our simulations and what happened in reality from year two onwards.

Thirdly, we only simulated movie recommendation and selection, but not movie rating. In other words, the scenario determined which movies a user got recommended and chose, but how users actually rated a movie was determined by their True Profiles (TP-s), that were built for them on the whole true dataset. The reason for this choice was, similar to the choices described above, to keep more control over the simulation.

The latter two design choices resulted in three kinds of profiles for a simulation: an **Initial Profile** (IP) was built using one year of data from the true dataset, a **True Profile** (TP) was built on the whole true dataset, and a **Simulated Profile** (SP) was updated in each round of the simulation. A **Simulated Profile** can be viewed as a profile that evolves from an **Initial Profile** over time, as an interaction between users and the recommender system.

The selection of users in a given round was also controlled. In each round, we selected the same users that rated movies in the true dataset in the corresponding period. For example, if there were ratings from 5000 users in the dataset for February 2002, then our simulation also used those 5000 users in the corresponding round. As we used the same ID-s in our simulation as those in the true dataset, we achieved a control over users' decisions to rate movies in a round (or, for that matter, to join and start rating movies at all).

Throughout the simulations, we assumed that no user should rate a movie more than once, which was also the case in the true dataset. Allowing for a movie to be rated several times would not have provided us with valid simulation data. It had been observed in test simulations that users would get the same recommendations in successive rounds if an item would be allowed to be recommended once already rated.

The number of movies that were recommended to a user in a round was determined by the scenario in use (see power-law and uniform versions of some scenarios). To allow for comparison with the real situation reflected by the true dataset, the overall number of movies rated in a simulation round was always the same as in the true dataset.

3.5 Measures of Diversity

For each of the simulations determined by the ten scenarios described in Section 3.3, we measured diversity in various ways. An overview of the used diversity measures is presented in Table 1.

Table 1. Overview of diversity measures used in our experiments

	Unique Movies	Global variance	User-based variance	Entropy	Cosine
Overall	x	x			
Per movie				x	
Per user			x		x
Source data: Binary	x			x	
Source data: 1-5		x	x		x

In addition to the diversity measures described below in more detail, we also calculated the *average rating over all users per simulation round*. It is considered to be positively related to user satisfaction and recommender system performance. We calculated the following diversity values in each round of a simulation:

- The **number of Unique Movies** rated in the given round. This measure allows us to see whether all the movies available in a simulation round were rated by at least one person. It shows diversity in the sense of breadth of rated movies. Note that, by definition, the number of unique movies in a simulation round can not be higher than that of the true dataset for the same period.

- **Global variance of ratings** in a simulation round, and that of the corresponding period in the true dataset. This measures diversity that describes the breadth of values of ratings given in a certain round. Both this and the previous measure are overall measures, i.e., they consider a round’s data without using detailed information about individual users or movies.
- **Mean User-based variance of ratings.** We take the variance of ratings for each user of the current round, and then report the mean of these variances. Note that this measure is not concerned with the number of movies a user rates.
- **Normalised Shannon Entropy of rated movies** in a round. The entropy H is defined as follows:

$$H = - \sum_{i=1}^n p_i \cdot \log(p_i); p_i = \frac{\text{occ}(movie_i)}{\text{count}(ratings)} \quad (4)$$

where n is the number of unique movies rated in a round in question, and $\text{occ}(movie_i)$ denotes the number occurrences of the i th movie. As the maximum value of H depends on the number of movies n , we normalise H by dividing it by $\log(n)$.

- **Cosine diversity.** The Cosine coefficient is a commonly used similarity measure, which we base another diversity measure upon. For each pair of selected users, the vectors U_i and U_j are considered, where $U_{i,m}$ denotes the rating given by user i to movie m , where m varies over all available movies. Then the Cosine coefficient is calculated for each possible pair of users. Because the maximal value of the Cosine coefficient – one – indicates perfect similarity, and we are rather interested in *diversity*, we consider a *Cosine diversity* measure which is defined as shown in Equation 5. For computational efficiency reasons, we calculate the coefficients on a randomly selected sample of $n = 1000$ users. Our tests show that this sample size is sufficiently representative of the whole population of users in a given round.

$$\hat{C} = 1 - \frac{1}{n(n-1)} \sum_{i=1}^{n-1} \sum_{j=i+1}^n \frac{U_i \cdot U_j}{\|U_i\| \cdot \|U_j\|} \quad (5)$$

Having described scenarios, the simulation setup, and diversity measures in this section, the next section presents and discusses results of the simulations that we carried out.

4 Results and Analysis

In this section, we present and discuss the results of our simulations. We do this using six figures: Figure 3 shows the mean ratings per round per scenario, and Figures 4 to 8 show various diversity values that we measured throughout the simulations.

There are a number of aspects from which results of the simulations can be investigated. After comparing our simulation results to the true dataset, we move on to discuss further three aspects as identified in the Introduction:

1. *What is the effect of forcing users to rate the same number of items in each round?*
2. *What is the effect of changing the type of information that a recommender gives to users?*
3. *What is the effect of mixing groups of users of different type?*

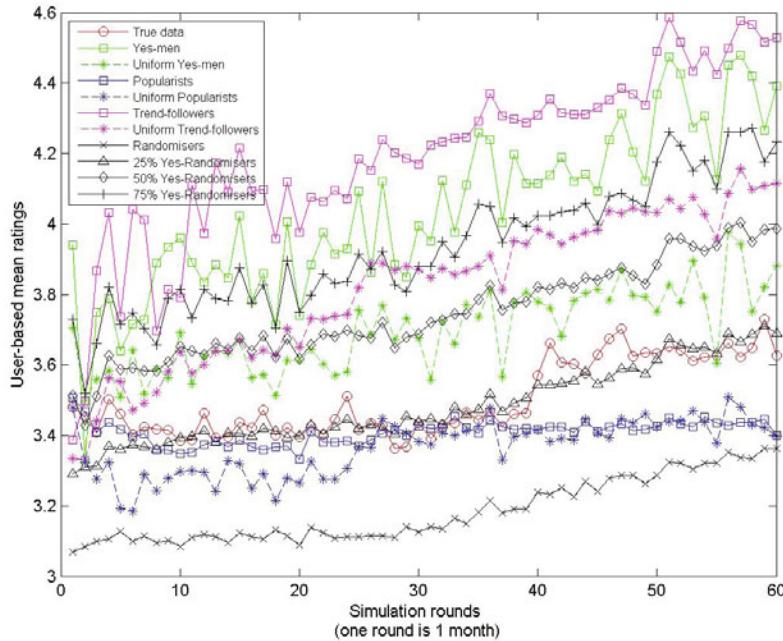


Fig. 3. Mean ratings over users

4.1 True Data and Simulation Results

As stated above, we first compare our simulation results to corresponding measurements on the true dataset.

Figure 3 shows that several of our scenarios resulted in mean ratings consistently higher than values in the true dataset, while some of our simulated values are lower. One of our scenarios with a mixed model (25% Yes-Randomisers) matches values of the true dataset quite closely. In other words, from the mean ratings perspective, when people accept personalised recommendations in 25% of the time, otherwise rate random movies, we get a similar situation to reality.

In terms of variance (Figures 5 and 6), we can observe that the true dataset's variances are consistently and considerably higher than those in our simulations. This observation can be explained as follows. In our simulations, everyone rates

movies according to the same “true rating model” that was built from the Netflix data. Therefore, user ratings are fully consistent with this model. However, it is well-known that humans are inconsistent when they are allowed to rate the same movie several times. Cosley *et al.* [6] measured the consistency of user ratings by asking 212 users to rate some movies and re-rate them again after 6 weeks, collecting in this way 1892 ratings on the 5-stars scale. It turned out that although the mean of new ratings was almost the same as for old ones (the difference in means was about 0.01 stars), the new ratings were quite different from the original ones: 20% of them were lower and 20% were higher. This means that the variance that could be attributed to a non-deterministic rating behaviour of humans is about 0.4. However, to avoid problems with extreme ratings (1 and 5 stars) the experiment considered only ratings that originally had values of 2, 3 or 4 starts. Therefore, we could speculate that when using the full scale of 5 stars, there would be additional 20% of the original ‘5 stars’ ratings re-rated as ‘4 stars’ and another 20% of ‘1 star’ ratings that would be re-rated as ‘2 stars’. In the Netflix dataset 27.65% of ratings had value 1 or 5 what leads to a more realistic estimate of the variance that could be attributed to user inconsistency: $0.3447 = 0.2 * 0.2765 + 0.4 * 0.7235$. Here we assume that a lower or higher rating differs from the original rating by exactly one star. Let us notice that after correcting the true dataset’s variances by 0.3447 (Figures 5 and 6), they will still be slightly higher than the variances measured on simulated data.

According to results displayed in Figure 7, the entropy of rated movies per round is higher than in the true data in several simulations, while lower in some others, suggesting that perhaps some combination of our scenarios would be able to model the true data, at least in the entropy sense.

Values of the Cosine diversity (Figure 8) reveal that the selection and ratings of movies in reality was close to a 1:3 mixture of Yes-men and Randomisers (= 25% Yes-Randomisers). Apart from the pure random scenario, other scenarios resulted in lower Cosine diversity than those already mentioned in this paragraph. In short, in reality, the differences in selections of movies between users are quite high.

Based on the findings above, we can conclude that the true dataset’s diversity can be simulated as a mixture of Yes-men and Randomisers, but only in a limited sense. In other senses of diversity, other mixtures might model reality better. Looking at this from another perspective, if a recommender system owner wants to influence diversity of ratings, e.g., they want higher diversity in some sense, they might want to make different recommendations to different people (e.g., personalised or popularity-based recommendations), or they might also want to influence user behaviour (e.g., by trying to achieve that people act uniformly, so perhaps they might be easier to ‘handle’). In the next subsection, we investigate the effect of forcing users to act uniformly, which is followed by the comparison of what a recommender can offer to users, and that, by the comparison of some mixture scenarios.

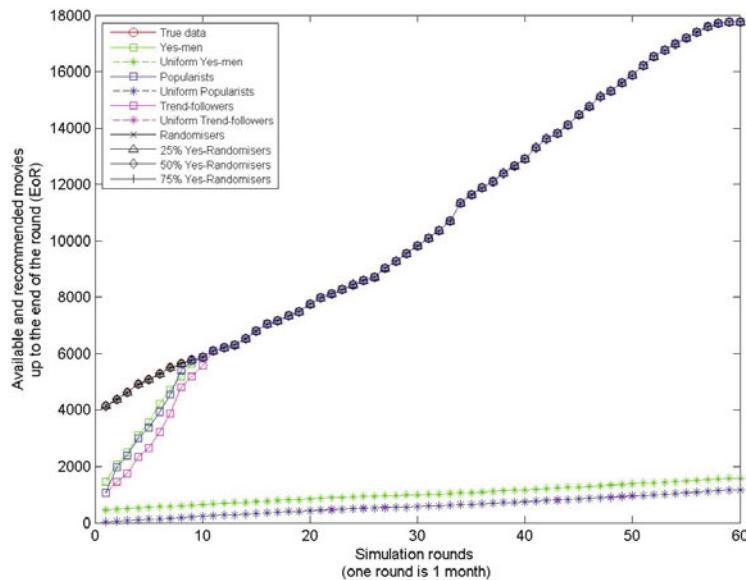


Fig. 4. Number of unique movies rated

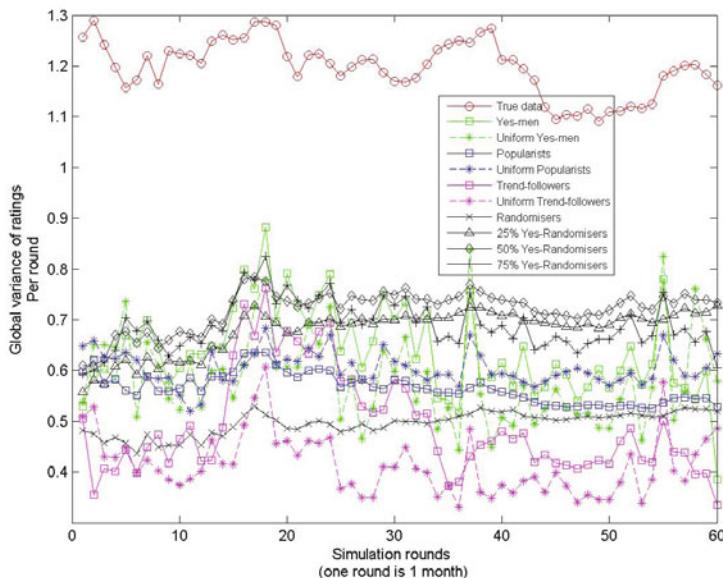
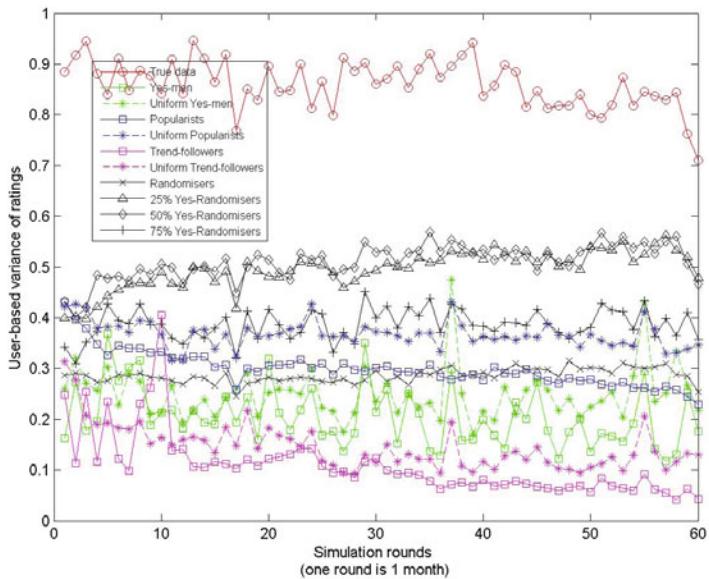
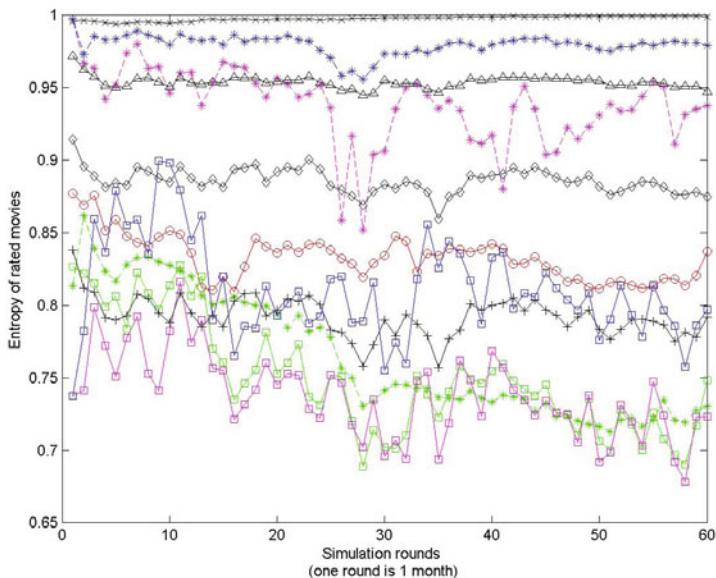


Fig. 5. Global variance of ratings

**Fig. 6.** User-based average variance of ratings**Fig. 7.** Entropy of rated movies

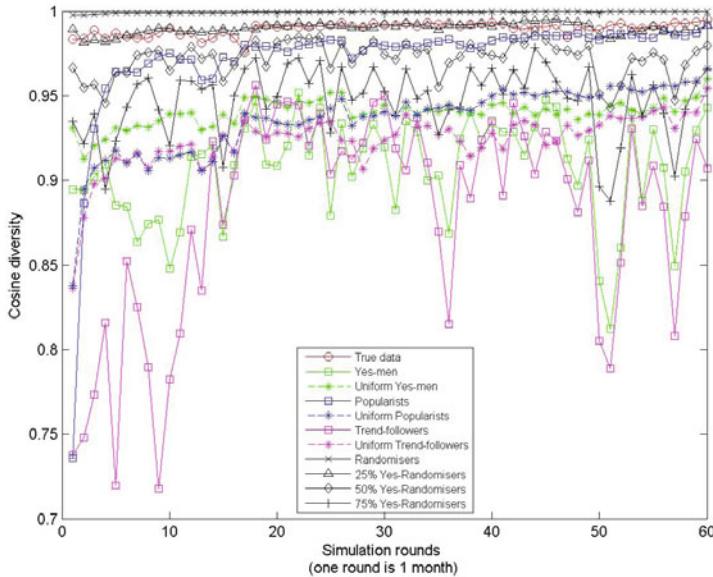


Fig. 8. Mean Cosine diversity values

4.2 ‘Normal’ and ‘Uniform’ Scenarios

To study the effect of every active user rating the same number of movies in a round (i.e., a kind of forced behaviour), we consider six of our ten simulations. We analyse the ‘transition’ from Yes-men to Uniform Yes-men, from Popularists to Uniform Popularists, and from Trend-followers to Uniform Trend-followers, respectively. We attempt to find effects that are common in these three kinds of transitions.

It is an important observation that uniformity, in the sense we described it in the paragraph above, drastically decreases the number of movies ever watched. Figure 4 shows that, by the end of the simulation, in the unforced cases, almost 18.000 movies have been rated by at least one person, while in case of uniformly distributed numbers of ratings per movie, only around 1.000 movies have been rated.

When using the uniform versions of scenarios, the global variance of ratings tends to become lower in case of Yes-men and Trend-followers (Figure 5), however, user-based variance of ratings tends to increase in over three pairs of scenarios (Figure 6). Considering the two inconsistent pairs of scenarios, this is a kind of change also observed in [11], where it was found that the use of a recommender system decreases global diversity but increases individual diversity. As having to use a recommender system can be considered as a forced way of acting in a uniform way, some of our scenarios, which also simulate a forced way

of acting uniformly while using a recommender, are in line with their findings. Popularist scenarios do not confirm this, however.

The entropy of rated movies (Figure 7) is almost uniform, for uniform versions of choice models. This is particularly evident when we consider the differences between Trend-followers and Uniform Trend-followers as well as between Popularists and Uniform Popularists, and it seems to hold partially in case of the Yes-men scenarios. However, if one of these uniformities is not true, i.e., they can either get personalised recommendations or choose as many movies as they want, entropy stays lower, meaning that there will be movies increasingly more/less popular over time.

The Cosine diversity values in Figure 8 show that uniform versions of our models tend to show higher diversity in case of Yes-men and Trend-followers, and not in Popularists. The latter case can be explained by the fact that, almost by definition, the popular movies become even more popular, so Popularists rate a very limited set of movies (just the most popular ones) what automatically reduces the diversity of their ratings. In the former two cases, the number of movies that users have to rate is increasing with the number of required ratings. This leads to the increase of rating diversity.

On a different note, according to our simulations, forced uniformity makes the mean global ratings decrease (Figure 3), unless the mean ratings are very low already. This is probably caused by the fact that many users (in the long tail of the ‘normal’ power-law distribution) need to rate more movies than they actually like (or know).

Summary Forced uniformity has some negative effects on user behaviour, such as that the total number of selected items, and that the mean ratings tend to decrease. However, in several diversity measurements we do not get a very clear tendency of change. Looking at the results from another angle, if a recommender owner wants to achieve certain results (higher/lower diversity in some sense, higher mean rating which might be associated with user satisfaction), they need to know which recommendation types match which user behaviours. Then, they might either target groups of users with certain types of recommendations (e.g., highest rated movies or most popular ones), or try to change user behaviour some other way to match what they offer as recommendations.

4.3 Yes-Men, Popularists and Trend-Followers

In this subsection, we compare three groups of simulation results in order to find out how various – fundamentally different – movie recommendation approaches result in various diversity values. The first group contains the Yes-men type results (personalised recommendation), the second is a group of Popularists, and the third is results by Trend-followers.

In terms of number of unique movies rated (Figure 4), all three approaches produce the same results. In other words, it does not seem to matter which of the three approaches is used, the movie coverage will be very similar (though we are limited here to the movies rated in the real data).

The variance of ratings is lowest in case of Trend-followers (Figures 5 and 6), which shows that when highest rated movies are offered to users, the ratings will be very consistent. The global variance values of Yes-men tend to be higher than those of Popularists (Figure 5), while Popularists are clearly associated with higher user-based variance levels than Yes-men (Figure 6). So although the Popularist approach results in globally more consistent ratings, on a user level, users get more consistent quality of recommendations in case of a personalised approach. Quality is more consistent despite the greater fluctuation in variance for Yes-men (Figure 6), as variance values still tend to stay below those of Popularists.

With respect to the entropy of rated movies (Figure 7), Yes-men and Trend-followers have the same level of entropy¹, while Popularist movies are more evenly distributed in terms of how many users watch and rate individual movies. From these and other results presented in previous paragraphs it is becoming evident that recommending popular movies results in very different simulation behaviours from the other two approaches.

The difference of Popularists from Yes-men and Trend-followers is further backed up by Cosine diversity values in Figure 8. Selections and ratings of movies by the Popularist approach are considerably more diverse per user, while the other two approaches produce comparable Cosine diversity values. The difference is also shown by the Cosine values being more stable in case of Popularists.

If we take the mean ratings per user (Figure 3), we can see that the ratings of Popularists are the lowest among the three approaches studied in this subsection. Also, mean ratings corresponding to Popularists do not increase over time as with other approaches. The popularity-based approach is clearly different from the other two. As Figure 3 also shows, offering the so far highest rated movies (Trend-followers) results in the highest mean rating values, which is surprising given that we would have expected a personalised approach to produce the highest mean ratings (which we also associate with highest user satisfaction with the recommender). We speculate that either the level of personalisation is not of high enough quality then, or that recommending what has been rated highest so far is actually better than personalisation (of this kind).

Summary. Trend-followers are associated with the highest mean values, and often with the lowest diversity values (which indicate consistent quality of recommendations). Recommending highest rated movies, hence, makes sense, and it seems that the rating scale is just good enough to let new movies also be recommended (so they get a chance of getting high ratings). Also, a strictly popularity-based approach does not perform well according to our simulations, and does not produce consistent ratings: it is highly dependent on the initial set of movies rated, so when new items enter the database, they stand little chance of being recommended to many users. Hence, we think that a popularity-based approach should always be coupled with a promotion-based one (not

¹ Now we only consider the power-law versions of scenarios as i) they are the ‘normal’ cases (see true dataset), and ii) there are great entropy differences between these and uniform versions, which have already been addressed in the previous subsection.

investigated in this paper), so there can be a balance between old, popular movies, and new, potentially popular movies.

A different kind of mixture is investigated in the next section, which studies the gradual transition from a Yes-men approach to a totally random selection of movies.

4.4 From Yes-Men to Random Selection

In this section we investigate how diversities and main ratings change as we introduce noise into personalised recommendations. Whether the source of noise, i.e. randomness, comes from the user rejecting recommendations and rating other movies, or from the recommender introducing random movies (for example, to facilitate exploration of movies), it does not make a difference for our purpose. One of the questions we aim to answer is whether we can model the true dataset as a combination of Yes-men and noise. If this can be done, further research into recommenders and user behaviour could be made easier. For our investigation, we consider five scenarios in which randomness gradually ranges from 100% to 0%.

As Figure 3 shows, as we introduce more randomness into the personalised recommender approach, average ratings will gradually decrease. This is natural, as noise can not really be considered ‘quality’. What is noticeable though, is that if users take only 25% of the personalised recommendations, we end up with mean ratings that closely match reality. So we might say that there is a lot to improve on the recommender producing the true dataset, but we also need to keep in mind that there are considerable differences between reality and our simulations (different recommender engine, various user behaviours in reality, etc.).

In terms of both types of rating variances measured (Figures 5 and 6), there do not seem to be a mixture that would model reality (even if reality is corrected by intra-user variability mentioned already). Also, there is no monotonic function that would show a clear transition from Yes-men to Randomisers. As the figures show, a 50% noise level (an even mixture) seems to give the highest variance in ratings. This is due to the fact that the mean ratings of Yes-men and Randomisers are relatively far apart, and low in variance, and, as there are both low and high ratings in mixture scenarios, the variance of mixtures should indeed be higher, reaching their peak at around a 50-50% mixture.

Regarding entropy (Figure 7), it seems possible to match reality relatively closely, although this is not achieved the same way as for the mean (i.e. with 25% Yes-Randomisers). The mixture that models reality would roughly be a 65% Yes-Randomisers scenario. Comparing entropy values of various levels of randomness, by definition, a totally random approach results in the highest entropy, while less noise results in movies less evenly distributed.

As for entropy, Cosine diversity values (Figure 8) also show the highest possible value for Randomisers, and that a mixture can model reality quite closely. However, now it is again the 25% Yes-Randomisers that fit the true data well (just as for the mean ratings). With respect to the overall values of Cosine diversity, total randomness results in users’ selections and ratings being the most

different from one another, and less noise makes people select and rate movies more similarly.

Summary. Introducing noise to personalised movie recommendations can model reality, although not consistently across diversity measures. This shows that, naturally, reality is more complicated than a single recommendation algorithm plus a somewhat random user behaviour. However, for specific purposes, we believe that such mixtures can be useful for future studies of recommenders.

5 Conclusions

In this paper, we have investigated how diversity changes under different scenarios determined by 1) the type of information a recommender offers, and 2) the behaviour of users of the recommender. We have considered four main types of recommender-user interaction styles: Yes-men, Trend-followers, Popularists and Randomisers. Depending on the number of items that users rate we have identified two variants of each type (except Randomisers): ‘power law’, where users rate the same number of movies as in Netflix data, and ‘uniform’, where each user rates the same number of movies. Additionally, we have analysed mixtures of users of Yes-men and Randomiser type, using proportions of 25%, 50% and 75%.

For each of the 10 scenarios we have performed extensive simulations of 60 rounds (covering the period of 5 years) collecting, for each round, the following diversity measures:

- the mean and the variance of ratings,
- the mean of user-based variances of ratings,
- the number of unique movies rated in a given round,
- the normalised Shannon entropy of rated movies,
- Cosine diversity of users’ ratings.

The analysis of results demonstrates a big variability of the simulated scenarios. In particular, it turned out that for non-uniform scenarios:

- the mean rating is highest for Trend-followers, then Yes-men, followed by Popularists and Randomisers,
- the diversity of rated movies (entropy) is biggest for Randomisers, then Popularists, followed by both Yes-men and Trend-followers,
- the diversity of user ratings (Cosine diversity) is biggest for Randomisers, then Popularists, followed by Yes-men and Trend-followers.

None of the simulated scenario mimics the behaviour of Netflix clients. This may have been caused by the fact, that these clients do not form a uniform group that falls under one of our scenarios. Finding mixtures of scenarios that reflect real-life user behaviour is beyond the scope of this paper, but is certainly worth further investigation.

Our results have some practical consequences for designers of recommender systems. We have found that imposing the uniform selection mechanism has some negative effects. For example, the average ratings of Yes-men and Trend-followers significantly drop, suggesting lower user satisfaction. Additionally, to achieve certain targets (e.g., high ratings which could be attributed to the high performance of the recommender system), user groups that behave according to certain patterns need to be identified. Once it is known how groups of users behave, they can be offered appropriate kinds of recommendations (e.g., personalised lists of items, most popular items, or certain mixtures) in order to achieve the previously identified targets.

References

1. Adomavicius, G., Tuzhilin, A.: Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge and Data Engineering* 17(6), 734–749 (2005)
2. Ahn, H.J.: Evaluating customer aid functions of online stores with agent-based models of customer behavior and evolution strategy. *Information Sciences* 180, 1555–1570 (2010)
3. Alam, S., Dobbie, G., Riddle, P.: Exploiting swarm behaviour of simple agents for clustering web users' session data. In: *Data Mining and Multiagent Integration*, ch. 4. Springer, Heidelberg (2009)
4. Andrain, C.F.: Comparative political systems: policy performance and social change. M.E. Sharpe (1994)
5. Bennett, J., Lanning, S.: The netflix prize. In: *KDD Cup and Workshop in Conjunction with KDD*, pp. 3–6 (2007)
6. Cosley, D., Lam, S.K., Albert, I., Konstan, J.A., Riedl, J.: Is seeing believing? how recommender interfaces affect users' opinions. In: Cockton, G., Korhonen, P. (eds.) *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (2003)
7. Deerwester, S.C., Dumais, S.T., Landauer, T.K., Furnas, G.W., Harshman, R.A.: Indexing by latent semantic analysis. *Journal of the American Society of Information Science* 41(6), 391–407 (1990), citesear.ist.psu.edu/deerwester90indexing.html
8. Dehuri, S., Cho, S.B., Ghosh, A.: Wasp: A multi-agent system for multiple recommendations problem. In: *Proc. of 4th International Conference on Next Generation Web Services Practices*. IEEE (2008)
9. Derbyshire, J.D., Derbyshire, I.: *Political Systems of the World*. Palgrave Macmillan (1996)
10. Dias, M.B., Locher, D., Li, M., El-Deredy, W., Lisboa, P.J.G.: The value of personalised recommender systems to e-business: A case study. In: Pu, P., Bridge, D.G., Mobasher, B., Ricci, F. (eds.) *Proceedings of the 2008 ACM Conference on Recommender Systems*, pp. 291–294 (2008)
11. Fleder, D., Hosanagar, K.: Blockbuster culture's next rise or fall: The impact of recommender systems on and sales diversity. *Management Science* 55(5), 697–712 (2009)
12. Forsyth, D.: *Group Dynamics*, 3rd edn. Wadsworth Publishing (1998)
13. Funk, S.: Netflix Update: Try This at Home (2006), <http://sifter.org/~simon/journal/20061211.html>

14. Garfinkel, R., Gopal, R., Pathak, B., Venkatesan, R., Yin, F.: Empirical analysis of the business value of recommender systems. Tech. Rep. Working Paper 958770, Social Science Research Network (2006)
15. Gursel, A., Sen, S.: Improving search in social networks by agent based mining. In: Kitano, H. (ed.) Proceedings of the International Joint Conference on Artificial Intelligence (2009)
16. Gursel, A., Sen, S.: Producing timely recommendations from social networks through targeted search. In: Decker, K., Sichman, J., Sierra, C., Castelfranchi, C. (eds.) Proc. of 8th Int. Conf. on Autonomous Agents and Multiagent Systems, AAMAS 2009 (2009)
17. Ishigaki, T., Motomura, Y.: Toward computational modeling of the consumer based on a large-scale dataset observed in a real service. In: Proc. of International Conference of Soft Computing and Pattern Recognition. IEEE Computer Society (2009)
18. Koren, Y., Bell, R., Volinsky, C.: Matrix factorization techniques for recommender systems. Computer 42, 30–37 (2009)
19. Lee, W.P., Yang, T.H.: Personalizing information appliances: a multi-agent framework for tv programme recommendations. Expert Systems with Applications 25, 331–341 (2003)
20. Ono, C., Kurokawa, M., Motomura, Y., Asoh, H.: A context-Aware Movie Preference Model using a Bayesian Network for Recommendation and Promotion. In: Conati, C., McCoy, K., Palioras, G. (eds.) UM 2007. LNCS (LNAI), vol. 4511, pp. 247–257. Springer, Heidelberg (2007)
21. Said, L.B., Drogoul, A., Bouron, T.: Multi-agent based simulation of consumer behaviour: Towards a new marketing approach. In: International Congress on Modelling and Simulation Proceedings (2001)
22. Said, L.B., Bouron, T.: Multi-agent simulation of virtual consumer populations in a competitive market. In: Proceedings of the Seventh Scandinavian Conference on Artificial Intelligence. IOS Press (2001)
23. Said, L.B., Bouron, T., Drogoul, A.: Agent-based interaction analysis of consumer behavior. In: Proceedings of First International Joint Conference on Autonomous Agents and Multi-Agent Systems. ACM (2002)
24. Sarwar, B., Karypis, G., Konstan, J., Reidl, J.: Item-based collaborative filtering recommendation algorithms. In: Proceedings of the 10th International Conference on World Wide Web, WWW 2001, pp. 285–295. ACM, New York (2001), <http://doi.acm.org/10.1145/371920.372071>
25. Sen, S.: Finding Useful Items and Links in Social and Agent Networks. In: Cao, L., Bazzan, A.L.C., Gorodetsky, V., Mitkas, P.A., Weiss, G., Yu, P.S. (eds.) ADMI 2010. LNCS, vol. 5980, pp. 3–3. Springer, Heidelberg (2010)
26. Sen, S., Saha, S., Airiau, S., Candale, T., Banerjee, D., Chakraborty, D., Mukherjee, P., Gursel, A.: Robust Agent Communities. In: Gorodetsky, V., Zhang, C., Skormin, V.A., Cao, L. (eds.) AIS-ADM 2007. LNCS (LNAI), vol. 4476, pp. 28–45. Springer, Heidelberg (2007)
27. Su, X., Khoshgoftaar, T.M.: A survey of collaborative filtering techniques. Advances in Artificial Intelligence, 1–19 (2009), <http://dx.doi.org/10.1155/2009/421425>
28. Takács, G., Pilászy, I., Németh, B., Tikk, D.: Scalable Collaborative Filtering Approaches for Large Recommender Systems. J. Mach. Learn. Res. 10, 623–656 (2009)
29. Weng, L.T., Xu, Y., Li, Y., Nayak, R.: Towards information enrichment through recommendation sharing. In: Cao, L. (ed.) Data Mining and Multiagent Integration. Springer, Heidelberg (2009)

Opinion Formation in the Social Web: Agent-Based Simulations of Opinion Convergence and Divergence

Pawel Sobkowicz¹, Michael Kaschesky¹, and Guillaume Bouchard²

¹ Bern University of Applied Sciences, Morgartenstrasse 2a, Bern, Switzerland
{pawelsobko,michael.kaschesky}@gmail.com

² Xerox Research Center Europe, 6 chemin de Maupertuis, Meylan, France
guillaume.bouchard@xrce.xerox.com

Abstract. Recent research on opinion formation in the social web – particularly blogs, comments, and reviews – investigates opinion dynamics but reaches opposite conclusions whether consensus formation occurs or not. To address this issue, a model of consensus formation is described that takes into account not only factors leading to convergence of opinions, but also those that strengthen their divergence. Nonlinear interplay between these tendencies might lead to interesting results, and decoupling the technical basis of the interactions (e.g. network dynamics) from the human perspective of opinions and sympathies (e.g. social dynamics) is at the core of such an approach. The model presented here combines the features of epidemic diffusion and cascading models of opinions with simulations including presence of large part of society which remains neutral with respect to the issue at question. The presence of such neutral community changes significantly the topology of resulting social network and dynamics of majority opinion acceptance.

Keywords: Opinion Research, Agent-based Simulations, Measurement, Economics, Reliability, Experimentation, Human Factors.

1 Introduction

Study of dynamics of opinion spreading is of important practical value in many diverse domains, from commercial marketing through judging effectiveness of government activities to counterterrorism. The field of study has been recently subject of intensive growth, drawing upon experiences from many disciplines. In addition to traditional sociological studies, tools borrowed from statistical physics, Bayesian statistical methods and agent-based computer modeling have become important. Moreover, the increased role played by the Internet based communication channels allowed the researchers to access massive amounts of data documenting people's activities, expressed sentiments and interactions. This changes the perspective of the research field, enabling the use of datasets obtained from the Internet to improve theoretical models to the point of usability in predicting social behavior.

Recent research on opinion formation in the social web – particularly blogs, comments, and reviews – investigates opinion dynamics but reaches opposite conclusions whether consensus formation occurs or not. Many standard opinion dynamic models postulate a form of averaging of opinions towards a mean value [7,13], others assume that as a result of the interaction between two agents one of them changes his or her opinion by adopting the other opinion [28]. Unfortunately, large part these studies concentrate on mathematical formalisms or Monte Carlo simulations, and not on descriptions of real-life phenomena. However, situations where consensus formation does not occur have also been observed [26]. Indeed, prolonged interactions may even increase the rift between participants. This effect should be studied in more detail, as it possibly suggests modifications of the models of consensus formation in other situations. The need of bringing simulations and models closer to reality has been realized and voiced quite a few times [10,26].

The persistence of differences of opinions exhibited in online political discussions [25] stands in contrast to observations of Wu and Huberman [29], who measured a strong tendency towards moderate views in the course of time for book ratings posted on Amazon.com. However, there are significant differences between book ratings and expression of political views. In the first case the comments are generally favorable and the voiced opinions are not influenced by personal feuds with other commentators. Moreover, the spirit of book review is a positive one, with the official aim of providing useful information for other users. This helpfulness of each of the reviews is measured and displayed, which promotes pro-sociality and ‘good’ behavior. In the case of political disputes it is often the reception in one’s own community that counts, the show of force and verbal bashing of the opponents. The goal of being admired by supporters and hated by opponents promotes very different behavior than in the cooperative activities. For this reason, there is little to be gained by a commentator when placing moderate, well-reasoned posts – neither the popularity nor status is increased. These kinds of behavioral considerations must be taken into account by computer modeling, particularly in opinion formation (for recent review see [5]).

Both history and literature are full of examples of undying feuds, where acts of aggression follow each other, from Shakespearean Verona families to modern political or ethnic strife. Observations of the Internet discussions should therefore be augmented by sociological data on flesh-and-blood conflicts and arguments, and the dynamics of opinion shifts. But even before such studies are accomplished the basic assumptions of the sociophysical modeling of consensus formation should be expanded. This is a very interesting task, because ostensibly we are faced with two incompatible sets of observations:

- Using what each side considers as ‘facts’ and ‘evidence’ to support their viewpoint, participants in political discussions tend to reinforce their opinions, strengthening their resolve with each exchange. In this case, interactions do not lead to opinion averaging or switching.

- Most participants have well defined opinions. These must have formed in some way. Specific opinions on concrete events or people cannot be explained by genetic or cultural coding – they must be reached individually in each case, usually through influence of other people.

So, we are, at the same time, resistant to external opinions and yet build our views under such influences. We suggest the existence of two mechanisms: fast consensus formation within one's own group (including adoption of common, stereotyped views and beliefs); and persistence of differences between groups. An interesting experimental confirmation of such phenomenon has been published by Knobloch-Westerwick and Meng [16], whose findings '*demonstrate that media users generally choose messages that converge with pre-existing views. If they take a look at 'the other side,' they probably do not anticipate being swayed in their views. [...] The observed selective intake may indeed play a large role for increased polarization in the electorate and reduced mutual acceptance of political views.*'

In the current paper, we address the question of how individual characteristics of individual opinion adoption can be incorporated into opinion formation modeling. We describe a model of consensus formation that takes into account not only factors leading to convergence of opinions, but also those that strengthen their divergence. Nonlinear interplay between these tendencies might lead to interesting results, and decoupling the technical basis of the interactions (e.g. network dynamics) from the human perspective of opinions and sympathies (e.g. social dynamics) is at the core of such an approach.

Our work extends the previous approach ([26]) by introducing the ‘silent majority’: large part of society that has initially no formed opinion on the issue in question. Its presence significantly changes both the network properties of communication of opinions, allowing indirect links between people holding opposing opinions and the dynamics of global consensus achievement.

2 Research Background

Recent years have brought significant interest in interdisciplinary studies, combining tools and methods known from physics with social analyses. Computational models of opinion formation often combine results derived from statistical physics with agent based simulations [4]. Within a simplified framework focusing on selected aspects (such as interpersonal communication network, susceptibility to influences, contrariness etc.) it is possible to derive general trends of behavior of large societal groups. However, one of the major problems with opinion modeling by computer simulation is the lack of real-life data. Recent works reiterate the need for a real data by emphasizing evidence over conceptual models and theory and prediction, explanation and guiding data collection in opinion modeling observation. Our task here is a challenge and innovation, using Internet data for large-scale social observations, analyzing its past evolution, and simulating potential future developments.

Opinion modeling based on sociophysics typically focuses on global properties of the modeled system. However, to be able to provide practical insights

and real-life tools for social monitoring and policy formulation, models must include factors absent from simplified viewpoints. An important way of extending such an approach is via agent-based simulations, which allow integration of agent descriptions (characteristics) that are more detailed on the micro-level of individual behavior and therefore enable the combination and cross-fertilization of observations across several levels of social organization, from the individual micro-levels to the aggregated macro-levels of society. Due to the rise of online media and user-generated content, opinion modeling has become very popular in recent years. The recent review by Castellano et al. summarizes many interesting results obtained via theoretical analyses and computer modeling [5]. There are several models of opinion formation (Sznajd-Weron [28], Deffuant [7], voter model, Krause-Hegelsman [13]), and a significant number of research papers based on these approaches. Other approaches have been proposed by Nowak and Latané [21,20] and Kacperski and Holyst [14,15]. Many of these models suffer from three important deficiencies:

- Most of them focus on achieving consensus and fail to account for divergences, such as persistence of minority groups, reluctance to accept policies, influence of individuals and subgroups etc.
- There is a fundamental problem in relating ‘computer time’ (usually measured in simulation steps) with real time for social phenomena, which is important for predicting dynamic real situations.
- Lastly, many studies are abstracted from social reality, only a few attempts to explain concrete events or actions observed in the real world [26].

In addition, several works on propagation models for online information diffusion are based on simple infection models, which address one or more of the following issues:

- Link structure between blogs to track flow of information. Classification-based technique to model opinion propagation (called ‘infection’) [1];
- Macroscopic topic model and microscopic propagation model [12];
- Tracing chain-letter data, propagation trees, small world network [19].

Cascading models are better suited to track and simulate large-scale information cascades, which is at the core of our research. In cascading threshold models, an agent (person) will adopt an opinion if a specified fraction of its neighbors have adopted that opinion. The network is initialized by ‘seeding’ a small number of agents with new information. If it spreads throughout a significant portion of the network, it is referred to as an information cascade, resembling the spread of ideas and fads in real-world complex systems. As said earlier, the foundations were laid out by Thomas C. Schelling [22]. Some recent approaches include:

- Modelling opinion formation as cascades based on efficient inference and learning algorithms [11];
- Cascades for modelling the propagation of recommendations for commercial products based on evolution of the recommendation network over time [17];
- Propagation of rumours and ideas based on a cascading model [18].

Research on the diffusion of information over time often assumes a prior model for information diffusion that is valid on every individual in the network. The goal of the inference technique is to recover the most probable graph that explains how information propagates in blogs, news feeds, emails, etc. There is a direct analogy between information diffusion and disease propagation. The most common approach is to model the population as a network of individuals fitting into one of three categories: Susceptible to the disease (S), currently Infected (I) and Recovered (R). In the case of information diffusion, the following analogy is made:

- S-state is the state of the individual before hearing the news (or discovering the new product),
- I-state corresponds to the state when user is keen to relay information/speak about the product, and
- R-state means that the user does not speak about the news or the product anymore.

When studying such models, researchers found interesting differences between classical disease diffusion and information propagation [12]:

- network structure on the internet is different because it typically has a power-law structure,
- transmission model for information must account for decay in the infection rate
- parameters of the diffusion model (such as information adoption rate) are dependent on individual characteristics, which are less important for disease contagion.

Estimations and probability intervals for parameters in complex systems, such as communication networks, are best calculated with probabilistic inference techniques, sometimes called Bayesian inference although the modeling is not always Bayesian (i.e. assuming a distribution over the parameters). The literature on social or network learning can be distinguished according to two criteria: whether learning is adaptive or myopic, and whether individuals learn from communication of exact signals or from the payoff of others, or simply from observing others' actions. Typically, probabilistic models focus on learning from past actions and communications, while most, but not all, myopic learning models focus on learning only from communications [3,24]. In addition, research in engineering studies related problems, especially motivated by aggregation of information collected by decentralized sensors. These existing models and methods are used for modeling special aspects of social communicated networks.

2.1 Agent-Based Modeling of Opinions

Among advances necessary to overcome existing shortcomings is providing better correspondence between parameters describing computer agents used in simulations and psychological and sociological characteristics of human participants, relevant to specific situations. Such mapping will account for, among others:

- correlations between opinions held on various subjects (multidimensional opinion description);
- dependence of individual opinion changes on pre-existing emotional states and on emotional formulation of the messages and policies;
- variances of attention and of communication capacities of individuals;
- tendencies to preserve one's own opinions and selective attention to views differing from them. Therefore, the objectives of the proposed research approach are as follows:
- opinion diffusion model, based on advanced agent-based modelling that could be used in variety of policy situations, especially in the context of analysis of data obtainable from online media;
- analyze which factors describing individual and social situation are relevant for opinion change modelling, by comparing simulations with observations and analyses;
- baseline computational algorithms for simulations of opinion formation and diffusion;
- statistical learning algorithms for effective opinion simulations with predictive capabilities.

There is a rich literature on agent-based modeling of opinions, but the main interest is often in finding conditions which lead to a society reaching unanimous opinion, either via internal interactions or due to external influences. For this reason the field is often called 'consensus formation'. On the other hand, the real world examples of societies where full consensus is present are quite rare. In fact, where important issues are at stake, we observe extremely stable situations of opinion splits, ranging from relative parity (for example in political setup of many countries) to configurations where small minority persists despite strong social pressures (for example extremist views). Our goal is to study opinion formation taking into account some processes that are usually omitted, which, in our opinion play a crucial role in real social situations and increase the stability of split opinion configurations. We study changes of opinion in conjunction with other processes, such as segregation.

Current 'standard' approaches to opinion formation modeling focus on changes of individual opinions due to encounters between agents. It is surprising that the tendency of voluntary social separation due to attitude or opinion similarities and differences, well known and recorded even in folklore ('birds of a feather group together') has been neglected in opinion simulations so far. Instead of the choice of withstanding the arguments of an opponent or surrendering to them, most of us have the choice of severing the relationships with them. This results in relatively closed sub-societies, with little interaction between them, especially when the issues under study are of crucial value. A classical example of social modeling study invoking the separation tendencies is the famous Schelling model of social segregation [23]. It should be noted that the increased reliance of society members and organizations on Internet channels of communication provide unprecedented levels of flexibility of association. People are free to join groups of interest and express their views. The perceived anonymity and lack of face-to-face contact encourages expressions of extreme opinions. This leads to situation

in which social networks, in addition to those built on similarity of opinions and interests, may also be built on differences and even hate [25]. Thus, social segregation, as seen through the Web media is more complex than traditional models allow. Figure 1 below shows an example of changes in simulated opinion distribution and social network topology leading to opinion diffusion and network rearrangement. While the predominant opinion reaches an increasing number of previously undecided participants, minorities may remain untouched, separated from the main part of society. The left of Figure 1 shows the initial social state with a large majority of undecided participants without a specified opinion on the issue (grey), and smaller groups of proponents and opponents concerning the issue (respectively light grey and black). The social links enabling information flow and opinion influences exist between all groups. The right panel of Figure 1 shows the final state of simulated opinion diffusion with most of the participants having adopted the proponents' position. However, small minorities persist at the outskirts of society, mainly because they have severed and cut most of the information links with the majority [26]. Their existence and vitality can be monitored for example, via internet media activity, especially when filtering for high negative emotions. This enables policymakers to adjust implementation to better integrate dissenters and communication to better target expected benefits and consequences, by monitoring the effects of activities of individual persons and leaders [27].

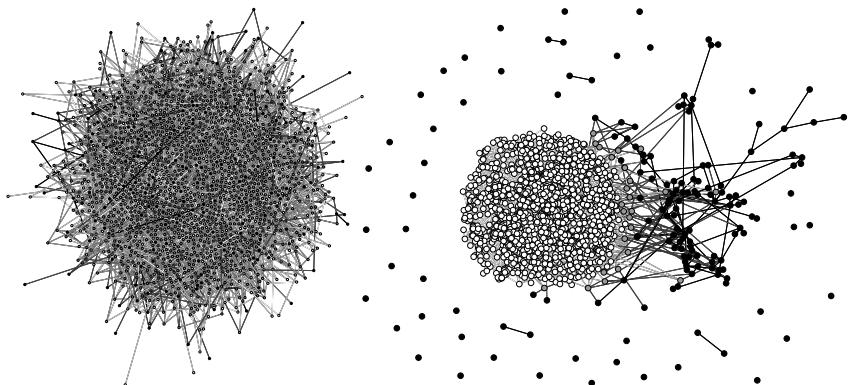


Fig. 1. Initial state with large majority undecided (left) and final state of simulated opinion diffusion with majority adopting a mainstream opinion (right)

3 Simulation Model with Predictive Capabilities

The simulations presented here allow computer agents to cut the social links with those they disagree with and form new links with agents sharing the same opinion. This changes the network structure. To take into account the fact that

in real societies some links cannot be broken (e.g. family or work relationships) we have simulated situations where certain percentage of the links remain static, while the rest are free, allowing changes in topology of the social network. The first novel aspect of the simulations is direct inclusion of agents with no preferred opinion (neutral agents). This allows significant change from models where only committed agents are present, changing both the social network and opinion change dynamics. Appropriate real life examples of application of the simulation include political preferences and highly controversial opinions on topics such as abortion or evolution. Within the model, the strength of influence between agents decreases with their social separation, reflecting the fact that our opinions are swayed less by remote acquaintances or strangers than by the closest associates.

Secondly, the opinion of a given agent may be changed in reaction to perceived cumulative social opinion of others, corresponding to a properly averaged ‘peer pressure’ rather than the individual encounters. Many of the classical models have stressed the importance of such individual contacts in opinion changes of agents, but in our belief the constant background of perceived opinions, resulting from numerous meetings (not necessarily related to the issue in question) and information on opinions held by other society members is more relevant. In a way, this can be described as each agent continuously measuring and responding to the ‘discomfort’ due to difference between its opinion and properly averaged opinions of other agents. In addition to this intrinsic pressure it is possible to simulate, via simple parameterized factor the external influence, aimed to describe propagandist efforts. The main features of the model can be summarized as follows:

- A set of N interacting agents forms a society. Agents are connected to form a scale-free Albert-Barabási network. Such networks are typical for many social environments, exhibiting fat-tailed, power-law distribution of social connectivity. One of the most important features is presence of highly connected hubs, which significantly influence information and opinion flow within society. Such networks were found in many empirical studies e.g. [2,9,8,6].
- Each agent i has, at a given time, his ‘opinion’ $\sigma(i)$ (+1, -1 or 0). The addition of the class of neutrals forms the crucial enhancement of the model.
- The network is assumed to have a dynamic character. Depending on simulation parameters, certain ratio of links is assumed to be static, corresponding to family or workplace connections, which are difficult to break even if there is a difference in opinions of the connected agents. The rest of the links are free, and may be cut off when the agents have conflicting (+1/ -1) opinions. To preserve the general network properties the process of destruction and creation of new links is performed in a way that keeps the number of connections constant. The creation of replacement links may happen within strict similarity scenario: new links form only between agents with the same opinion (+1/ +1 and -1/ -1) or in promiscuous scenario, when links may form between agents that are not in conflict. Thus the neutral agents act as possible intermediaries between extreme opinion holders. Until the neutral

part of the society is convinced by the majority, there are still possibilities of indirect communication between the opposing factions, in contrast to the previous model.

- Influence of agents on other agents is governed by their respective opinions as well as on their social distance. We measure this distance simply via the number of links separating the agents in the network. This is described in detail in later part of the paper.
- Opinion of an agent is changed due to combined influence of all other agents. To describe the natural human tendency of resisting such external pressures, we have added in our simulations additional factor so, describing the influence of an agent on itself, always opposing the direction of external pressures.

As the starting distribution of all the above mentioned parameters is random, it has little connection with real societies, with their correlated structures and patterns of opinions. To keep the model closer to reality, we start with a relatively small number of time steps (< 5000 , corresponding to 2.5 time steps per agent), during which the agents adjust the network links but are not allowed to change their opinions. This pre-separates (if there are free links) the network in accordance with initial random distribution of opinions. The reason for such initial phase is to avoid spurious results that arise from fully random distributions. For simulations where all the links are assumed static, this phase simply does not change anything. Later simulations involve much larger number of steps (> 200 steps per agent). In each of them first a randomly chosen agent is allowed to change opinion and then the same agent may adjust its network neighborhood by cutting a link with an opponent and adding one with a friendly agent. These two key processes of the simulation are described below.

3.1 Key Process 1

For a randomly chosen ‘initializing’ agent i we calculate the network separation for all other agents and separate them into the four spheres S_1 to S_4 . These are composed of, respectively, the direct neighbors, the second neighbors, third neighbors and the rest of the society. The strength of the influence of an agent j on i depends on the sphere that j belongs to. For each agent j we calculate also a modification factor which reflects our human tendency to give more value to opinions of those who agree with us and to give less credibility to contrary opinions. Thus $f_\sigma(i, j) = 2$ if $\sigma(i) = \sigma(j)$ and $f_\sigma(i, j) = 1$ otherwise. In Figure 1 below, S_1 corresponds to nearest neighbors, S_2 to next nearest neighbors etc. S_4 holds the large majority of the ‘rest of society’. Influences of agents within each sphere are averaged and normalized, and the relative importance of the spheres is described by geometrically decreasing series with ratio d . In this work we have used $d = 0.3$, which leads to 70.5 percent of the weight of the influence originating from the immediate sphere of neighbors S_1 .

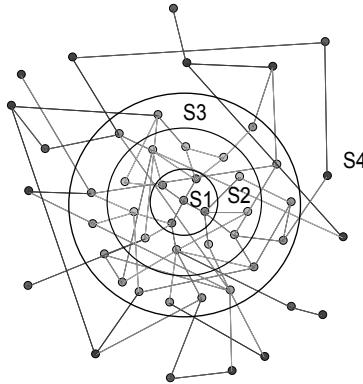


Fig. 2. Schematic division of social network into four spheres, following the distance from the chosen agent i

For each of the spheres S_K we calculate the averaged, normalized influence over agent i as follows:

$$I_K = \frac{\sum_{j \in S_K} f_\sigma(i, j)s(j)\sigma(j)}{\sum_{j \in S_K} f_\sigma(i, j)s(j)}, \quad (1)$$

where $s(j)$ measures individual strength of agent j . The denominator acts to normalize I_K , so that if all agents j within S_K are of the same opinion, then their effect has the maximum absolute value $\|I_K\| = 1$. Allowing selected agents to have $s(j)$ much larger than the rest provides grounds for modeling leaders.

The overall influence of the society on agent i is then calculated as a weighted sum of I_K . To reflect the observation that close acquaintances have much more influence than strangers we have assigned geometrically decreasing weights, for the spheres of increasing distance. The total influence of the society on agent i is as follows:

$$T(i) = \sum_{K=1}^4 I_K d^{(K-1)} \frac{1-d}{1-d^4}, \quad (2)$$

where $d < 1$ and the last factor serves to provide normalization in addition to social pressure. Choice of d determines how important are close acquaintances in influencing agent's opinion, for example choice of $d = 0.3$ leads to 70.5 percent of influence coming from the nearest neighbors, and only about 2 percent from sphere S_4 , containing the large number of agents forming the 'rest of society'. It is possible to include in $T(i)$ and additional element describing external influence, such as media coverage, government propaganda etc. applied independently of agent-to-agent contacts. This is done by adding a constant term h , the same for all agents. The change of agent's opinion is determined by comparing the external influences of the society and propaganda with self influence factor s_o , which determines the general susceptibility of agents in simulation to such influences.

It is one of the main simulation parameters, upon which the stability of social mix depends to a large degree.

3.2 Key Process 2

The second step in simulations is the modification of network connections. To maintain the number of links constant we have alternated cutting and adding links. The first of these processes is very simple. Keeping the previously chosen agent i , we pick another, ‘target’ agent t randomly from the set of nearest neighbors that disagree with i . The link between i and t is then cut off. Obviously, if all i ’s neighbors agree with it, then we move to next simulation round.

Creating new links is more complex. The target agent t is not chosen totally at random among the spheres but rather with probability decreasing on social distance between i and t , so that probability of forming a new link with agents from further spheres decreases geometrically. Again, we chose this form for conceptual simplicity, with the aim of getting a simple equivalent to the observations that we meet more often with our close social neighbors. The newly formed link between i and t is treated as a free one. The resulting network preserves, with only minor deviations, the general degree distribution characteristic for Albert-Barabási scale free networks.

When there are no neutral agents in the simulations the process leads very quickly to separation of subnetworks of +1 and -1 opinion holders if the free links are dominant. As Figure 3 below shows, there are only a few links between agents supporting different opinions, dividing the society into separate parts, with weak chances of achieving consensus of opinions, as each agent is surrounded socially by proponents of his own view. While the average number of links per agent remains unchanged and only minor changes in degree distribution are observed, other relevant network properties change radically, for example the shortest path distribution and clustering coefficient. This is especially important when all the links are free, because +1 and -1 opinion holders segregate to separate communities. If we calculate the network properties of these communities separately, then in the case of large majority we observe that there is similarity to the original configuration in such characteristics as clustering coefficient. On the other hand, the minority has much larger fraction of disconnected agents, so that there is significant difference in related network characteristics. In comparison, Figure 4 demonstrates that when as little as a third of links between the separate communities are held static, there are more links between supporters of opposite views but social division remains. In previous models the ‘static’ links (representing family or workplace relationships, which can not be cut off by the user simply because of difference of opinions) had to be ‘hard wired’ into simulations and do not change with the time evolution. With the introduction of neutral agents into the simulations the connections between the opposing factions, mediated by neutral agents are dynamical in nature and significant changes in the network evolution are observed. Figure 5 shows that neutral agents, marked in gray, serve as a link between the opposing opinion holders, shortening the average communication paths, even though there are no

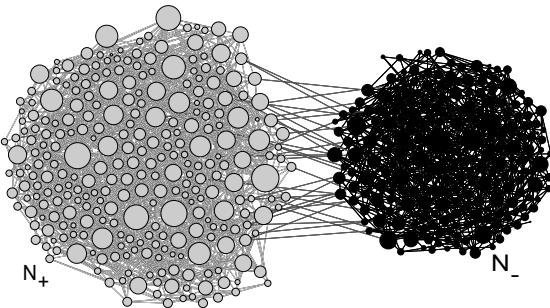


Fig. 3. Social network resulting from simulation where no neutral agents are present and all links are free

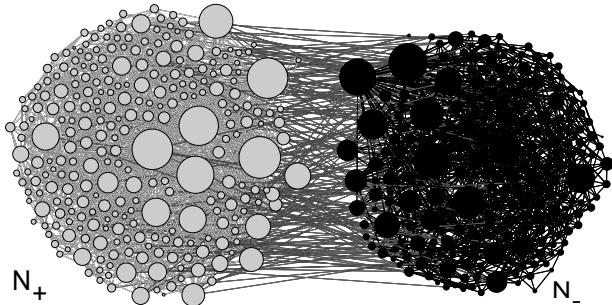


Fig. 4. Social network resulting from simulation where no neutral agents are present and 1/3 of the links are static

direct links between the two communities. In short, presence of neutrals acts as social glue between the committed groups, even when all links are free.

An example of time evolution of opinions in society is shown in Fig. 6. Starting from large population of neutrals ($N_0 = 1200$), 600 N_+ and 200 N_- agents the system where all links are free evolves in simulation time t initially following exponential decay formula $N = N_S - N_1 \exp(-t/T_1)$. This may continue for varying amount of time and sometimes it is the only observed process. For certain values of simulation parameters, however, a second process switches on, described by logistic function $N_2 / (1 + \exp(-\frac{t-T_0}{T_2}))$. The characteristic time of the second process, T_0 is distributed roughly following exponential decay, shown in Fig. 7. There are large differences between simulations run for the same set of parameters, due to specific realizations of the initial network configuration. We find these differences encouraging, as they correspond to high dependence on individual events found in real social phenomena.

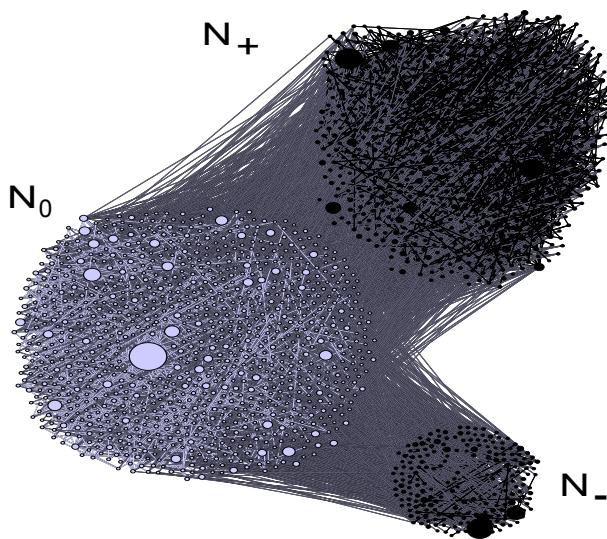


Fig. 5. Social network resulting from simulation where neutral agents are present

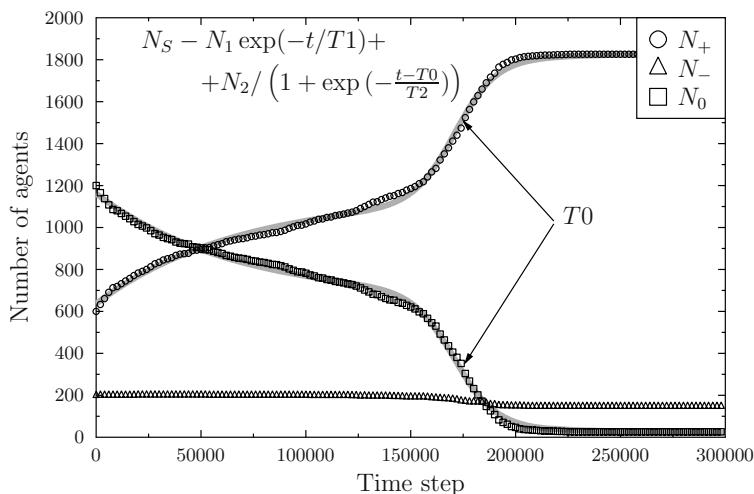


Fig. 6. Example of time evolution of support for +1, -1 and neutral opinions, showing two quasi-independent time regimes, initial exponential decay and logistic massive switch of neutral agents to majority opinion

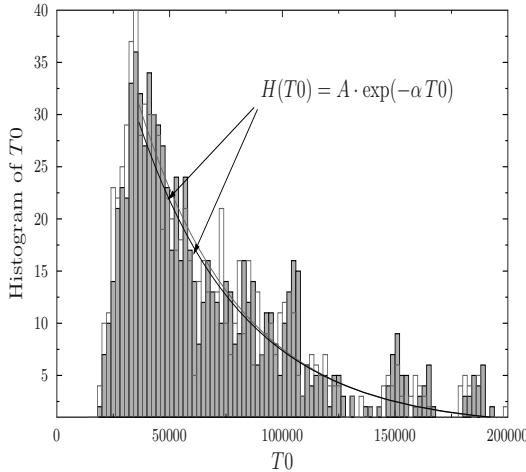


Fig. 7. Histogram of the characteristic time T_0 of the logistic part of the opinion dynamics evolution for the system shown in Fig. 6. Two independent histograms were extracted from growth of the number of majority opinion holders N_+ and from decay of the number of neutrals N_0 , both giving very close results.

4 Conclusion

In comparison to situations where only committed +1 and -1 agents are present [27], the range of situations where a mix of opinions persists within a society is extended when neutral agents are initially present. Depending on the s_o values and the ratio of free links within the network, we observed interesting regimes of opinion dynamics. For large enough s_o , the initial social configuration remains stable, with only a few conversions. At the other extreme, for small s_o , neutrals are quickly converted to majority opinion, but the time gained allows the opposing minority to separate itself from the rest of society. Interesting behavior may be observed for intermediate values where opinion dynamics shows two separate processes. In a first phase, described by the exponential decay, some neutral agents are converted by the majority while the number of minority agents remains almost constant. At some stage a second process may appear, described by logistic function centered at T_0 . During the second phase almost all neutrals adopt the majority opinion. The tempo of the change is relatively constant between various runs of simulations, but the onset of this process may vary over very wide range of values, so that some simulations remain in a three state society (such as the one depicted in Figure 5) even after many hundreds of thousands of time steps. It is the presence of the neutral group that allows persistence of the mixed society (where no consensus has been reached). Simulations indicate that the transition times (spurred by conversion of the neutrals to majority) and eventual separation of majority and minority into almost unconnected networks happens rapidly, but the threshold time T_0 have fat tailed distribution.

The capacity to provide nontrivial behavior by the simulation model is an indication that despite model simplicity, there are enough features to suggest its usability for practical applications of understanding social behavior. Of course there are still many issues related to precise mapping of simulation environment to real life situations, but it has to be remembered that the ultimate goal of computer simulations is not in models *per se*, but in enabling to ask the right questions, improve observations of social phenomena and finally to provide predictions for evolution of specific social cases [14]. For these reasons the current model still needs improvements based on such mapping of simulation parameters to real life.

References

1. Adar, E., Adamic, L.: Tracking information epidemics in blogspace. In: Proceedings of the 2005 IEEE/WIC/ACM International Conference on Web Intelligence, pp. 207–214 (2005)
2. Albert, R., Barabási, A.L.: Statistical Mechanics of Complex Networks. *Review of Modern Physics* 74, 67–97 (2002)
3. Bikhchandani, S., Hirshleifer, D., Welch, I.: Learning from the behavior of others: Conformity, fads, and informational cascades. *The Journal of Economic Perspectives* 12(3), 151–170 (1998)
4. Cao, L., Gorodetsky, V., Mitkas, P.A.: Agent Mining: The Synergy of Agents and Data Mining. *IEEE Intelligent Systems* 24(3), 64–72 (2009)
5. Castellano, C., Fortunato, S., Loreto, V.: Statistical physics of social dynamics. *Rev. Mod. Phys.* 81, 591–646 (2009)
6. Chmiel, A., Kowalska, K., Holyst, J.: Scaling of human behavior during portal browsing. *Physical Review E* 80(6), 66122 (2009)
7. Deffuant, G., Neau, D., Amblard, F., Weisbuch, G.: Mixing beliefs among interacting agents. *Advances in Complex Systems* 3, 87–98 (2000)
8. Ding, F., Liu, Y.: Modeling opinion interactions in a BBS community. *The European Physical Journal B* 78(2), 245–252 (2010)
9. Dorogovtsev, S., Mendes, J.: Evolution of networks. *Advances in Physics* 51, 1079–1087 (2002)
10. Epstein, J.M.: Why model? *Journal of Artificial Societies and Social Simulation* 11(4), 12 (2008)
11. Gomez Rodriguez, M., Leskovec, J., Krause, A.: Inferring networks of diffusion and influence. In: Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1019–1028. ACM (2010)
12. Gruhl, D., Guha, R., Liben-Nowell, D., Tomkins, A.: Information diffusion through blogspace. In: Proceedings of the 13th International Conference on World Wide Web, pp. 491–501 (2004)
13. Hegselmann, R., Krause, U.: Opinion dynamics and bounded confidence models, analysis, and simulation. *Journal of Artificial Societies and Social Simulation (JASSS)* 5(3) (2002)
14. Kacperski, K., Holyst, J.: Opinion formation model with strong leader and external impact: a mean field approach. *Physica A* 269, 511–526 (1999)
15. Kacperski, K., Holyst, J.: Phase transitions as a persistent feature of groups with leaders in models of opinion formation. *Physica A* 287, 631–643 (2000)

16. Knobloch-Westerwick, S., Meng, J.: Looking the Other Way: Selective Exposure to Attitude-Consistent and Counterattitudinal Political Information. *Communication Research* 36(3), 426–448 (2009)
17. Leskovec, J., Adamic, L., Huberman, B.: The dynamics of viral marketing. *ACM Transactions on the Web (TWEB)* 1(1) (2007)
18. Leskovec, J., McGlohon, M., Faloutsos, C., Glance, N., Hurst, M.: Patterns of cascading behavior in large blog graphs. In: *Society of Industrial and Applied Mathematics-Data Mining (SDM 2007)*, Minneapolis, Minn. (2007)
19. Liben-Nowell, D., Kleinberg, J.: Tracing information flow on a global scale using Internet chain-letter data. *Proceedings of the National Academy of Sciences* 105(12), 4633 (2008)
20. Nowak, A., Lewenstein, M.: Modeling Social Change with Cellular Automata. In: Hegselmann, R., Mueller, U., Troitzsch, K.G. (eds.) *Modelling and Simulation in the Social Sciences From A Philosophy of Science Point of View*, pp. 249–285. Kluwer, Dordrecht (1996)
21. Nowak, A., Szamrej, J., Latané, B.: From Private Attitude to Public Opinion: A Dynamic Theory of Social Impact. *Psychological Review* 97(3), 362–376 (1990)
22. Schelling, T.: Hockey helmets, concealed weapons, and daylight saving: A study of binary choices with externalities. *Journal of Conflict Resolution* 17(3), 381–428 (1973)
23. Schelling, T.: Dynamic models of segregation. *Journal of Mathematical Sociology* 1, 143–186 (1971)
24. Smith, L., Sørensen, P.: Pathological outcomes of observational learning. *Econometrica* 68(2), 371–398 (2000)
25. Sobkowicz, P., Sobkowicz, A.: Dynamics of hate based Internet user networks. *The European Physical Journal B* 73(4), 633–643 (2010)
26. Sobkowicz, P.: Modelling opinion formation with physics tools: call for closer link with reality. *Journal of Artificial Societies and Social Simulation* 12(1), 11 (2009)
27. Sobkowicz, P.: Studies of opinion stability for small dynamic networks with opportunistic agents. *International Journal of Modern Physics C (IJMPC)* 20(10), 1645–1662 (2009)
28. Sznajd-Weron, K., Sznajd, J.: Opinion Evolution in Closed Community. *Int. J. Mod. Phys. C* 11, 1157–1166 (2000)
29. Wu, F., Huberman, B.A.: How Public Opinion Forms. In: Papadimitriou, C., Zhang, S. (eds.) *WINE 2008. LNCS*, vol. 5385, pp. 334–341. Springer, Heidelberg (2008)

A Data-Driven Approach for Resource Gathering in Real-Time Strategy Games

Dion Christensen, Henrik Ossipoff Hansen, Jorge Pablo Cordero Hernandez,
Lasse Juul-Jensen, Kasper Kastaniegaard, and Yifeng Zeng

Department of Computer Science, Aalborg University, Denmark
`kogle,ossipoff,jorgecordero,ljuul,kkkas,yfzeng}@cs.aau.dk`

Abstract. In real-time strategy games, resource gathering is a crucial part of constructing an army and becoming victorious. In this paper we present an algorithm for resource gathering and show how accumulated game data can be used to approximate travel times in a real-time strategy game. The algorithm builds upon a queueing system for resource collecting agents and optimises resource gathering by utilising travel times of agents in the game world. We implement the algorithm in the testbed of StarCraft: Brood War and compare it with the built-in method for resource gathering in this game. Experimental results show a gain in the amount of resources gathered when the algorithm is compared to the built-in method. In addition, the results demonstrate better predictability when our approach is used to gather resources for this particular game.

Keywords: Resource gathering, data-driven, real-time strategy game.

1 Introduction

In real-time strategy (RTS) games, players engage each other in real-time battles for map domination. This is done by collecting resources, building an army and controlling the units to attack the opponent [3]. A typical RTS game requires players to initially focus on resource gathering, in order to provide an economic foundation for their armies.

Resource management in an RTS game is the act of collecting resources and converting them into new units. It is an important part of becoming the victor of an RTS game since resource management directly affects the ability to create more units. Resource gathering is the concrete act of having agents moving to a resource site, spending time gathering resources from the site and returning to the resource deposit. Intuitively, resource gathering must play an important role in RTS games since resources are needed both for offensive and defensive units. As these units can be destroyed in the course of the game, a steady income of resources is needed to replace the units lost in battles. We did an analysis of 200 StarCraft replays gathered from iCCup¹ that suggests a correlation between

¹ <http://www.iccup.com/>

spending resources and winning the game. Fig. 1 shows the amount of minerals spent by the winners along with their raze score—a score indicating the number of buildings a player has destroyed. The figure shows a tendency to an increased amount of minerals for higher raze scores. This suggests that it is important to obtain a high amount of resources in order to perform well in RTS games.

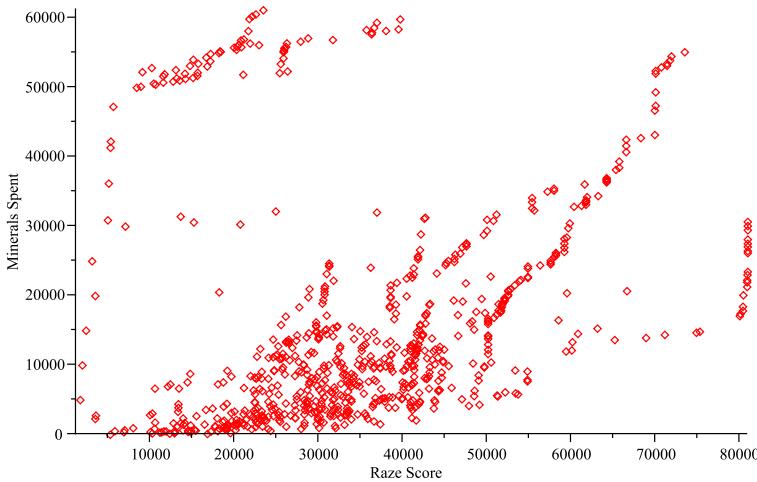


Fig. 1. The amount of minerals gathered compared to the raze score. Data is gathered from the winning players every 120 frames from a total of 200 replays.

Unfortunately, topics on resource management (e.g. resource gathering) have not been well explored in the literature. Previous work by *Chen et al.* [2] neglects the importance of the time required for agents to travel in the game world. *Wintermute et al.* [6] are concerned with strategies for preventing agents from colliding, while gathering resources. We take a step further to uncover the reason for the difficulty of designing a realistic approach for resource gathering. Essentially, it is due to a large amount of uncertainty in RTS games [4]. Efficient resource gathering algorithms require the computation of travel time between a resource site and a resource deposit. However, the details of the underlying pathfinding method may be inaccessible or prone to slight variance due to the nondeterministic nature of some pathfinding algorithms.

In this paper we take a data-driven approach [1] to designing a resource gathering algorithm using the RTS game, StarCraft: Brood War as the test bed. We exploit accumulated gameplay data to compute travel times of agents for use in the algorithm. The use of gameplay data has already made an impact on the strategy prediction in RTS games [5] by the use of replays, made possible by an increasing amount of available replay data on the internet. An example

of this is iCCup² that supports data for StarCraft, Warcraft III and DotA. In order to control the resource gathering actions of agents, we present a queueing algorithm for distributing agents to individual resource sites in the game world and evaluate the performance of this, by comparing it with the built-in resource gathering algorithm of StarCraft: Brood War.

The rest of the paper is organised as follows. Section 2 introduces the algorithm for resource gathering in RTS games. Section 3 shows the algorithm performance in the experiments. Section 4 summarises our contribution and discusses related research issues.

2 Efficient Resource Gathering

In RTS games, a player usually does not take a direct control of the agents that have been assigned to gather resources. When an agent is ordered to gather resources, the agent will move to a resource site, gather an amount of the resources, return to the nearest resource deposit, and then continue to gather resources until given a different order.

Depending on the RTS game, different numbers of agents may be allowed to mine a resource site. If a resource site is completely occupied when an agent returns to continue gathering, the agent may go to a different resource site if one exist, or wait until the resource site is available. Choosing a different resource site may led to the appearance of an agent regretting its previous goal and choosing a new goal. This behaviour will, in some cases, result in the agent leaving a resource site that is becoming available shortly for a resource site that will be occupied shortly, thus wasting time on traveling. Waiting for availability may cause an agent to spend time on waiting instead of moving to another resource site. Any time spent not moving to the correct resource site, or waiting for the wrong resource site, causes a loss compared to the optimal behaviour.

The erratic movement may cause the resource income rate to spike or drop—when an agent chooses a new path—making it difficult to predict the amount of resources available at a later point in time. To avoid this, a direct control can be applied to agents, where the future availability of the resource site is considered before they move.

2.1 Problem Definition

Given a set of agents $A = \{a_i | i = 1, \dots, n\}$ and a set of resource sites $M = \{m_j | j = 1, \dots, l\}$ located in a two-dimensional Euclidean space, each site m_j having an attached amount of resources $r_j \in \mathbb{Z}^+$, choose a subset $S \subseteq G$ of gathering tasks $G = A \times M$ such that the total amount of resources $R = \sum_{j=1}^l r_j$ is gathered in a minimal time T .

All agents have a fixed maximum capacity for carrying resources, r_k and will collect $r_a = \min(r_k, r_j)$ from a site before needing to unload their cargo at a

² <http://www.iccup.com/>

resource deposit D . The time required for the actual gathering is a constant C . For each site m_j there exists exactly one resource site queue, Q_j , containing a set of agents that will gather resource from this site. A resource site queue is defined as a totally ordered set of agents $Q_j = \{a_h \in A | h = 1, \dots, z_j\}$, $\forall a_h \in Q_j \implies a_h \notin Q_k$ where $j \neq k$, such that each agent is assigned to at most one resource site queue at a time. When an agent has finished gathering, it is removed from the queue. The first element $a_1 \in Q_j$ is the agent that may use the resource site first.

We define T to be the total time that is required to execute every gathering task $s_{i,j} = (a_i, m_j) \in S$. Specifically, a gathering task $s_{i,j}$ is completed in a round-trip time $t_{i,j}$ after agent a_i travels from D to a site m_j , potentially waits in line, collects resources and goes back to D .

Equation 1 shows a calculation of the round-trip time, which we aim to minimise by applying efficient resource gathering.

$$t_{i,j} = tt_{D,j}^i + \max[0, rt_j - tt_{D,j}^i] + C + tt_{j,D}^i. \quad (1)$$

where $tt_{D,j}^i$ is the time required for the agent a_i to travel from depot D to the resource site m_j . rt_j is the remaining time for site m_j to become available after all agents have completed their work in queue Q_j . Thus, the time agent a_i would wait in line is equal to the remaining time for the queue to become empty, rt_j , minus the time that has already been spent on the travel. The constant collecting time C is also added along with the travel time to return to the resource deposit $tt_{j,D}^i$.

By using site queues, it is possible to determine the best site to send agents in order to minimise the time required for the agent to return to D with an amount of resources. Equation 2 states the remaining time for an agent h in queue Q_j to finish gathering the resource m_j . c refers to the remaining collection time of the first agent in the queue, $a_1 \in Q_j$ where $0 \leq c \leq C$.

$$rt_j^h = \begin{cases} tt_{D,j}^h + c_h & \text{for } h = 1 \\ rt_j^{h-1} + \max[0, tt_{D,j}^h - rt_j^{h-1}] + C & \text{for } h \neq 1 \end{cases}. \quad (2)$$

The required time for agents ahead of a given agent for finishing the collection of resources from a resource site is included in the recursive definition of rt_j^h . The total time of Q_j is therefore $rt_j = rt_j^{z_j}$, meaning the time required for the last agent in Q_j to finish gathering the resource.

The time required for an agent to finish gathering is dependent on the time required for the preceding agent, as gathering can not occur before the preceding agent has completed gathering. The time required for the first agent in the queue, rt_j^1 , does not depend on any preceding agents as there are none.

2.2 Algorithm Definition

An algorithm was developed as a practical utilisation of the equations presented in Section 2.

Minimising the round-trip time will cause agents to insist a resource site that will allow for the fastest delivery of an amount of resources in a greedy fashion. The main mining algorithm, shown in Algorithm 1, uses a sub-algorithm $Work(Q, a)$ presented in Algorithm 2.

Algorithm 1. Resource gathering algorithm

Declarations

Q_1, \dots, Q_l : resource site queues

A : candidate queue

m_h : resource site

```

1: time  $\leftarrow \infty$ 
2: for all  $a_i \notin \bigcup_{j=0}^l Q_j$  do
3:   for  $x = 1 \dots l$  do
4:      $A \leftarrow Q_x \cup \{a_i\}$ 
5:     if  $Work(A, a_i) + tt_{w,D}^i < time$  then
6:        $time \leftarrow Work(A, a_i) + tt_{w,D}^i$ 
7:        $best \leftarrow h$ 
8:     end if
9:   end for
10:   $Q_{best} \leftarrow Q_{best} \cup \{a_i\}$ 
11: end for
```

Algorithm 1 is executed every time when an agent needs to be assigned to a resource site queue. In line 2, the algorithm iterates through every agent that is not currently assigned to a queue. Lines 3-10 iterate through the set of resource site queues to find the queue that requires the least amount of time for the agent to return with a deposit, and then adds the agent to the queue.

Algorithm 2 takes as parameters a queue Q and an agent a , and returns the total time spent on traveling to the resource site, waiting in line and gathering from the resource site. The algorithm works by recursively calculating the workload of the agents ahead in the queue (lines 4-5) to get waiting time in line. Since the time spent on traveling is included in the return value (line 12), the travel time will have to be subtracted from the waiting time (lines 7-11).

Consider the scenario on Fig. 2 where three agents, A, B and C , must gather resources from m_0 and m_1 by transporting resources from these positions to the drop-off location R . By applying Algorithm 1, we show the actions of three agents in Table 1. In the initialization, each queue is empty and agents are not assigned to any resource site.

The first three rows display each agent being assigned to their first resource site. The algorithm takes into account the distance to be travelled for each agent as well as the remaining time for agents that have already been assigned to a resource site. When agents return to R , Algorithm 1 is run again, to assign the agents to their next resource site. Over the course of a session, each agent may be assigned to different resource sites, depending on their availability.

Algorithm 2. *Work(Q, a)*Parameters Q : resource site queue a : agentDeclarations $w \leftarrow 0$: workload of the queue $z = |Q|$: number of elements in Q $p \leftarrow h$, where $a = a_h \in Q$, $1 \leq h \leq z$: position of a in Q

```

1: if  $p = 0$  then
2:   return  $tt_{D,j}^i + C$ 
3: end if
4: for  $h = 1 \dots p - 1$  do
5:    $w \leftarrow w + Work(Q, a_p)$ 
6: end for
7: if  $w > tt_{D,j}^i$  then
8:    $w \leftarrow w - tt_{D,j}^i$ 
9: else
10:   $w \leftarrow 0$ 
11: end if
12: return  $tt_{D,j}^i + w + C$ 

```

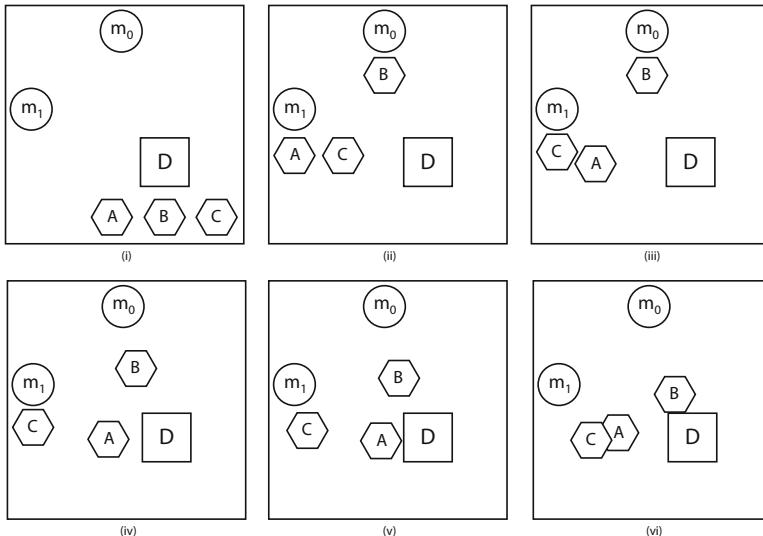


Fig. 2. (i) None of the agents have moved. A is assigned to m_1 , B to m_0 , C to m_1 . (ii) A is done using m_1 , B is gathering m_0 , C starts gathering m_1 . (iii) B is done using m_0 , A is moving to D , C is still gathering m_1 . (iv) C is done using m_1 , A is still moving to D , B is moving to D . (v) A delivers to D and is assigned to m_1 , B is still moving to D , C is moving to D . (vi) B delivers to D and is assigned to m_0 , A is moving to m_1 , C is still moving to D .

Table 1. A sample trace for the scenario on Fig. 2. From the left is the current time, the affected agent, the action of the agent, the current state of both resource site queues and the amount of gathered resources.

Time	Agent	Action	Queues	Resources
0	A	Move to m_1 (110)	$Q_{m_0} = \emptyset$ $Q_{m_1} = \{A[\text{work} \leftarrow 190]\}$	0
0	B	Move to m_0 (126)	$Q_{m_0} = \{B[\text{work} \leftarrow 206]\}$ $Q_{m_1} = \{A[\text{work} \leftarrow 190]\}$	0
0	C	Move to m_1 (126)	$Q_{m_0} = \{B[\text{work} \leftarrow 206]\}$ $Q_{m_1} = \{A[\text{work} \leftarrow 190], C[\text{work} \leftarrow 80]\}$	0
190	A	Move from m_1 (90)	$Q_{m_0} = \{B[\text{work} \leftarrow 16]\}$ $Q_{m_1} = \{C[\text{work} \leftarrow 80]\}$	0
206	B	Move from m_0 (108)	$Q_{m_0} = \emptyset$ $Q_{m_1} = \{C[\text{work} \leftarrow 64]\}$	0
270	C	Move from m_1 (90)	$Q_{m_0} = \emptyset$ $Q_{m_1} = \emptyset$	0
280	A	Deliver & move to m_1 (90)	$Q_{m_0} = \emptyset$ $Q_{m_1} = \{A[\text{work} \leftarrow 170]\}$	8
314	B	Deliver & move to m_0 (108)	$Q_{m_0} = \{B[\text{work} \leftarrow 188]\}$ $Q_{m_1} = \{A[\text{work} \leftarrow 136]\}$	16

2.3 Algorithm Analysis

In StarCraft: Brood War, agents do not move at a constant speed. An agent, given a movement order while standing still, will first accelerate towards a maximum speed and decelerate before reaching its destination. Since the testbed of the algorithm, StarCraft: Brood War, is a closed source, access to the algorithm used for pathfinding in the game is restricted. To compensate for this, travel times are measured and stored.

When an agent moves between the resource deposit and a resource site the time of the travel is recorded. Information on the travel is saved in a lookup table, including the starting position, the destination and the time. As the function will only be used to decide the travel time between a resource deposit and the stationary resource site, the amount of starting position/destination pairs are limited, making this approach possible. Whenever another agent is moving in a path equal to a recorded path, the original value is used and potentially revised. Given enough data, all possible paths an agent may use when mining resources, is known for a specific map.

The information has been gathered by using the resource gathering algorithm where the travel function is defined as a function that returns a previously calculated value. If no data matching the source and destination exists, the function returns a low value to allow the agent to take the path and thereby gather previously unknown data.

It should be noted that this approach necessitates that the environment is either static or that changes does not influence the travel times. This is not the case in StarCraft: Brood War, as new travelling paths may become available

when resource sites are depleted. Depleting a resource site causes the area previously blocked by the resource site to become passable. This does not impact the experiments performed for this paper as the experiments does not run long enough for any resource site to become depleted.

The time complexity of Algorithm 1, given k resource sites and n workers, is shown as follows: At a given run of the algorithm, the loop starting in line 2 is run a maximum of n times (when no workers have been assigned to a queue). In each of these loops, the loop starting in line 3 is run k times. Finally, in each of these runs, Algorithm 1 makes at most 2 calls to Algorithm 2, as seen in lines 5-6. This makes for the worst case scenario of $n * k * 2 * f(n)$, where $f(n)$ is the time complexity of Algorithm 2, given the maximal possible size of a queue, n .

$$f(n) = \begin{cases} 0 & \text{for } n = 1 \\ 2^{n-1} - 1 & \text{for } n > 1 \end{cases} .$$

The complexity of Algorithm 2 is due to its recursive call structure, as seen in line 5.

Algorithm 1 is general in the sense that it can be applied for RTS games having the following rules.

- Several resource sites are present
- A resource site contains several units of the resource
- Agents move between a resource site and a drop-off location
- Agents do not collide
- Only one agent may use a resource site at a time

The travel time between positions must either be known or be possible to be estimated. A general algorithm for calculating travel time has not been presented as the value is very dependant on the environment. If an estimate for the value is used, the algorithm does not guarantee to use the lowest round-trip time for each agent.

3 Experiments

We evaluated our algorithm against the built-in method used for resource gathering in StarCraft: Brood War, a science-fiction RTS game developed by Blizzard Entertainment in 1998. This particular game was chosen as the testbed for various reasons: StarCraft: Brood War is an add-on for the best selling RTS game to date³, incorporates all the general features of an RTS and is accessible through open source APIs.

³ According to VGChartz,
<http://www.vgchartz.com/worltdtotals.php?genre=Strategy>



Fig. 3. In-game screenshot from a game of StarCraft: Brood War featuring a range of agents and buildings

3.1 StarCraft: Brood War

A StarCraft: Brood War game features three distinct races each of which has individual technology trees. Each race has a unit type that is capable of gathering resources and constructing buildings. Two types of resource sites exist—minerals and gas—which are gathered by an agent traveling to the resource site, spending time gathering the resource site and then returning to a resource deposit to deliver the gathered resources. Fig. 3 shows an example of a game of StarCraft: Brood War. The middle building is the resource deposit of the *Terran* race. The blue crystals are resource sites containing minerals while the top left building is an extractor for the gas resource. The extractor must be constructed on a gas geyser and functions as a resource site for gas afterwards..

All communication with StarCraft: Brood War is done using the *Brood War Application Programming Interface*⁴ (BWAPI), an open source C++ framework allowing two-way communication with StarCraft: Brood War. BWAPI allows communication through objects and function calls, as opposed to input via human input devices. In effect, this allows the user, among other things, to observe events in the game and react upon these.

Currently, StarCraft as well as StarCraft: Brood War provides an automatic resource gathering system that is responsible for resource management in the game. The method is initialised by selecting an agent and invoking the *gather* command on a specific resource site. The agent will then move to and gather

⁴ <http://code.google.com/p/bwapi>

from the resource site, return to the resource depot with a deposit, move to the resource site again, and so forth. If a resource site is occupied on arrival, the agent will move to the closest resource site available and start gathering from that instead.

3.2 Experimental Results

We implemented our algorithm and compared the performance with the built-in resource gathering method in StarCraft: Brood War. We used both methods while creating 18 agents on the map *Astral Balance*. Each time an agent was created, or had performed its task, it was assigned to a resource site queue, using the algorithm. In order to ensure fairness, all tests were performed in the upper right corner, even though the difference of starting positions for agents is usually considered negligible. The starting position is next to a mineral cluster consisting of eight mineral fields, each holding 1500 minerals, as well as a single gas geyser.

We evaluate the performance of both algorithms on two aspects. One is the average cumulative amount of resources collected by agents over the course of a game, and the other is a standard deviation of gathered resources as time passes.

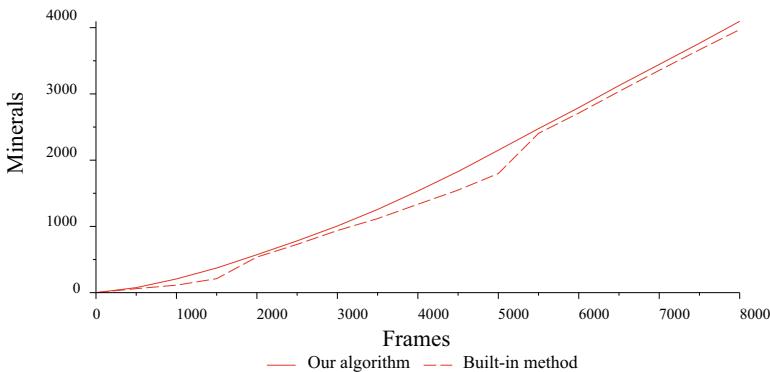


Fig. 4. Comparison between two methods, showing the amount of minerals gathered over time

Fig. 4 shows a comparison between two methods on the amount of collected minerals in the course of 8000 frames. The graph shows the average amount of minerals, collected at different time intervals during experimental runs of StarCraft: Brood War. The graph is based on data of 10 runs. Clearly, our algorithm increases the income of the player compared to the built-in method. Furthermore, the smoothness of the curve for our algorithm as opposed to the

built-in method indicates a higher predictability. The predictability is further elaborated in Fig. 5, depicting the standard deviation in the course of the 10 runs. Fig. 5 shows that the standard deviation of the minerals collected by both methods grows as time passes. There is a clear tendency that the deviation of the built-in method grows faster than that of the proposed algorithm.

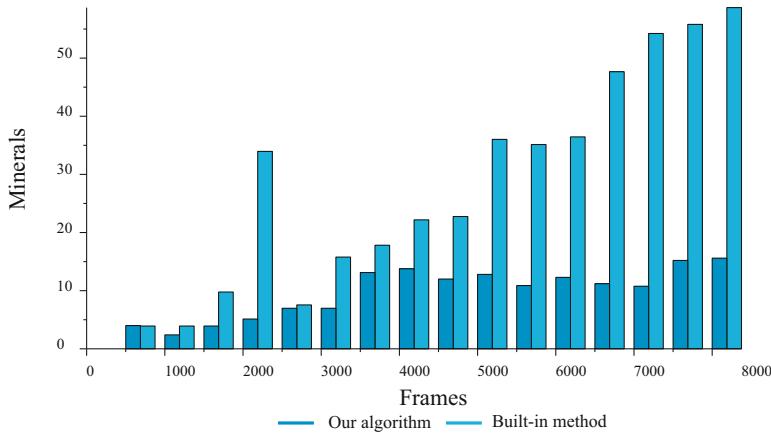


Fig. 5. Comparison of the standard deviation of the both methods

4 Discussion

Exploiting game data to compute travel times facilitates the design of a resource gathering algorithm in spite of an unknown pathfinding method. Experiments show that the algorithm provides an increase in the amount of resources gathered, compared to the built-in approach. Furthermore the experiments show that our algorithm is more predictable. The predictability can be used to predict the amount of resources collected within some time frames.

Currently our algorithm is *selfish* in the sense that each agent makes decisions that allow for the fastest mineral gain for itself, but not necessarily the best for all agents in a team. It might be beneficial to run the algorithm over a group of agents by having them act as a team and optimising the group income instead of the individual income over time.

The increased predictability of income using our algorithm makes it eligible for use in other aspects of an RTS game. An example is to develop a scheduler for construction of, for example, military units. By using our algorithm it is possible to predict the time when the scheduled production line is finished, even if the resources for the units should first be gathered. The algorithm enables the possibility of acquiring an accurate estimate on the potential gain from utilising resource sites from other areas in the game world. This is interesting for RTS games in which new resource deposits may be created by a player.

Notice that the algorithm depends on the implementation for computing travel times, potentially necessitating a consideration of agent collisions. In some RTS games, the travel time is insignificant due to the distance from a resource site to a resource deposit, the number of agents or other game specifics. In RTS games similar to StarCraft: Brood War we expect to observe similar performance of our proposed algorithm. The simple approach to calculating travel times, presented in Section 2.3, only works in a static environment which is, in our case, ensured by not running the game long enough for any resource site to deplete. In essence, this keeps the environment static. For a travel time function to work in a dynamic environment, an approach complying to environmental change is needed.

References

1. Cao, L., Gorodetsky, V., Mitkas, P.A.: Agent Mining: The Synergy of Agents and Data Mining. *IEEE Intelligent Systems* 24(3), 64–72 (2009)
2. Chan, H., Fern, A., Ray, S., Wilson, N., Ventura, C.: Online Planning for Resource Production in Real-Time Strategy Games. In: Proceedings of the Seventeenth International Conference on Automated Planning and Scheduling, pp. 65–72 (2007)
3. Rabin, S. (ed.): *Introduction to Game Development, A Brief History of Video Games*, ch. 1.1, pp. 3–36. Charles River Media Inc. (2005)
4. Sharma, M., Holmes, M., Santamaria, J., Irani, A., Isbell, C., Ram, A.: Transfer Learning in Real-Time Strategy Games Using Hybrid CBR/RL. In: Proceedings of the Twentieth International Joint Conference on Artificial Intelligence (2007)
5. Weber, B., Mateas, M.: A Data Mining Approach to Strategy Prediction. In: Proceedings of the 5th International Conference on Computational Intelligence and Games, pp. 140–147. IEEE Press (2009)
6. Wintermute, S., Joseph, X., Laird, J.E.: SORTS: A Human-Level Approach to Real-Time Strategy AI. In: Proceedings of the Third Artificial Intelligence and Interactive Digital Entertainment Conference. The AAAI Press (2007)

Data Cloud for Distributed Data Mining via Pipelined MapReduce

Zhiang Wu, Jie Cao, and Changjian Fang

Jiangsu Provincial Key Laboratory of E-Business,
Nanjing University of Finance and Economics, Nanjing, P.R. China
`zawuster@gmail.com, {caojie690929, jselab1999}@163.com`

Abstract. Distributed data mining (DDM) which often utilizes autonomous agents is a process to extract globally interesting associations, classifiers, clusters, and other patterns from distributed data. As datasets double in size every year, moving the data repeatedly to distant CPUs brings about high communication cost. In this paper, data cloud is utilized to implement DDM in order to move the data rather than moving computation. MapReduce is a popular programming model for implementing data-centric distributed computing. Firstly, a kind of cloud system architecture for DDM is proposed. Secondly, a modified MapReduce framework called pipelined MapReduce is presented. We select Apriori as a case study and discuss its implementation within MapReduce framework. Several experiments are conducted at last. Experimental results show that with moderate number of map tasks, the execution time of DDM algorithms (i.e., Apriori) can be reduced remarkably. Performance comparison between traditional and our pipelined MapReduce has shown that the map task and reduce task in our pipelined MapReduce can run in a parallel manner, and our pipelined MapReduce greatly decreases the execution time of DDM algorithm. Data cloud is suitable for a multitude of DDM algorithms and can provide significant speedups.

Keywords: distributed data mining (DDM), Cloud Computing, MapReduce, Apriori, Hadoop.

1 Introduction

The last decade has witnessed the successful development of agents and data mining techniques which have been applied to a variety of domains - marketing, weather forecasting, medical diagnosis, and national security [1,2]. As data mining become more pervasive, the amount of data is increasing larger. The great amount of data is often partitioned into many parts and distributed in many sites. Distributed data mining (DDM) is a process to extract globally interesting associations, classifiers, clusters, and other patterns from distributed data, where data can be partitioned into many parts either vertically or horizontally [3,4].

Agents are scattered to many sites for handling distributed problem-solving, cooperation and coordination. A high performance DDM system is designed to

control agents and exploit the synergy of agents. Traditional DDM system has been designed to take advantage of powerful, but shared pools of CPUs. Generally speaking, data is scattered to the processors, the computation is performed using a message passing, the results are gathered, and the process is repeated by moving new data to the CPUs [5]. As CPU cycles become cheaper and data sets double in size every year, the main challenge for efficient scaling of applications is the location of the data relative to the available computational resources - moving the data repeatedly to distant CPUs is becoming the bottleneck [6].

On the basis of cluster computing, P2P, Grid computing and Web 2.0, cloud computing rapidly emerges as a hot issue in both industrial and academic circles [7]. Cloud computing has been adhering to the belief that moving computation is cheaper than moving data since its birth. Therefore, cloud computing is suitable for solving computation-intensive and data-intensive problems in DDM. MapReduce has emerged as an effective method to implement the *data-centric* belief held by cloud computing. MapReduce has been applied to a multitude of domains for data processing, such as machine learning, satellite data processing, PageRank, scientific data analysis, etc. Since traditional MapReduce stores all middle results in file system and most of DDM applications produce a large amount of middle results, preserving storage for temporary files is extremely expensive and inefficient. Moreover, in traditional MapReduce framework, the reduce task does not start running until all map tasks are finished. This sequential execution manner between map task and reduce task increases the job completion time.

Motivated by the above remarks, this paper proposes a kind of data cloud system architecture for DDM. We make an improvement in the traditional MapReduce framework, namely, pipelined MapReduce framework. The dataflow and the fault tolerance strategy of pipelined MapReduce are addressed in detail. We then discuss the implementation of Apriori algorithm, which is a well-known algorithm for tackling this problem, in MapReduce framework. At last, experimental evaluation of our work is presented.

2 Related Work

2.1 Cloud Computing

In 2007, IBM and Google first proposed cloud computing. Currently, providers such as Amazon, Google, Salesforce, IBM, Microsoft and Sun Microsystems have begun to establish new data centers for hosting cloud computing applications in various locations around the world to provide redundancy and ensure reliability in case of site failures [8]. Data cloud, which is proposed by R. Grossman and Y. Gu in 2008, refers to a class of cloud systems that provide resources and/or data services over the Internet [5].

The Google's MapReduce programming model, which serves for processing large data sets in a massively parallel manner, is a representative technique in cloud computing [9]. Hadoop developed by the Apache Software Foundation is an open source MapReduce framework [10]. The key components of Hadoop are

HDFS (Hadoop Distributed File System) and MapReduce. HDFS is a distributed file system designed to run on hardware, and it also manages all files scattered in nodes and provides high throughput of data access. MapReduce provides a programming model for processing large scale datasets in distributed manner. Jobs submitted to Hadoop consist of a map function and a reduce function. Hadoop breaks each job into multiple tasks. Firstly, map tasks process each block of input (typically 64MB) and produce intermediate results, which are key-value pairs. These are saved to disk. Next, reduce tasks fetch the list of intermediate results associated with each key and run it through the reduce function, which produces output. Hadoop is selected as core middleware in data cloud proposed in this paper.

MapReduce is a good fit for problems that need to analyze the whole dataset, in a batch fashion. In the meanwhile, MapReduce works well on unstructured or semi-structured data, for it is designed to interpret the data at processing time. In other words, the input keys and values for MapReduce are not an intrinsic property of the data, but they are chosen by the programmer. The input of MapReduce job can be either a text format or data stored in Hbase which is a distributed, column-oriented database provided by Hadoop.

Agents' pro-activeness is suitable for collecting information about resource status and databases. Traditional monitoring systems generate alarm when a failure occurs by monitoring resources continuously. The storm of alarm often occurs. Utilizing agent to gather information can avoid "the storm of alarm" and extract valid knowledge from data which is known as data intelligence in [11].

2.2 Mining Association Rules upon Parallelizing (MARP)

Association rules mining is an important branch of data mining. Since the mass-data often is often distributed in many sites and putting all data together to amass a centralized database is unrealistic, MARP employs distributed computing technology to implement concurrent data mining algorithms. MARP developments endeavor to scale up data mining algorithms by changing existing sequential techniques into parallel versions. Implementing data mining algorithms within MapReduce framework belongs to MARP. However, parallel algorithms within MapReduce have at least two advantages over traditional MARP techniques.

- Parallel algorithms within MapReduce can process unstructured or semi-structured data conveniently.
- Since MapReduce tries to colocate the data with the compute node, data access of algorithms within MapReduce is fast for it is local.

"Market-Basket Analysis problem", a classic association rules mining problem, is taken as a case study. The supermarket owners may be interested in finding associations among its items purchased together at the check-stand [12]. Apriori is a seminal algorithm for finding frequent itemsets using candidate generation [13]. It is characterized as a level-wise complete search algorithm using

anti-monotony of itemsets, “if an itemset is not frequent, any of its superset is never frequent”. Since Apriori is time consuming and the transaction database is currently distributed in many sites, it is necessary to apply parallel and distributed methods to Apriori. However, the major difficulty lies in that computing support of an itemset should scan the whole database, but each node only stores a split of the whole database. To tackle this difficulty, two kinds of methods have been presented. The first method takes pre-treatment on database to decompose the database into several fragments, and then each node could run totally independently on each fragment of database to compute frequent itemsets using a classical sequential algorithm of Apriori. At last, final results are obtained from these nodes. Although V. Fiolet has suggested three fragmentation methods [12], these methods are time consuming and none of them can be applied to all databases. Furthermore, the pre-processing on database leads to restore data, and data should be moved among clusters (or PC servers), which violates the essence of cloud computing-moving computation is cheaper than moving data. The second method is called *CountDistribution* which is incorporated in this paper. In cloud environment, assuming that the database is initially distributed in a horizontal split, namely non-overlapping subsets of records are randomly stored in clusters (or PC servers). Each node can thus independently get partial supports of the candidates from its local database fragmentation. Reducer then executes reduce function to compute the sum of supports with the same key. Note that only the partial counts need to be communicated, rather than the records of the database. Since the partial counts are represented as $(key, value)$ pairs in MapReduce, the method can minimize communication.

3 Data Cloud System Architecture for DDM

Based on campus grid environments we have designed and implemented layered data cloud system architecture as Fig. 1 depicted. Physical cloud resources along with core middleware form the basis of the system. The user-level middleware aims to provide PaaS (platform as a service) capabilities. The top layer focuses on application services by making use of services provided by the lower layer services. Emerging DDM applications such as social security, enterprise, stock markets and scientific workflows can operate at the highest layer of the architecture. The representative data mining algorithms used by DDM applications such as Apriori, PageRank, kNN and k-Means can be implemented in data cloud system, and some of them even can be implemented in MapReduce framework to improve their performance.

- System level: Physical resources integrated by data cloud system consist of two clusters each with 32 blade servers and several PC servers. The total number of CPU reaches 140.
- Core middleware: This layer is comprised of two sub-layers: VM (virtual machine) and Hadoop. VM management and deployment transparently virtualizes these servers and shares their capacity among virtual instances of

servers. These VMs are isolated from each other, which aid in achieving fault tolerant behavior and isolated security context. On the top of VMs, Hadoop framework is deployed, on which Java programs based on MapReduce model can be executed. All data are stored on HDFS. The master node called *JobTracker* is the point of interaction where the user submits jobs, along with location of the input data on the HDFS. The *JobTracker* assigns and distributes the map and reduce tasks to the *TaskTrackers* which assumes the role of worker nodes. The *TaskTracker* performs the task and updates the status to the *JobTracker*. In the MapReduce framework of this architecture, pipeline is added between Mapper and Reducer. The new MapReduce framework is called Pipelined MapReduce framework which will be discussed in the next section.

- User-level middleware: This layer provides Web 2.0 programming paradigms such as JavaScript with AJAX, Ruby with Ruby on Rail, and PHP etc. Users can utilize Web APIs to create novel browser-based applications. This layer also provides the programming environments and composition tools that facilitate the creation, deployment, and execution of applications in clouds.
- Security management: This module provides authentication and permission control for cloud users. Single sign-on is adopted by cloud system architecture proposed in this paper. Cloud user obtains long-term certificate from Certificate Authority (CA), and user can encrypt this long-term certificate to generate temporary proxy certificate which is used to identify user by data cloud. The deadline of proxy certificate is often short (i.e. 12 hours in our data cloud) to decrease the harm created by various network attacks.
- Agent-based monitoring system: Agents are deploy to all resources including clusters, servers, software, database and jobs. Agent is a flexible and powerful toolkit for displaying, monitoring and analyzing results to make the best use of the collected data. Status data gathered by agents is stored to Tivoli data warehouse and is displayed in portal. Since virtualization technology is used in data cloud, the relation between agent and resource is many-to-one.

4 Pipelined MapReduce Framework

In the data cloud system proposed in this paper, we make an improvement in the traditional MapReduce framework, namely, Pipelined MapReduce Framework. In traditional MapReduce framework, the output of each Mapper is managed by the *OutputCollector* instance and stored in an in-memory buffer. The *OutputCollector* is also responsible for spilling this buffer to disk (i.e., HDFS in Hadoop) when the output reaches capacity of memory. The execution of a reduce task includes three phases: *shuffle* phase, *sort* phase and *reduce* phase. The *shuffle* phase fetches intermediate results from each Mapper. However, *a reducer cannot fetch the output of a mapper until JobTracker informs that the mapper is finished*. The output produced by Reducers may be required by the next map step, which is also written to HDFS. There are at least two disadvantages within the traditional MapReduce framework.

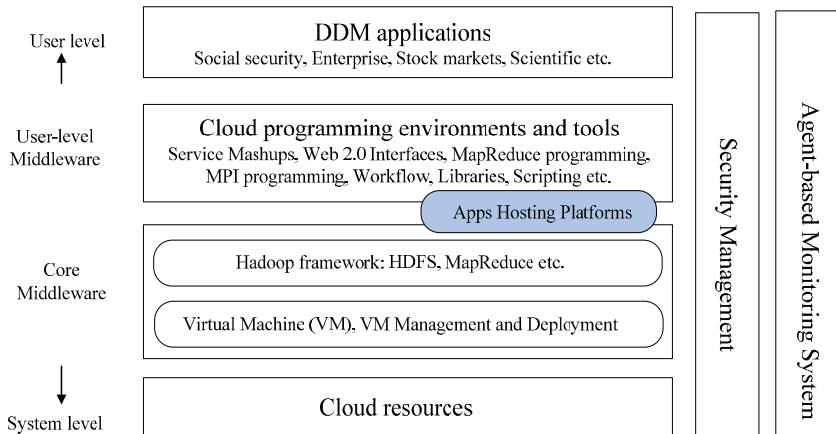


Fig. 1. Data cloud system architecture for DDM

- HDFS should maintain enough storage for temporary files. Since each file has three copies in HDFS in default manner and most of DDM applications will produce a large amount of middle results, preserving storage for temporary files is extremely expensive and inefficient.
- Because Mapper and Reducer execute in a serial manner, and both Mappers and Reducers should spend plenty of time in reading middle data from HDFS, the execution speed of the traditional MapReduce framework is very slow.

To solve these above-mentioned disadvantages, pipeline is added between Mapper and Reducer. Middle data is stored in pipeline files and only final results are written to HDFS. Moreover, when the Mapper is executing, it simultaneously pushes middle data to Reducer via pipeline. Reducers then pull the middle data synchronously. Pipelined MapReduce Framework makes Mappers and Reducers execute in a parallel manner and also enhance the robustness of the fault tolerance.

4.1 The Dataflow of the Pipelined MapReduce Framework

Fig. 2 compares the dataflow between traditional and pipelined MapReduce framework. The dataflow on the right depicts the approach utilized by the pipelined MapReduce framework. Each Mapper obtains its input data from HDFS, and when the Mapper is executing, it simultaneously pushes middle data to pipeline. It is noted that when the intermediate data exceeds the memory size, the intermediate data should also be written to HDFS. Each Reducer synchronously pulls data from pipeline and starts running. The middle data

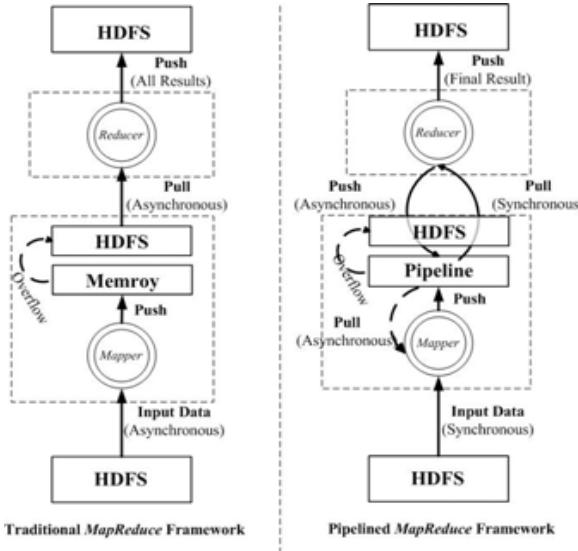


Fig. 2. The dataflow comparison between traditional and pipelined MapReduce framework

produced by the Reducer, which may be required by the next map step, is also pushed to pipeline. Final result produced by the Reducer is written to HDFS.

The precondition that reducers and mappers can run concurrently is that the reduce function of the application is incrementally computable. The applications exist commonly, such as sorting, graph algorithms, Bayes classification, TF-IDF (Term Frequency - Inverse Document Frequency), Apriori, and so on. The pipeline between Mapper and Reducer is implemented through utilizing TCP socket. Each Reducer contacts every Mapper upon initiation of the job, and opens a socket which will be used to pipeline the output of the map function. As each map output is produced, the Mapper determines which partition the record should be sent to, and immediately sends it via the appropriate socket. A Reducer accepts the pipelined data that it has received from each Mapper and then stores it in an in-memory buffer. The Reducer may start running on the basis of these partial data. Once the Reducer learns that all Mappers have completed, it continues to perform the remaining task and writes the output to pipeline or to HDFS. One practical problem in the above-mentioned method is that when the number of mappers becomes large, each reducer should maintain a large number of TCP connections. To reduce the number of concurrent TCP connections, we restrict the number of connections with mappers at once. The reducer obtains data from the remaining map tasks in the traditional Hadoop manner.

4.2 The Fault Tolerance Strategy of the Pipelined MapReduce Framework

It is known that the role of cluster node is divided into *master* and *worker*. Both mappers and reducers are *worker* nodes. Although Hadoop could handle master failure, the case unlikely exists [14]. The *master* detects worker failure via periodic heartbeats. New fault tolerance strategy is implemented in the pipelined MapReduce framework. The reducer treats the output of a pipelined map task as “tentative” until the *JobTracker* informs the reducer that the mapper has committed successfully. The reducer merges together spill files generated by the same uncommitted mapper. Log is added to reducer to record which mapper produced each pipelined spill file. Thus, if a mapper fails, each reducer can ignore any tentative spill file produced by the failed map attempt. The *JobTracker* will then restart a new mapper. If a Reducer fails and a new copy of the task is started, all the input data that was sent to the failed reducer must be sent to the new reducer. To prevent mappers from discarding their output after sending it to pipeline, mappers should retain their output data until the entire job is completed successfully. This allows the output of mappers to be reproduced if any reducer fails.

5 Case Study

Finding frequent itemsets from a transaction database and deriving association rules is called “Market-Basket Analysis” problem which is one of the most popular data mining problems. This section takes ”Market-Basket Analysis” problem as a case study. Although “Market-Basket Analysis” problem has been described in many papers, to make it clear and easy for understanding, we briefly describe it. Given a transaction database D , the number of its records is denoted as $|D|$. We assume there are n distinct items in D , denoted as $I = \{i_1, i_2, \dots, i_n\}$.

Remark 1. The support of an itemset X is the percentage of records in the database D , which contains this itemset X . The support measures how interesting the itemset is, that is, its frequency in the database. The support of the itemset X can be calculated by equation (1).

$$\text{support}(X) = \frac{D(X)}{|D|} \quad (1)$$

where $D(X)$ is the number of records containing itemset X in database D , and $|D|$ is the total number of records of database D . The “Market-Basket Analysis” problem is to find out all association rules whose support is greater than the given thresholds. The support threshold is defined by user.

5.1 Apriori Algorithm within the Framework of MapReduce

A well-known algorithm for the computation of frequent itemsets is the Apriori algorithm which is used as follows:

- to compute the supports of items, and then to identify frequent items (frequent 1-itemsets)
- to generate candidate 2-itemsets, to count their supports, and then to identify frequent 2-itemsets
- ” to generate candidate 3-itemsets, to count their supports, and then to identify frequent 3-itemsets, and so on ...

To compact the search space of Apriori, the guiding principle "every subset of a frequent itemset has to be frequent" is utilized.

As section 2 described, this paper incorporates *CountDistribution* method to implement Apriori within the framework of MapReduce. In cloud environment, assuming that the database is initially distributed in a horizontal split, namely non-overlapping subsets of records are randomly stored in clusters (or PC servers). These nodes which store the splits of the database execute map function and become *Mappers*. A program implementing the map function of Apriori is sent to all mappers. Each mapper can thus independently get partial supports of the candidates from its local database fragmentation. All partial results from mappers are collected to one or several nodes called *Reducer(s)*. And then, Reducer executes reduce function to compute the sum of supports with the same key and global frequent itemsets $F_k - 1$ are obtained. Note that only the partial counts need to be communicated, rather than the records of the database. Since the partial counts are represented as $(key, value)$ pairs in MapReduce, this method can minimize communication. Fig. 3 depicts the MapReduce framework of Apriori.

Since Apriori algorithm utilizes iteration to obtain all frequent itemsets, it needs to scan database at most $n+1$ times when the maximum size of frequent itemsets is set at F_n . Therefore, *mappers* should be started at most $n+1$ times. Within the framework of MapReduce, though reduce operation also can be run in parallel manner, *reducer* of Apriori is run in a single node because the reduce operation of Apriori only needs to carry out simple count operation. The pseudocode of map and reduce function executed by *mappers* and *reducer* respectively for the Apriori is as follows.

The map function for the Apriori

```

Input:the split of database,
      and the last global frequent itemset  $F_{k-1}$ 
Output: (key, value)pairs
1:  $C_k := \text{AprioriGen } (F_{k-1})$ 
2: for each element  $c$  in  $C_k$ 
3:   scan the split of database and count support for  $c$ 
4:   generate pair  $(c, c.\text{support})$ 
5: end for
6: return Union( $(c, c.\text{support})$ )

```

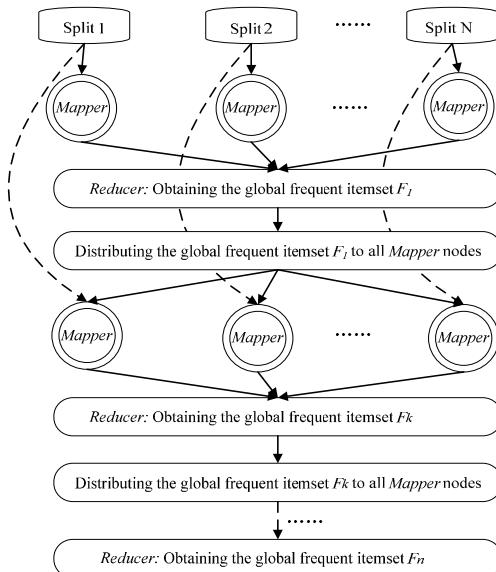


Fig. 3. MapReduce framework of Apriori

The reduce function for the Apriori

```

Input: (c, c.support) pairs generated by Mapper nodes
Output: the global frequent itemset Fk
1: for all (c, c.support) pairs with same c
2: compute the sum of their support
3: denote the sum as (c, sum_support)
4: end for
5: Fk := {all (c, sum_support) pairs | c.sum_support >= min_sup}
6: return Union(Fk)

```

In the Map function, AprioriGen function generates new candidates denoted as C_k based on the last global frequent itemset F_{k-1} . Each mapper scans its split of database and counts support for all elements in C_k . In the Reduce function, all (key, value) pairs from all mappers are grouped by key and supports of the pairs with the same key are added. The pairs whose sum of support is greater than threshold are selected into the current global frequent itemset F_k .

This case study indicates that the MapReduce framework provided by data cloud system architecture exhibits good scalability to other DDM algorithms. We needn't take any pre-treatment on database and needn't move data among clusters. Only map function and reduce function of DDM algorithms need to be provided. Data cloud can automatically move programs containing map function to a multitude of nodes storing the input data.

5.2 Performance Evaluation

We have built a data cloud system based on the architecture shown in Fig. 1. Machines integrated by this data cloud system consist of two clusters each with 32 blade servers in SEUGrid (Southeast University Grid) and an army of PC servers in our laboratory. Two VMs are created in each PC servers with 2.03GH and 1GB of RAM. Two clusters in SEUGrid utilize Red Hat Enterprise 4.0 and all VMs in PC servers utilize Fedora Core 11, and on the top of Linux, Hadoop 0.20.1 is installed and configured as core middleware. The transaction database contains two attributes: one is the transaction identifier and the other is the list of items. In the first experiment, we investigate the effect of the number of *mappers* on the execution time to obtain all frequent itemsets utilizing Apriori. A transaction database is created with 100,000 records and 500 kinds of items. The transaction database is split on average according to the number of *Mapper* nodes. In other words, equal number of records is stored in all *mapper* nodes. Fig. 4 shows the effect of number of *mappers* on execution time. It indicates that with the increase of the number of *mappers*, the execution time decreases. At the beginning, the execution time decreases dramatically with the increase of the number of *mappers*. When the number of *mappers* reaches a threshold (i.e. 16 in Fig. 4), the execution time varies moderately or even increases slightly (i.e. 24, 28 and 32 in Fig. 4). The reason for the above-mentioned phenomenon is that with the increase of *mappers*, the number of records stored in each *mapper* node decreases. When the number of records reduce to a threshold, the time of scanning the database is hard to further decline. Therefore, the time of *Reducer* and communications is rising, thus playing the major role in the whole execution time.

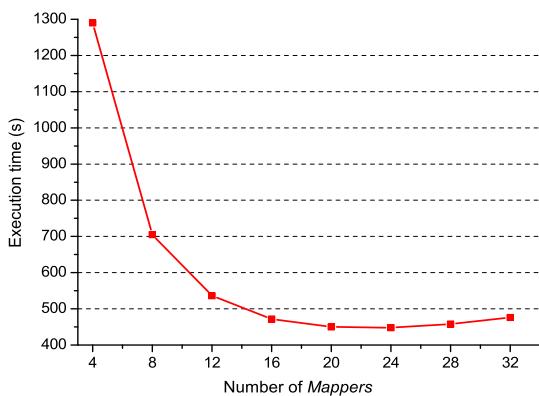


Fig. 4. Effect of number of Mappers on execution time

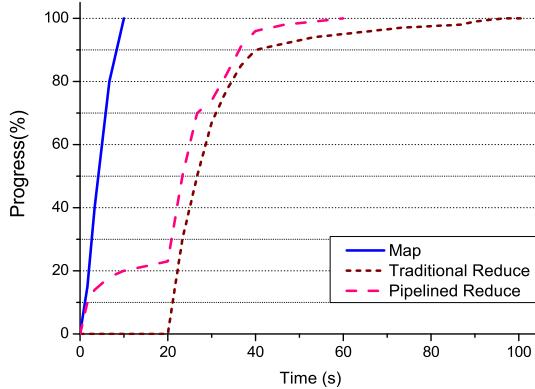


Fig. 5. Comparison of map and reduce task completion times between traditional and pipelined MapReduce

The second experiment compares pipelined MapReduce with traditional MapReduce. We analyze map progress and reduce progress in an iteration of Apriori (i.e. computing F_k based on F_{k-1}). Hadoop provides support for monitoring the progress of task executions. As each task executes, it is assigned a *progress score* in the range $[0, 1]$, based on how much of its input that the task has consumed. There are 80,000 records split on 8 mappers including 500 kinds of items. The first iteration computing F_1 based on the input from HDFS is selected as a statistical object. Fig. 5 describes the progress score of map task and reduce task along with the timelapse. The map tasks in both traditional and pipelined MapReduce are same for their input is read from HDFS. There exists obviously difference between the progress of traditional reduce and the progress of pipelined reduce. Traditional reduce task starts after the map task and it consumes more time than pipelined reduce task. However, pipelined reduce task starts immediately after the map task starts, and they are running in a parallel manner (i.e. from 0s to 10s). These results suggest that pipelined MapReduce framework can substantially reduce the response time of a job. In the third experiment, we implement other applications in Hadoop and investigate the speedup as we scale the number of processor cores. The following are brief descriptions of the selected applications:

- WordCount: It counts the frequency of occurrence for each word in a set of files. Mappers process different sections of the input files and return intermediate data that consist of a word (key) and a value of 1 to indicate that the word was found. The reducers add up the values for each word (key).
- Kmeans: It implements the popular Kmeans algorithm to group a set of input data points into clusters. Since Kmeans is iterative, in each iteration, mappers find the distance between each point and each mean and assign the point to the closest cluster. For each point, we emit the cluster ID as the key

and the data vector as the value. Reducers gather all points with the same cluster ID, and finds their mean vector.

- Reverse Index: It traverses a set of HTML files, extracts all links, and compiles an index from links to files. Each mapper parses a collection of HTML files. For each link it finds, it outputs an intermediate pair with the link as the key and the file info as the value. The reducer combines all files referencing the same link into a single linked-list.

We evaluate the speedup of four algorithms as we scale the number of processor cores used in data cloud. The concept of speedup stems from high-performance computing (HPC). *Speedup* is defined as the ratio of the execution time of an algorithm on one node to the execution time of the same algorithm on several nodes. Fig. 6 presents the experimental result. Data cloud provides significant speedups for all processor counts and all algorithms. With a large number of cores, the speedup of some applications cannot be improved further (i.e., WordCount, Kmeans, Apriori), where the reason is similar to the analysis in the first experiment. The speedups of Kmeans and Apriori are smaller than other algorithms, because they contain iteration process. Fitting iterative algorithms into the MapReduce framework leads to major overheads compared to sequential code and reduces the overall speedup.

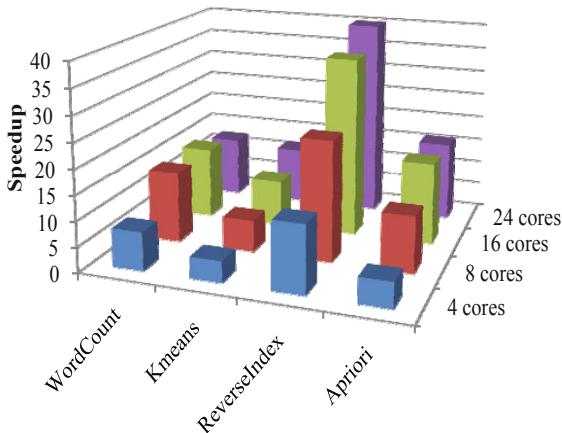


Fig. 6. Speedup of four algorithms with increase of processor cores

6 Conclusion

The increasing amount of data has led to concerns regarding the execution efficiency of DDM. The *data-centric* belief held by cloud computing can be utilized to enhance the execution efficiency of DDM. In this paper, we propose a kind of data cloud system architecture for DDM. Pipelined MapReduce framework is presented and implemented in data cloud. We then take Apriori algorithm as

a case study and implement Apriori within the framework of MapReduce. This case study indicates that data cloud system architecture proposed in this paper exhibits good scalability to various DDM algorithms. The fragmentation (or pre-treatment) methods of database needn't to be considered. Users should only provide map function and reduce function of DDM algorithms to data cloud. Data cloud can move computation in terms of the initial data location. We conduct several experiments in order to evaluate our work. The experimental results indicate that the execution time can be reduced remarkably with moderate number of mappers. Performance comparison between traditional MapReduce and our pipelined MapReduce has shown that: (1) the map task and reduce task in our pipelined MapReduce can run in a parallel manner; (2) our pipelined MapReduce greatly decreases the execution time of DDM algorithm. Data cloud is suitable for a multitude of DDM algorithms and can provide significant speedups.

Acknowledgments. This research is supported by National Natural Science Foundation of China under Grants No.71072172, the program for New Century Excellent Talents in university under Grants No.NCET-07-0411, Jiangsu Provincial Key Laboratory of Network and Information Security (Southeast University) under Grants No. BM2003201, Transformation Fund for Agricultural Science and Technology Achievements under Grants No. 2011GB2C100024 and Innovation Fund for Agricultural Science and Technology in Jiangsu under Grants No. CX(11)3039.

References

1. Cao, L., Gorodetsky, V., Mitkas, P.A.: Agent Mining: The Synergy of Agents and Data Mining. *IEEE Intelligent Systems* 24(3), 64–72 (2009)
2. Pech, S., Goehner, P.: Multi-agent Information Retrieval in Heterogeneous Industrial Automation Environments. In: Cao, L., Bazzan, A.L.C., Gorodetsky, V., Mitkas, P.A., Weiss, G., Yu, P.S. (eds.) *ADMI 2010. LNCS*, vol. 5980, pp. 27–39. Springer, Heidelberg (2010)
3. Yi, X., Zhang, Y.: Privacy-preserving naïve Bayes classification on distributed data via semi-trusted mixers. *Information Systems* 34(3), 371–380 (2009)
4. Cao, L.: Domain-Driven Data Mining: Challenges and Prospects. *IEEE Transactions on Knowledge and Data Engineering* 22(6), 755–769 (2010)
5. Grossman, R., Gu, Y.: Data mining using high performance data clouds: experimental studies using sector and sphere. In: Proceeding of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 920–927 (2008)
6. Szalay, A., Bunn, A., Gray, J., Foster, I., Raicu, I.: The Importance of Data Locality in Distributed Computing Applications. In: NSF Workflow Workshop (2006)
7. Above the clouds: A Berkeley View of Cloud computing. UCB/EECS-2009-28 (2009)
8. Buyya, R., Yeo, C.S., Venugopal, S., Broberg, J., Brandic, I.: Cloud computing and emerging it platforms: Vision, hype, and reality for delivering computing as the 5th utility. *Future Generation Computer Systems* 25(6), 599–616 (2009)

9. Ralf, L.: Google's MapReduce programming model - Revisited. *The Journal of Science of Computer Programming* 70(1), 1–30 (2008)
10. Hadoop: The Apache Software Foundation, <http://hadoop.apache.org/core>
11. Cao, L., Luo, D., Zhang, C.: Ubiquitous Intelligence in Agent Mining. In: Cao, L., Gorodetsky, V., Liu, J., Weiss, G., Yu, P.S. (eds.) *ADMI 2009. LNCS*, vol. 5680, pp. 23–35. Springer, Heidelberg (2009)
12. Fiolet, V., Tournel, B.: Distributed Data Mining. *Scalable Computing: Practice and Experience* 6(1), 99–109 (2005)
13. Wu, X., Kumar, V., Ross Quinlan, J., Ghosh, J., Yang, Q., Motoda, H., McLachlan, G.J., Ng, A., Liu, B., Yu, P.S., Zhou, Z., Steinbach, M., Hand, D.J., Steinberg, D.: Top 10 algorithms in data mining. *Knowledge and Information Systems* 14(1), 1–37 (2008)
14. Hadoop, W.T.: *The Definitive Guide*. O' Reilly Publishers (2010)

Successful Efficient and Intelligent Retrieval Using Analytic Hierarchy Process

Monika Arora¹, Uma Kanjilal², and Dinesh Varshney^{3,4}

¹ Department of IT, Apeejay School of Management,
Dwarka Institutional Area, New Delhi-110075, India

² Department of Library and Information Science, Indira Gandhi Open University,
Maidan Garhi, New Delhi-110 068, India

³ School of Physics, Devi Ahilya University, Khandwa Road Campus,
Indore- 452001, M.P. India

⁴ Multimedia Regional Centre, Madhya Pradesh Bhoj (Open) University,
Khandwa Road Campus, Indore- 452001, M.P. India

Abstract. An overview of successful data/information retrieval is presented in this article. It briefly reviews importance of efficient data retrieval and intelligent data retrieval. The study focuses on the factors and sub-factors dependent on retrieval mechanism. It discusses system, user and data as important categories of successful efficient and intelligent data retrieval model. This article also defines the number of factors to be used for the intelligent and efficient retrieval. The criteria that focus of information retrieval identifies efficient and intelligent retrieval. The sub-factors discussed are retrieval, relevancy, ranking and the layout. The concepts or situations for successful retrieval can be identified for the implementation. The experimental observations are properly explored with the particular emphasis on the necessities of the successful retrieval using analytical hierarchy process.

Keywords: Analytical Hierarchical Process, Information Retrieval, Efficient data retrieval, Intelligent Data retrieval, Relevancy, Layout, retrieval, Ranking.

1 Introduction

Information Retrieval implementation is a process involving factors namely the data, the user and the system. The previous work shows strength of the relationships between the factors for retrieval. The framework, however, does not reflect the prioritisation of these factors. These factors are equally important and showing the two way relationships between each other. The aim is to gain insight into the essential factors influencing Information Retrieval; a model containing minimum selected theoretical constructs has been developed and empirically tested. There have been three major concerns, upon building the research model. First, the model does not propose delineation of all the variables or processes that affect Information Retrieval.

Second, focus has been on efficient and intelligent retrieval as the leading expression of Information Retrieval, among development of semantic web. Third,

system, data and user in the model, have been perceived to affect Efficient Data Retrieval and Intelligent Data Retrieval. Therefore, the model highlights a few key factors that can explain prioritization of efforts towards successful retrieval initiatives. The proposed evaluation model shows cause and effect links between system, data and user, its components with Efficient Data Retrieval and Intelligent Data Retrieval, which in turn is associated with successful retrieval. The empirical evaluation model illustrates hierarchical relationships among all the variables. One of the techniques used to evaluate hierarchical relationships for ascertaining priorities amongst factors is the Analytic Hierarchy Processing (AHP).

2 Analytic Hierarchy Processing (AHP) Technique

The AHP is a decision approach followed in the solution of complex multiple criteria problems in a number of application domains. In view of the complexity and uncertainty of decision-situations, which almost all retrieval problems are prone to it, it becomes very important that the problems are micro-managed and that a hierarchy of means-ends is created so that the critical factors impacting performance are duly taken care of. It is here that AHP can be an important aid in decision-making. The important advantages of AHP are its simplicity, robustness and the ability to incorporate both ‘tangibles’ and ‘intangibles’ into the decision-making process. The AHP proposed in this research is to handle factors and sub-factors affecting successful Information Retrieval. The decision maker judges the importance of each criterion in pair-wise comparisons. The outcome of AHP is the weight of each decision alternative. Three steps are followed for considering decision problems by AHP namely constructing hierarchies, comparative judgment, and synthesis of weights.

2.1 Structural Hierarchy

This step allows a complex decision to be structured into a hierarchy descending from an overall objective to various ‘criteria’, ‘sub-criteria’, and so on, until the lowest level. The objective or the overall goal of the decision is represented at the top level of the hierarchy. The criteria and sub-criteria contributing to the decision are represented at the intermediate levels. Finally, the decision alternatives or selection choices are laid down at the last level of the hierarchy. A hierarchy can be constructed by creative thinking, recollection, and using people's perspectives. There is no set of procedures for generating the levels to be included in the hierarchy [1]. The structure of the hierarchy depends upon the nature or type of managerial decisions. Also, the number of the levels in a hierarchy depends on the complexity of the problem being analyzed and the degree of detail of the problem that an analyst requires to solve. As such, the hierarchy representation of a system may vary from one person to another.

2.2 Comparative Judgments

Once the hierarchy has been structured, the next step is to determine the priorities of elements at each level ('element' here means every member of the hierarchy). A set of comparison matrices of all elements in a level of the hierarchy with respect to an element of the immediately higher level are constructed so as to prioritize and convert individual comparative judgments into ratio scale measurements. The preferences are quantified using a nine-point scale [1]. The pair-wise comparisons are given in terms of how much more element 'A' is important than element 'B'. The synthesis of Priorities and the Measurement of Consistency is the pair-wise comparisons generate a matrix of relative rankings for each level of the hierarchy. The number of matrices at each level depends on the number of elements at the immediate upper level that it links to. The order of the matrix depends on the number of elements at each level. After developing all matrices and obtaining all pair-wise comparisons, eigenvectors or the relative weights (the degree of the relative importance amongst the elements), global weights, and the maximum eigenvalue (λ_{\max}) for each matrix are calculated. The (λ_{\max}) value is an important validating parameter in AHP. It is used as a reference index to screen information by calculating the consistency ratio (CR) of the estimated vector in order to validate whether the pair-wise comparison matrix provides a completely consistent evaluation. The consistency ratio is calculated as per the following steps:

- Calculate the eigenvector or the relative weights and (λ_{\max}) for each matrix of order n.
- Compute the consistency index (CI) for each matrix of order n as:

$$CI = (\lambda_{\max} - n) / (n - 1)$$

- The consistency ratio is then calculated using the formulae

$$CR = CI / RI$$

where RI is a known random consistency index obtained from a large number of simulation runs and varies depending upon the order of matrix (Exhibit 1).

The acceptable CR value for a matrix at each level is 0.1. If CI is sufficiently small, decision maker's comparisons are probably consistent enough to give useful estimates of the weights for the objective functions. If $CI/RI \leq 0.10$, the degree of consistency is satisfactory, but if $CI/RI > 0.10$, serious inconsistencies may exist, the AHP may not yield meaningful results and the evaluation process should therefore be reviewed, and improved.

Exhibit 1. Random index values

Order of Matrix	2	3	4	5	6	7	8	9	10
RI	0	0.58	0.9	1.12	1.24	1.32	1.41	1.45	1.51

Exhibit 2. Fundamental scales for pair-wise comparisons

Verbal Scale	Numerical Values
Equally important, likely or preferred	1
Moderately more important, likely or preferred	3
Strongly more important, likely or preferred	5
Very strongly more important, likely or preferred	7
Extremely more important, likely or preferred	9
Intermediate values to reflect compromise	2, 4, 6, 8
Reciprocals for inverse comparison	Reciprocals

In the present study for the pair-wise comparison, we have relied on actual data, that is, the data extracted from the survey. The question for pair-wise comparison of quantitative criteria is considered as:

“Of two elements i and j , how many times i is preferred to j ”

If the values for the criteria i and j are, respectively, w_i and w_j , the preference of the criteria i to j is equal to w_i/w_j . Therefore, the pair-wise comparison matrix is

$$\begin{pmatrix} w_1/w_1 & w_1/w_2 & \cdots & w_1/w_n \\ w_2/w_1 & w_2/w_2 & & w_2/w_n \\ \cdots & & & \\ w_n/w_1 & w_n/w_2 & & w_n/w_n \end{pmatrix}$$

As this matrix is consistent [1], the weight of each element is its relative normalized amount, i.e.

$$\text{weight of the } i-th \text{ element} = \frac{w_i}{\sum_{i=1}^n w_i}$$

The priority of criteria i to j for negative criteria, such as risk, is equal to w_j/w_i . The pair-wise comparison matrix is therefore

$$\begin{pmatrix} w_1/w_1 & w_2/w_1 & \cdots & w_n/w_1 \\ w_1/w_2 & w_2/w_2 & & w_n/w_2 \\ \cdots & & & \\ w_1/w_n & w_2/w_n & & w_n/w_n \end{pmatrix}$$

The above matrix is also consistent (Saaty 2000). The weight of the $i-th$ element is given by

$$\frac{1}{\sum_{i=1}^n \frac{w_i}{w}}$$

The data so collected were analyzed with the AHP techniques to arrive at weights (priorities).

3 Proposed AHP Model of Successful Information Retrieval

Any search engine's implementation requires being aware of and having the information about the issues/problem. Similarly, in case of Information Retrieval, it is important to have information about the influential factors for the implementation of Information Retrieval. Not all of the influential factors are equally important for the Information Retrieval. For this reason we have used the AHP frame work for finding the importance of the influential factors. AHP has been widely used as an analytical tool for decisions related to Information Retrieval. Recent work by [2] in presenting an effectiveness measurement model for Information Retrieval using AHP is a contribution in this direction.

In the present study, the influential factors are determined via widespread investigations and consultations with various experts, and owner/developers of semantic web. The synthesizing the literature review from [3-5], the opinions of the experts are employed to obtain the two main factors: Efficient Data Retrieval and Intelligent Data Retrieval. From these factors, 11 influential sub-factors for the implementation of Information Retrieval are briefly described as follows:

3.1 Efficient Data Retrieval (C1)

For the implementation of Information Retrieval, it is important that owner/developers of semantic web make themselves aware of the associated benefits that capturing and storing the documents for retrieval. They should perceive that by formal Information Retrieval process, they might contribute to the growth of the research of field of information retrieval. The process of capturing, storing and retrieving is very challenging as the documents are increasing in every seconds for the purpose. To meet the challenge, it is important to work on the retrieval part of the system, that may be associated with searching and storing. To motivate the research in this area is to create and share the documents at WWW and the owner/developers need to work on the latest technology for the web searchers. The Efficient Data Retrieval as a construct is very complex and requires further decomposition for a better understanding. Accordingly, the Efficient Data Retrieval has been decomposed into three sub-factors:

- System (C11): Efficient Data Retrieval is likely to be influenced by retrieval relevancy, ranking and layout of the documents/components. It depends upon the Information Retrieval systems and uncertain return on the relevant documents or not.

The retrieval off coarsely depends on the system or the application that you are using .It also depends upon the user where the user can select the layout also of his/her choice. The retrieval documents depend on the user to user; relevant documents attempts to focus at the high rank. These tend to be one of the important parameters for the successful implementation of Information Retrieval. The precision and recall measures for the evaluation of efficient and intelligent retrieval

- Data (C12): The retrieval of relevant documents depend on the user-to-user; relevant documents attempts to focus at the high rank. These tend to be one of the important parameter for the successful implementation of Information Retrieval. The data to be organized at the backend is very important to be taken care off.

- User (C13): Efficient Data Retrieval can be enhanced if owner/managers view benefits directly associated with Information Retrieval in terms of decrease in operational application, user layout, process improvement, and user satisfaction.

From the critical investigation of the associated with Intelligent Data Retrieval, the following three sub factors for system, data and user can be culled out. The sub-factor C11 further includes four sub-factors, C111: relevancy: C112: retrieval C113: ranking and C114: layout. Similarly the sub-factor C12 further includes four sub-factors, C121: relevancy: C122: retrieval C123: ranking and C124: layout. Also, sub-factor C13 includes three sub-factors, C131: relevancy: C132: retrieval C133: ranking and C134: layout.

3.2 Intelligent Data Retrieval (C2)

Intelligent Data Retrieval of Information Retrieval is very important. This Information Retrieval would involve assessment of document needs, identification on basis of users or people, distribution and reuse of features within the organization. The Intelligent Data Retrieval of Information Retrieval shall elements in terms of terms of successful retrieval. Intelligent Data Retrieval of Information Retrieval has been decomposed into three sub-factors:

- System (C21): Intelligent Data Retrieval is likely to be affected by implementation of information retrieval system out of fallacies in assessing documents needs and lack of involvement of user in the process. Lack of appropriate information about the user and also the advanced technology issues. Moreover, obsolescence of technical infrastructure can also hinder Information Retrieval Intelligent Data Retrieval.

- Data (C22): Intelligent Data Retrieval is likely to be affected by implementation of document/data objects information retrieval system. This takes care of the data structure how the data is kept and organized in a way of retrieval. This is a case where the user involvement is very much required for categorizing the documents with respect to the user. Moreover, obsolescence of technical infrastructure and inappropriate of the categories may hinder Information Retrieval Intelligent Data Retrieval.

- User (C23): User benefit from Information Retrieval Intelligent Data Retrievals in terms of creation of Information Retrieval repositories or groups that can

be put to perpetual use, person autonomy where they are enabled to handle unforeseen situations independently and also shared in groups. This attempts to solve the problems in a most creative manner. Intelligent Data Retrieval also fosters continuous innovations, and research and development activities. Increased use of documents leads to system effectiveness – relevancy, retrieval, ranking and layout. These users reinforce Information Retrieval Intelligent Data Retrievals in the application.

From the critical investigation of the associated with Intelligent Data Retrieval, the following three sub factors for system, data and user can be culled out. The sub-factor C21 further includes four sub-factors, C211: retrieval; C212: relevancy C213: ranking and C214: layout. Similarly the sub-factor C22 further includes four sub-factors, C221: relevancy; C222: retrieval C223: ranking and C224: layout. Also, sub-factor C23 includes three sub-factors, C231: relevancy; C232: retrieval C233: ranking and C234: layout

According to the AHP methodology, weights (priorities) can be determined using a pair-wise comparison within each pair of factors. To determine the relative weights, owner/managers can be asked to make pair-wise comparisons using a 1–9 preference scale [6]. However, in the present study for the pair-wise comparison, there is a reliance on actual data, that is, the data extracted from the questionnaire survey. The advantage of using actual data (quantitative data) over preference scale for pair-wise comparison eliminates the need for consistency checks [7].

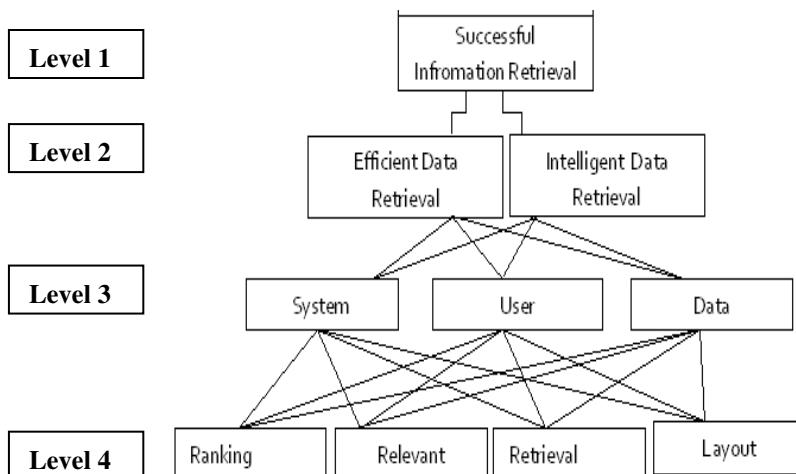


Fig. 1. Hierarchical Model to Successful Information Retrieval

As discussed above for the present study reliance has been on quantitative data as obtained from the questionnaire survey of small and medium sized enterprises. As has been mentioned in the questionnaire consisted of 20 questions in 5-point Likert scale divided into 4 sections namely-relevancy, retrieval, ranking, and layout. The following procedure has been adopted on the collected questionnaire survey data for the pair-wise comparison in AHP methodology. Firstly, the average value of 100

responses (preferences based on 5-point Likert scale) obtained for each question has been calculated. These average values were calculated to describe the central location of an entire distribution of responses. Then for every said category the Composite Preference Value (out of 5) using the following relations have been calculated:

$$\text{Composite Preference Value (CPF)} = (\text{Calculated Value} / \text{Maximum Value}) \times 5$$

Where, Calculated value = sum of the average values for the questions considered in a category.

Maximum value = sum of the highest possible values that a respondent can choose for the questions considered in a category.

4 AHP Model Description

The comparison matrices showing the measure of each attribute's relative importance with respect to the overall objective are summarized in Exhibit 3. For the pair wise comparison of the attributes and sub attributes, there has been reliance on inputs obtained from the survey.

Two attributes have been considered as important for Information Retrieval namely, Efficient Data Retrieval and Intelligent Data Retrieval [8-10]. The picture emerges from the pair-wise comparison suggest for Information Retrieval, Efficient Data Retrieval (52.09%) is more important over Intelligent Data Retrieval (47.91%). Thus it is important to generate Efficient Data Retrieval about Information Retrieval and its benefits amongst owner/managers.

Exhibit 3. Pair-wise comparison w.r.t. Information retrieval

Successful Information Retrieval			
Factors	Efficient Data Retrieval	Intelligent Data Retrieval	Weights
Efficient Data Retrieval	1	1.087097595	0.5209
Intelligent Data Retrieval	0.919880611	1	0.4791

According to the hierarchical model considered in the present study, Efficient Data Retrieval and Intelligent Data Retrieval has been further decomposed into system, data and user for capturing reality. On pair-wise comparison of system, data and user corresponding to Efficient data retrieval; system (32.73%), data (34.54%) and user (32.73%) weigh almost equally. When the same system and user are compared pair-wise corresponding to Intelligent Data Retrieval, both system (29.66%), data (34.7%) and user (35.64%) also weigh almost equally that means with better understanding about Information Retrieval, one can distinctly assess the system, data and user.

System and user corresponding to Efficient Data Retrieval and Intelligent Data Retrieval are of four types, viz. retrieval, relevancy, ranking and layout. The picture that emerges in pair wise comparison of the said types corresponding to system (Efficient Data Retrieval); retrieval (21.25%), relevancy(22.23%), ranking(27.64%) and layout (28.86%) is the most important component of data in comparison to

Exhibit 4. Pair-wise comparison w.r.t. Parent factor Efficient Data Retrieval

Efficient Data Retrieval					
Sub-factors	System	Data	User	Sum	Weights
System	1	0.96605039	0.96605039	2.932	0.3273
Data	1.0351427	1	1.05922355	3.094	0.3454
User	0.9872676	0.94408777	1	2.931	0.3272

Exhibit 5. Pair-wise comparison w.r.t. Parent factor Intelligent Data Retrieval

Intelligent Data Retrieval					
Sub-factors	System	Data	User	Sum	Weights
System	1	0.74497773	0.96160725	2.707	0.2966
Data	1.342322	1	0.8245471	3.167	0.347
User	1.0399256	1.212787	1	3.253	0.3564

retrieval (23.37%), relevancy(24.07%), ranking(27.06%) and layout (28.86%). This means that owner/developers of semantic web fare more concerned with financial losses involved in Information Retrieval over the fear of technical implementations and human attritions.

Exhibit 6. Pair-wise comparison w.r.t. System (Efficient Data Retrieval)

System (Efficient Data Retrieval)						
Sub-factors	Retrieval	Relevancy	Ranking	Layout	Sum	Weights
Retrieval	1	0.94846665	0.78232978	0.731	3.462	0.21254352
Relevancy	1.0543333	1	0.82483636	0.743	3.6223	0.222238592
Ranking	1.2782333	1.21236168	1	1.012	4.5031	0.27646105
Layout	1.367645	1.3456652	0.9876587	1	4.701	0.28860952

Similarly, in comparison of the said types corresponding to user(Intelligent data retrieval); and user at retrieval(27.97%), relevancy(28.52%), ranking(19.44%) and layout(24.04%). It implies that owner/developers of semantic web consider investments in technology as the ultimate solution to all problems. Similarly, in comparison of the said types corresponding to user (efficient data retrieval), relevancy dominates(28.52%) dominates the other types, i.e. ranking(19.44%), retrieval (27.97%) and layout(24.04%). It implies that owner/developers of semantic web consider investments in technology as the ultimate solution for relevancy.

Exhibit 7. Pair-wise comparison w.r.t. Data (Efficient Data Retrieval)

Data (Efficient Data Retrieval)						
Sub-factors	Retrieval	Relevancy	Ranking	Layout	Sum	Weights
Retrieval	1	0.9783	1.0692	0.783	3.8307	0.2337
Relevancy	1.0222	1	0.6175	1.306	3.9458	0.2408
Ranking	0.9353	1.6193	1	0.881	4.4361	0.2707
Layout	1.2767	0.7656	1.1345	1	4.1769	0.2549

Further, in pair wise comparison of the said types corresponding to system (Intelligent Data Retrieval); ranking (26.74%) and layout (29.21%) dominates the other types, i.e. retrieval (21.51%) and relevancy (22.51%). This means that owner/developers of semantic web are more concerned with investments in ranking and layout involved in Information Retrieval over the loss of retrieval and relevancy implementation.

Exhibit 8. Pair-wise comparison w.r.t. User (Efficient Data Retrieval)

User (Efficient Data Retrieval)						
Sub-factors	Retrieval	Relevancy	Ranking	Layout	Sum	Weights
Retrieval	1	0.9783	1.36	1.252	4.5905	0.2798
Relevancy	1.0222	1	1.3902	1.268	4.6805	0.2853
Ranking	0.7353	0.7193	1	0.736	3.1907	0.1945
Layout	0.7986	0.7885	1.3586	1	3.9457	0.2405

Exhibit 9. Pair-wise comparison w.r.t. System (Intelligent Data Retrieval)

System(Intelligent Data Retrieval)						
Sub-factors	Retrieval	Relevancy	Ranking	Layout	Sum	Weights
Retrieval	1	0.94846665	0.78232978	0.731	3.462	0.21518573
Relevancy	1.0543333	1	0.82483636	0.743	3.6223	0.22515048
Ranking	1.0782333	1.21236168	1	1.012	4.3031	0.26746647
Layout	1.367645	1.3456652	0.9876587	1	4.701	0.29219733

On the contrary, when the same said types are compared pair wise corresponding to data (Intelligent Data Retrieval); ranking (26.95%) and layout (29.63%) are more important than component of retrieval (22.92%) and dominates the other type, i.e. relevancy (20.48%). This means that owner/developers of semantic web are prepared to rank and layout for sharing retrieval for the benefit of the web searches.

Exhibit 10. Pair-wise comparison w.r.t. data (Intelligent Data Retrieval)

Data (Intelligent Data Retrieval)						
Sub-factors	Retrieval	Relevancy	Ranking	Layout	Sum	Weights
Retrieval	1	0.9783	1.0692	0.7264	3.7738	0.2293
Relevancy	1.0222	1	0.6175	0.7323	3.372	0.2049
Ranking	0.9353	1.6193	1	0.8814	4.4361	0.2695
Layout	1.3767	1.3656	1.1345	1	4.8769	0.2963

On the contrary, when the same said types are compared pair wise corresponding to user (Intelligent Data Retrieval); layout (30.62%) dominates the other types retrieval (24.59%) and relevancy (25.41%) and ranking (19.36%) are almost equally important component of reward and dominates the other type, i.e. layout (30.62%). This means that owner/developers of semantic web prepared to layout for using the retrieval system for the benefit of the search user.

Exhibit 11. Pair-wise comparison w.r.t. User (Intelligent Data Retrieval)

User(Intelligent Data Retrieval)						
Sub-factors	Retrieval	Relevancy	Ranking	Layout	Sum	Weights
Retrieval	1	0.9783	1.36	0.715	4.0533	0.246
Relevancy	1.0222	1	1.3902	0.7761	4.1884	0.2542
Ranking	0.7353	0.7193	1	0.7361	3.1907	0.1936
Layout	1.3986	1.2885	1.3586	1	5.0457	0.3062

In what follows next, is using the bottom up approach to get the global relevance of retrieval, relevancy ranking and layout aspects towards Information Retrieval.

Exhibit 12. Overall weight of system, data and user sub factors w.r.t. Information Retrieval

Successful Information Retrieval			
Sub-factors	Efficient Data Retrieval	Intelligent Data Retrieval	Weights
System	0.327322942	0.296574134	0.311948538
Data	0.345437329	0.34700978	0.346223555
User	0.32723973	0.356416086	0.341827908

Towards this we multiply the local relevance of retrieval, relevancy ranking and layout corresponding to sub-attributes with the local relevance of the sub-attributes corresponding to its parent attribute. This is further multiplied with the local relevance

of the parent attribute corresponding to the overall objective. Finally, the obtained relevance's of retrieval, relevancy ranking and layout aspects corresponding to the main attributes, i.e. Efficient Data Retrieval and Intelligent Data Retrieval are added to get the global relevance. Finally, the picture emerges for the global relevance of retrieval, relevancy ranking and layout aspects incorporating relevance of the main attributes i.e., Efficient Data Retrieval and Intelligent Data Retrieval as well as their sub-attributes system, data and user; data (34.62%) and user (34.18%) is most important followed as compare to system (31.19%). Thus, the key to Information Retrieval rests on appropriate Efficient Data Retrieval and Intelligent Data Retrieval data and user. In other words, owner/developers of semantic web need to prioritize their efforts towards Information Retrieval in terms of retrieval, relevancy, ranking and layout aspects necessarily in that order.

To summarize it is evident, layout aspects of Information retrieval, viz., and Information retrieval management account for the highest global weight in the Information Retrieval decision analytical hierarchy developed here. The data and user aspects of Information Retrieval, viz., are the main concern of the data storage and reuse the data for different aspects of retrieval. The creation and sharing of documents and objects weigh next and the retrieval aspects, viz., in efficient and intelligent data retrieval, etc. are of relatively less importance in determining success of the Information Retrieval endeavour in the perception of the respondents of the study.

Exhibit 13. Overall weights of retrieval, relevancy, ranking and layout sub factors w.r.t. Information Retrieval

Information Retrieval			
Sub-factors	Efficient Data Retrieval	Intelligent Data Retrieval	sum
Retrieval	0.2420179	0.23015113	0.23608451
Relevancy	0.2494697	0.22806934	0.23876954
Ranking	0.2471982	0.24354224	0.24537023
Layout	0.2613141	0.29823729	0.27977572

The retrieval aspects would, therefore, are better advised to fortify Information Retrieval processes, more so those pertaining to the planning, implementing and monitoring the layout or web development, to enhance of retrieval assets or the web embodied features in plug-in and agents for the better outcomes. The successful information retrieval is now needed to develop web industry-wide for conduct and ethics in order to storage and reuse the documents available in World Wide Web.

5 Conclusion

The successful efficient and intelligent retrieval is based on many factors. Some of the important factors are considered here. Among User, data and System it is found that user and system dependent factors are more important. Also for the successful retrieval ranking (priority results) and the layout of retrieval results are more

important. It is found to be very minor difference, it means that all the factors are equally important and cannot be ignored. AHP identifies the factors and the sub-factors as important or not for the further consideration of efficient and intelligent retrieval.

References

1. Saaty, T.L.: Fundamentals of decision making and priority theory with the AHP, 2nd edn. RWS, Pittsburg (2000)
2. Cheng, E.W.L., Li, H.: Analytic hierarchy process: an approach to determine measures for business performance. *Measuring Business Excellence* 5(3), 30–36 (2001)
3. Brin, S., Page, L.: The Anatomy of a Large-Scale Hyper textual Web Search Engine. In: Proc. of the 7th International World Wide Web Conference, pp. 107–117 (1998)
4. Griffiths, J.R.: Evaluation of the JISC Information Environment: Student perceptions of services. *Information Research* 8(4) (2003), <http://informationr.net/ir/8-4/paper160.html> (accessed June 11, 2009)
5. Johnson, F.C., Griffiths, J.G., Hartley, R.J.: "DEVISE: a framework for the evaluation of internet search engines", Research Report, No. 100, Library and Information Commission, The Council for Museums Archives and Libraries (2001),
<http://www.cerlim.ac.uk/projects/devise/devise-report.pdf> (accessed June 11, 2009)
6. Saaty, T.L.: The Analytic Hierarchy Process. McGraw-Hill, New York (1980)
7. Saaty, T.L.: How to make a decision: the analytic hierarchy process. *Interfaces* 24(6), 19–43 (1994)
8. Chatzoglou, P.D., Diamantidis, A.D.: IT/IS implementation system and their impact on firm performance. *International Journal of Information Management* 29, 119–128 (2009)
9. Kleinberg, J.M.: Authoritative Sources in a Hyperlinked Environment. *Journal of the ACM* 46(5), 604–632 (1999)
10. Arora, M., Kanjilal, U., Varshney, D.: Successful Efficient and Intelligent Data Retrieval: Using Analytic Hierarchy Process (AHP). In: *Handbook of Management and Behavioural Science*, vol. 1, pp. 109–116. Wisdom Publication, India (2011)
11. Cao, L., Gorodetsky, V., Mitkas, P.: Agent Mining: The Synergy of Agents and Data Mining. *IEEE Intelligent Systems* 24(3), 64–72 (2009)

A Hybrid System Based on Multi-Agent Systems in Case of e-WeddingThailand

Kunyanuth Kularbphettong¹, Phayung Meesad², and Gareth Clayton²

¹ Science and Technology Faculty, Suan Sunandha Rajabhat University, Bangkok, Thailand
kobkulriss@gmail.com

² Information Technology Faculty, King Mongkut's University of Technology North Bangkok,
Bangkok, Thailand
{Pym, gareth}@kmutnb.ac.th

Abstract. We describe the usage of the Multi-agent system in the on-going project, called e-WeddingThailand. The aim of this project is to utilize MAS and various approaches, like Web services, Ontology, and Data mining techniques, in e-Business that want to improve responsiveness and efficiency of systems so as to extract customer behavior model on Wedding Businesses. The multi agent system, which constitutes the backbone of the framework, connects these pieces together and makes them perform properly. JADE is the MAS platform implemented this project. JADE is quite easy to learn and use. Moreover, it supports many agent approaches such as agent communication, protocol, behavior and ontology.

Keywords: multi-agent system, Web services, ontology, data mining techniques, e-WeddingThailand, JADE.

1 Introduction

In recent years, Multi-Agents System (MAS) has become one of the most significant and useful contributions to Information Technology research. Considering the increasing needs of developing a high-level agent system for applications in e-commerce, decision support systems and Internet applications, thus there has been a considerable amount of research on multi-agent systems. A multi agent system is a computational system, or a loosely coupled network in which two or more agents interact or work together to perform a set of tasks or to satisfy a set of goals. Each agent is considered as a locus of a problem-solving activity which operates asynchronously with respect to the other agents[1]. However, most of the initial work devoted to MAS research has focused on closed systems, which are designed by one developer for one homogeneous environment, and one single domain[2]. In order to tackle the above problems there is a need for adaption and flexibility of real open systems that are capable of dynamically adapting themselves to a changing environment like electronic markets, communities and distributed search engines. Most participants are unknown beforehand and can change over time and can also be developed by different parties.

Therefore, in this project a framework is proposed for e-WeddingThailand business in order to overcome the limitations of the current MAS for open environments. This framework deals with construction material searching and recommending based on Multi-agents system and various techniques, as in Web services, Ontology and Data Mining techniques, so as to enhance the adaptability aspect of the system and to create highly flexible and dynamic systems. The wedding business is one of the crucial businesses that it becomes huge and still rapidly expanding. Booming of wedding business has also propelled the enhance of hotels, wedding studios, car hiring companies, flower shops, music, travel agencies, and even media businesses.

Due to overwhelming information widely spread on internet and real world environments, it is very time consuming for couple to search appropriate information. Moreover, almost successful e-Commerce systems are still handled by humans to make the important decisions. Therefore, with vastly developed advance mechanisms, in this paper we propose the framework of e-Wedding business applied various approaches like multi-agent, web services, ontology, and data mining techniques.

The remainder of this paper is organized as follows. Section 2 reviews about related literatures and research works in the use of multi-agent system blend with various techniques. Section 3 presents the related methodologies used in this work. Section 4 presents the experimental results based on the purposed framework based on multi-agent system. This project demonstrates how to success for adapting multi-agent system with various techniques. Finally, we conclude the paper with future research issues in section 5.

2 Related Works

A literature search shows that most of the related researches have deployed multi-agent to develop other fields with various techniques by following this: According to Bala and Majigsuren[3], they showed a prototype of the system using the JADE platform in the context of travel industry. DragonChain[4] was an implementation application to simulate a supply chain management system developed by the University of Pennsylvania from USA. It could reduce a bullwhip effect from the Beer Game, the MIT Beer Game and the Columbia Beer Game. The project used agents to look for the best ordering scheme by genetic algorithms. Agent Building Shell[5] was a reusable software components collection as well as interfaces necessary for any agent in a supply chain management system. The ABS was used to maintain perturbations from stochastic events in a supply chain system. MetaMorph[6] was an adaptive agent-based architecture for addressing system adaptation also extended-enterprise issues in four levels as virtual enterprise, distributed intelligent systems, concurrent engineering, and agent architecture. NetMan[7][8] was abbreviated from Networked Manufacturing. It was used to create manufacturing networks in a dynamic environment. BPMAT & SCL[9] or Business Process Modeling and Analysis Tool were a software library for supporting the company activities modeling. MASCOT[10] was an agent-based architecture for scheduling and planning in multilevel and reconfigurable way that was used as a supply chain agility improvement and used to evaluates new product designs as well as any strategies for business decisions such as supplier selection decisions.

DASCh[11] was the modeling techniques exploration for suppliers' networks. The items and information flow of this system are expressed as agents to form imperfections in the flows; Task Dependency Network[12] was used to allocate also schedule any tasks among agents contending for scarce resource. As well, it was constrained by a hierarchical task dependency network. MASC[13] was relational modules between supply chains among the companies. OCEAN[14] was short word from Organization and Control Emergence with an Agent Network. The designation of the system is to react for environment dynamics to view that the global level association may emerge from competitions at the local level. aKIA[15] is the MASs that affords the precise and competent search including retrieve items information along with the user's desire. PKI Authentication[16] is the MAS application that is used to maintain multi-clients authentication. It is basically based on the public key infrastructure (PKI) authentication method including the multi-agent technique and According to Andrey[17], MAS for Taxi Companies involves calling taxis from customers in different locations. This transmission system presently in use by a main taxi company divides the city (an operation of the system) into local areas. Each area has fixed selected nearby areas hand-coded by human specialists. The system chooses nearby areas for searching.

Moreover, other researchers propose an agent-based framework representing in various ways. For instance, Negri et al[18] integrated the agent technology with other key emerging technologies like semantic Web, Web service, rule engine and workflow technologies and KODAMA[19] is another system based on a multi-agent system and leveraged the Semantic Web services.

3 The Methodologies

In this section, we illustrate the specified methodologies used in this project including Multi-Agent System, Web Services, Ontology, and Data mining Techniques.

3.1 Multi-Agent System

A multi-agent system (MAS) can be defined as a collection of autonomous agents that interact with each other to coordinate their activities in order to solve collectively a problem that cannot be tackled by any agent individually[20][21]. According to Stone and Veloso[22], MAS was derived from the traditional Distributed Artificial Intelligence (DAI). DAI is the combination of distributed system and artificial intelligence. MAS emphasizes the joint behavior of agents with some degree of autonomy and the complexities rising from their interactions[23]. Also, Padghan and Winikopff[24] described that the concept of Agent refers to an entity acting on behalf of other entities or organizations and having the ability to perceive relevant information, and following and handling the objectives to be accomplished. Also, in open and dynamic environment like internet, a multi-agent system is one of the promising means to help reduce cost, increase efficiency, reduce errors and achieve optimal deal.

There are two issues related to the design of MAS: Agent Communication Language and agent development platform. The former concerns with the message

interchange between different agent such as KQML, and FIPA ACL. The latter is related with the platform development to provide an effective framework, such as IBM Aglets, ObjectSpace Voyager and etc, for the dispatching, communications, and management of multiple agents in the open and dynamic environment.

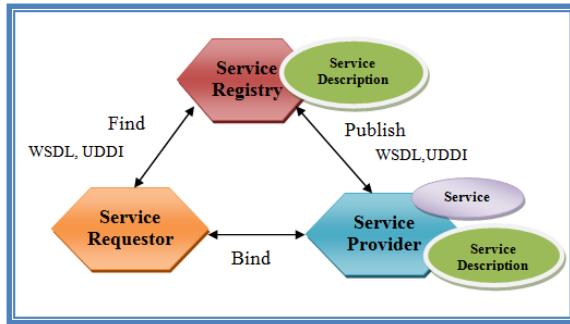
For this proposed project, JADE (Java Agent Development Framework) will be deployed as the prototype development tool. JADE (Java Agent Development Framework) is a software environment fully implemented in JAVA language aiming at the development of multi-agent systems that comply with FIPA specifications[25]. The goal of JADE is to simplify development while ensuring standard compliance through a comprehensive set of system services and agents. Each running instance of the JADE runtime environment is called a container as it can contain several agents. The set of active containers is called a platform. A single special container must always be active in a platform and all other containers register with it as soon as they start. Hence, the development framework based on JADE is considered very suitable for implementing applications that require distributing computation tasks over the network.

3.2 Web Services

Web Services are the services in the shape of software components accessible on the Internet, which provide useful information to users, businesses and organizations. These services model use WSDL[26], an XML format responsible for the service interfaces description along with the binding details to specific protocols, *UDDI*, which is a protocol responsible for publishing, finding services, services details and descriptions etc. *SOAP*[27] is an XML message based envelop format which has the bindings to specific protocols (e.g. HTTP, SMTP etc). The UDDI directory retrieves the WSDL description. WSDL descriptions are responsible for allowing the software systems to directly use and to extend the business to those of others. These services are invoked over the www (World Wide Web) using the SOAP/XMLP protocol.

The Web Services architecture is based on the interactions between three roles: service provider, service registry and service requester. The interactions involve with publish, find and bind operations. Together, these roles and operations act upon the Web Services artifacts: the Web Service software module and its description. In a typical scenario, a service provider hosts a network accessible software module an implementation of a Web Service.

The service provider defines a service description for the Web Service and publishes it to a service requester or service registry. The service requester uses a find operation to retrieve the service description locally or from the service registry and uses the service description to bind with the service provider and invoke or interact with the Web Service implementation. Service provider and service requester roles are logical constructs and a service can exhibit characteristics of both, as shown in Figure 1.

**Fig. 1.** Service oriented architecture

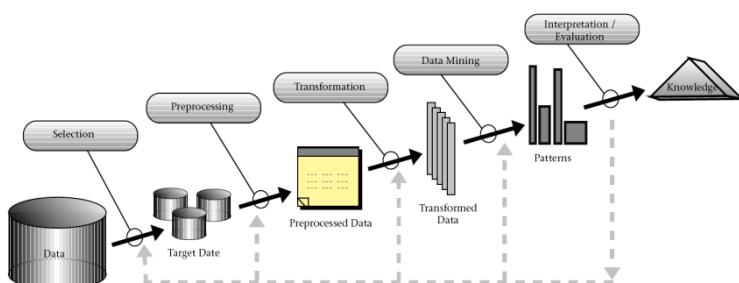
3.3 Ontology

Ontology has become a popular feature in many appliances in order to provide a semantic framework for knowledge management. Ontology refers to a content representing specific knowledge primitives (classes, relations, functions and constants). Ontology stands for the hierarchical knowledge structure about things by subcategorizing them by their essential qualities.

There are two kinds of ontology in the communication model, kqml-ontology and negotiation-ontology. Kqml ontology has been defined formally. We can find the OWL version from The DARPA Agent Markup Language web. Negotiation ontology is based on the idea that there are some general concepts that are present in any negotiation, and builds on finding commonalities across different negotiation protocols [28].

3.4 Data Mining Techniques

Data Mining is the data analyzing process from different perspectives also summarizing the useful information results. The data mining process uses many principles as machine learning, statistics and visualization techniques to discover and present knowledge in an easily comprehensible form. There is another definition [29] [30] as “the nontrivial process of identifying valid, novel, potentially useful, and ultimately understandable patterns in data”.

**Fig. 2.** The data mining steps[31]

In Figure 2 explained the data mining process, from this project we used clustering and classification techniques to handle data in this project, including EM algorithm, K-Mean, Support Vector Machine, Decision Tree Technique.

Expectation Maximization (EM) Algorithm

EM is a well known algorithm that attempts to maximize the likelihood of the model. EM models the distribution of instances probabilistically, so that an instance belongs to a group with a certain probability. EM calculates densities and the assumption is made by EM that the attributes are independent random variables. EM can handle both numeric and nominal attributes [32][33].

K-means

K-means is one of the simplest unsupervised learning algorithms for clustering data. The procedure follows a simple and easy way to classify a given data set through a certain number of clusters (assume k clusters) fixed a priori[34]. This algorithm aims at minimizing an *objective function*, in this case a squared error function. The objective function

$$j = \sum_{j=1}^k \sum_{i=1}^x \|x_i^{(j)} - c_j\|^2 \quad (1)$$

where $\|x_i^{(j)} - c_j\|^2$ is a chosen distance measure between a data point $x_i^{(j)}$ and the cluster centre c_j , is an indicator of the distance of the n data points from their respective cluster centers

Support Vector Machines

SVMs (Support Vector Machines) are a useful technique for data classification technique. The goal of SVM is to produce a model (based on the training data) which predicts the target values of the test data given only the test data attributes[35]. Also, Support Vector machines can be defined as systems which use hypothesis space of a linear functions in a high dimensional feature space, trained with a learning algorithm from optimization theory that implements a learning bias derived from statistical learning theory[36].

Decision Tree

Decision trees are one of the most widely used and practical algorithms of machine learning and data mining. Decision tree models are built by a process that is known as recursive partitioning. There are two significant steps to handle as follows:

First, the original data, also known as the root node, is broken up into 2 or more non-overlapping sub-samples, referred as nodes. The partitioning is done based on one of the independent variables known as the splitting attribute. Branches are drawn for different values of this splitting attribute. Each instance in the root node is sent down one of the branches (depending on its value for the splitting attribute) into one

of the nodes. The choice of splitting attribute is done by picking the attribute that will partition the original sample into sub-samples that are as homogenous as possible in relation to the class variable. It is a similar idea when choosing on what values of the splitting attribute to perform the partitioning. Second, the process is iterated for each of the nodes created in the first step. Each node is partitioned by considering only the cases in that node and new nodes are created from instances in that node alone. This process is repeated for each node until some stopping-rule is violated. When this happens, the node is not further partitioned and this node is referred to as leaf node. The whole process is terminated when there are only leaf nodes left.

4 The Proposed Framework and Implementation

This section presents the proposed framework of Multi-Agents System (MAS). In order to design and develop a web based application for our project we selected the JADE agent platform and various techniques to develop the e-WeddingThailand based on MAS. This section is organized as follows: Section 4.1 explains the Proposed Framework of Multi-Agent System. Section 4.2 presents the Implementation of the Proposed Framework.

4.1 The Proposed Framework of Multi-Agent System

The validation of our framework was done in the e-Commerce domain particularly in the wedding planning businesses. The wedding planning businesses is getting bigger and better; it can be a lucrative market for several supplier areas, as in the hotel food and beverage, wedding studio, travel agencies and etc. In recent years, the market needs for a wedding planning business are strongly shaped by the customers' desire to have a perfectly planned and executed wedding ceremony. With the growth of the internet and technologies supporting projects that are adapted to MAS, a wedding business based on web applications should be very beneficial. In order to enhance a wedding planning business and to support the decisions of couples Figure 3 shows the activity among multi-agent systems in a real environment.

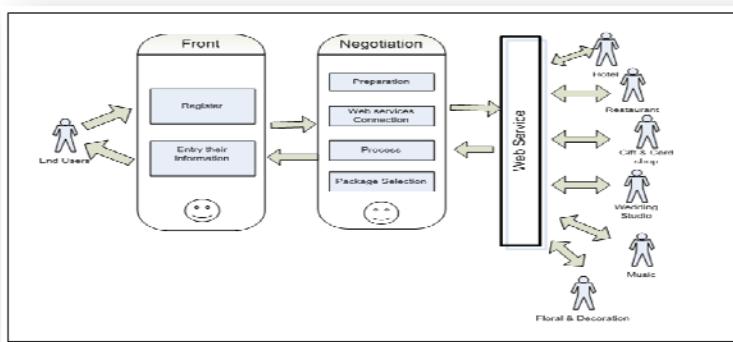


Fig. 3. The Activity among Multi-Agent System in a Real Environment

An e-WeddingThailand based on MAS, users visit the wedding web page, registers as a member and fills in his/her preferences that include engagement date, engagement place, wedding date, wedding place, expected guests on the engagement day, expected guests on the wedding day and their estimated budget.

After the user completes their preferences, the interface agent is initiated and acts on behalf of the user in order to contact with others. Then, the preparation agents will prepare data, calculate their preferences, connect with Web Service to find suppliers that propose their product to company, contact with supplier agent to find available products, find optimized packages that suitable with user preferences, purpose optimized packages to customers and etc.

The proposed e-WeddingThailand framework is a multi-agent system designed to cope with optimum wedding packages for couples relying on their preferences. This prototype system is divided to be two parts. First, a recommendation system for couples to find appropriated wedding packages and secondly takes responsibility of searching for a wedding business agency in order to get the optimal solution.

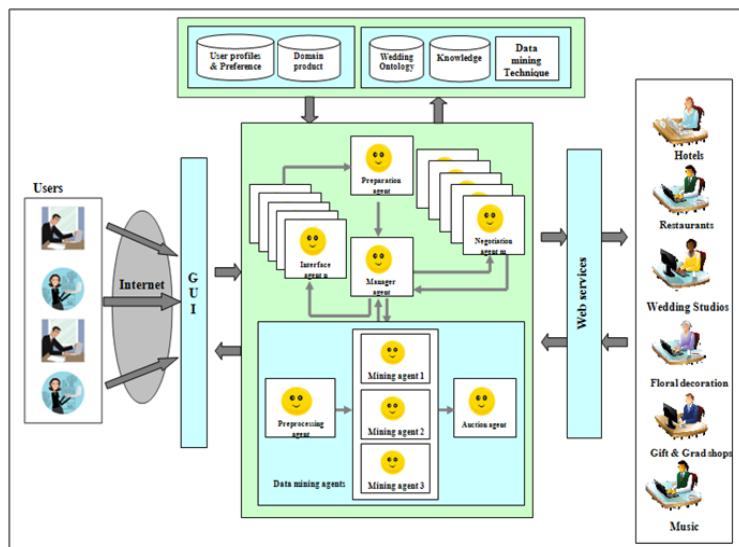


Fig. 4. The Purposed Architecture of e-WeddingThailand

In Figure 4, representation of the purposed architecture based on MAS, e-WeddingThailand. This is composed of four main elements: multi-agent system, Web services, ontology and data mining techniques. In the multi-agent system, each agent is autonomous to help decision making and act proactively. Agents can communicate, exchange knowledge, collaborate or negotiate with each other, to efficiently achieve a common goal. They receive and process users' requirements, make use of user's preference and perform as a mediator of services. They select and contact with appropriate wedding business agencies as in hotels, wedding studio, and so on through Web services to deliver couples the optimal result.

The web services can be defined as, providing related services, interacting with negotiator agents with meaningful data which can be directly accessed by agents or people through the Web. Each agent is identified to respond to a specific task. An interface agent is designed to assist couples and acts on behalf of a couple when using the e-WeddingThailand. A couple will fill their preference using an interface agent and pass through a preparation agent. Moreover, the interface agent observes a couple, adapts preference based on couple's requirement, and returns the result to that couple. Then a preparation agent responds to calculate the couples' budget, searches for an appropriate wedding business and then passes it through a manager agent. A manager agent will respond to communicate with a negotiator agent to send user information, to receive wedding packages, to look for and estimate suitable wedding packages based on user's profile and preference. Furthermore, it takes responsibility for contacting with the agents to purpose wedding packages suitable to that user. A negotiation agent carries out user's information and connects to web services so as to deliver information about wedding packages. Moreover it persuades its offer to wedding suppliers, evaluates value by comparing information with each other and sends it to a manager agent in order to evaluate the optimal solution and return the solution to an interface agent. Data mining agents in this purposed framework are separated into three parts: First, a preprocessing Agent will prepare data for mining to perform the necessary data cleaning, transforming and prepare data depended on specific data mining techniques. Secondly, mining agents implement specific data mining algorithms. Finally, auction agent get results from the mining agents, evaluates the optimum solution and passes it to the user.

4.2 The Implementation of the Proposed Framework

In order to demonstrate the effectiveness of the proposed framework and mechanisms, we develop and test the e-WeddingThailand project based on MAS as shown in Figure 5.



Fig. 5. Web Page of e-WeddingThailand

When a user gets to the web page, they must register as a member and fill in his/her preferences that include engagement date, engagement place, wedding date, wedding place, expected guests on the engagement day, expected guests on the wedding day and their estimated budget as depicted in Figure 6. The interface simulates and acts on behalf of the user.

Wedding Estimated Budget

ສະບັບຜົນ (Contact name):

ເລກທີ່ຜົນ (Contact number):

ອີເມວ (E-mail address):

ວິທະຍາໄນ (Wedding ceremony)

ວິທະຍາໄນ (Wedding Date):

ຄະຫຼາດວິທະຍາໄນ (Wedding Ceremony):

ຄວາມຕ້ອນຮັກ (The number of expected guests):

ຈຳນວດ (Wedding Estimated Budget): ແລ້ວ

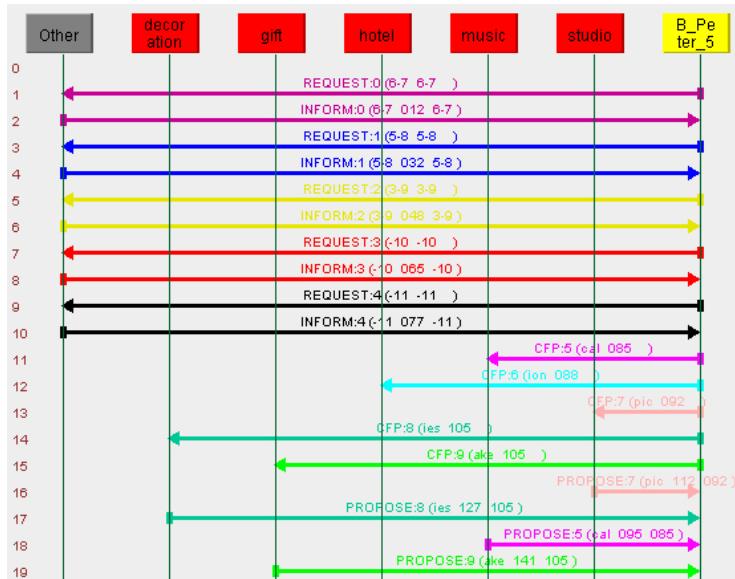
Fig. 6. A Web Page for the User Preference

Then, the preparation agent starts its task to calculate the estimated budget, as depicted in Figure 7, it will send an informed message to the manager agent. The manager agent sends requested messages to the negotiation agent so that the negotiation agents take responsibility to connect through Web Services in order to find suppliers that propose their products services to company, contacts supplier agents to find available products, find optimized packages that are suitable with user preferences and purpose optimized packages to customers.



Fig. 7. The Result of the MAS Preparation Data

Figure 8 displays the operation of agents in this system and shows the result of searched Products and services referring to user's requirement in Figure 9.

**Fig. 8.** The Result of the Agent's Operation**Fig. 9.** The Result of the Recommended Package

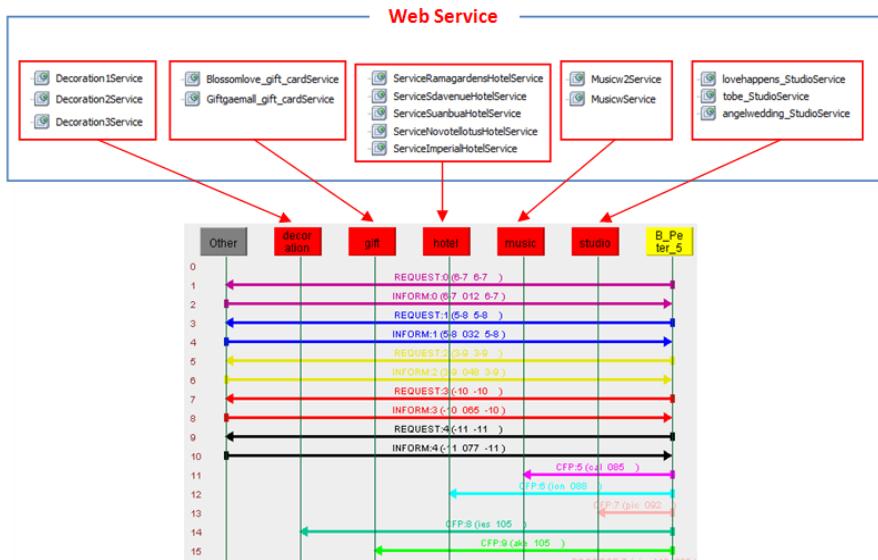


Fig. 10. The Processes of Agent's Operation with Web Services

In our project, we have divided our target segment according to income, budget and location. The four segment identified are the upper upper class (A++), the upper class (A+), the upper middle class (A) and the middle class (B+). The reason being that couples are always ready to shell out money for as grand and glorious wedding as possible. When discovering groups of similar users, two clustering technique are compared, simple K mean and EM clustering techniques as shown in Figure 11. For classification of the customer, we compared decision tree and SVM (support Vector Machine) classification algorithms. The result of the classification mining agent is shown in Figure 12. In addition, Figure 13 displays the operation of MAS throughout the mining process stages.

From Figure 11, the cluster analysis clearly indicates that these four segments are quite distinct, cohesive and compact. All basic information was used in profiling the four expected customer segments and a descriptive interpretation of these four market segments, together with descriptive labels for each group, is based on the average scores as given below.

For segment 1, the upper-upper class (A++) are established as wealthy, highest income, professionally employed, highly educated and highest incidence of home ownership; Segment 2, the upper class (A+) referring to professionals, above average income and above average education; Segment 3, is the upper-middle class (A) identified by average income and average education; and Segment 4, is the middle class (B+) who have a low income and below average education.

Agent Mining						
Cluster data						
EM				Simple K Mean		
RPraxed				LPMemorable	0.49	0.4211
mean	0.4136	0.638	0.5175	0.6207	0.619	0.3226
std. dev.	0.4925	0.4806	0.4997	0.4941		
RPNatural				LPOther	0.45	0.2632
mean	0.3623	0.3622	0.3388	0.4138	0.381	0.6452
std. dev.	0.4807	0.4806	0.4733	0.5238	0.4194	
RPClassic				RPCapturing	0.47	0.7368
mean	0.6828	0.2758	0.4917	0.3103	0.5238	0.6129
std. dev.	0.4654	0.4469	0.4999	0.4483	0.5238	0.6129
RPComments				RPTraditional	0.53	0.5263
mean	0.4556	0.5774	0.294	0.4828	0.6667	0.5806
std. dev.	0.498	0.494	0.4556	0.4762	0.6774	
				RPFunPictures	0.57	0.2632
				RPImages	0.53	0.4737
				RPGP	0.47	0.2105
				RPArtistic	0.38	0.4211
				RPClose-up	0.57	0.5263
				RPCandid	0.46	0.6316
				RPFashion	0.52	0.5263
				RPRelaxed	0.53	0.5789
				RPNatural	0.43	0.5263
				RPClassic	0.5	0.5263
				RPComments	0.44	0.3684
Clustered Instances				Clustered Instances		
0	22	(22%)		0	19	(19%)
1	16	(16%)		1	29	(29%)
2	36	(36%)		2	21	(21%)
3	26	(26%)		3	31	(31%)

Fig. 11. The Result of Clustering Techniques

Classified						
EM --> DTTree				Simple K Mean --> DTTree		
Correctly Classified Instances				Correctly Classified Instances	96	96 %
Incorrectly Classified Instances				Incorrectly Classified Instances	4	4 %
Kappa statistic				Kappa statistic	0.9458	
Mean absolute error				Mean absolute error	0.02	
Root mean squared error				Root mean squared error	0.1414	
Relative absolute error				Relative absolute error	5.4023 %	
Root relative squared error				Root relative squared error	32.8703 %	
Total Number of Instances				Total Number of Instances	100	
EM --> SVM				Simple K Mean --> SVM		
Correctly Classified Instances				Correctly Classified Instances	71	71 %
Incorrectly Classified Instances				Incorrectly Classified Instances	29	29 %
Kappa statistic				Kappa statistic	0.6033	
Mean absolute error				Mean absolute error	0.2858	
Root mean squared error				Root mean squared error	0.3634	
Relative absolute error				Relative absolute error	77.2079 %	
Root relative squared error				Root relative squared error	84.4721 %	
Total Number of Instances				Total Number of Instances	100	

Out put Auction : Simple K Mean -->DTTree

Fig. 12. The Result of Mining Techniques

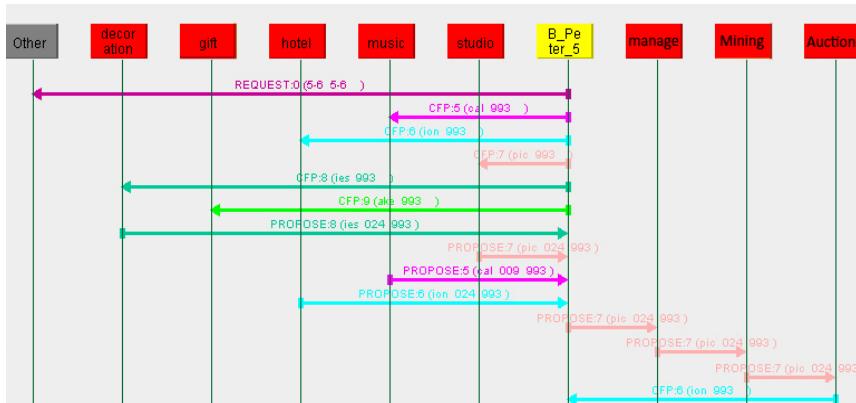


Fig. 13. The Operation of Agents in the Mining Process Stage

5 Conclusion and Future Work

The experiments of the Multi-Agent System with various techniques in e-Commerce have not only produced a successfully implemented and tested system but also acquired knowledge required to further develop or adapt to other e-business projects based on Multi-Agent System applications. Furthermore, the MAS model has been interesting and valuable for the personal knowledge of software development processes. As far as this project is concerned, Web Services Compositions and Customer Segmentation, based on the fact that no adequate and comparable Multi-Agent System has been implemented into a Web based application for wedding businesses in Thailand, thus the first prototype of an e-WeddingThailand for couples and wedding organizations is quite significant and a promising channel in order to assist couples and wedding business suppliers to collaborate, search and recommend information. However there is much that could be done to broaden the functionality of the software. The domain ontology and web services ontology could be extended as a means to help with data handling. There could also be better data persistence added to the software, such as objects that could store information for each user. Security is one of the crucial aspects to emphasize for future work. Each agent would have to have its information secure, so that agents from outside could not access it. The user interface could be extended as well, to provide new suitable ways of handling the user data according to human computer interaction techniques.

Acknowledgments. This research is funded in part by Suan Sunandha Rajabhat University and e-WeddingThailand project.

References

1. Sandholm, T., Lesser: Advantages of a Leveled Commitment Contracting Protocol. In: Thirteenth National Conference on Artificial Intelligence (AAAI-1996), Portland, OR, pp. 126–133 (1996)

2. Klein, M., Dellarocas, C.: Towards a Systematic Repository of Knowledge about Managing Multi-Agent System Exceptions. Massachusetts Institute of Technology, Cambridge (2000)
3. Balachandran, B.M., Enkhsaikhan, M.: Developing Multi-agent E-Commerce Applications with JADE. In: Apolloni, B., Howlett, R.J., Jain, L. (eds.) KES 2007, Part III. LNCS (LNAI), vol. 4694, pp. 941–949. Springer, Heidelberg (2007)
4. Kimbrough, S.O., Wu, D., Zhong, F.: Computers play the Beer Game: Can artificial agents manage supply chains. *Decision Support Systems*, 323–333 (2002)
5. Teigen, R., Barbuceanu, M.: The supply chain demonstrator Technical report and user guide by the Enterprise Integration Laboratory. Department of Industrial Engineering, University of Toronto, Ontario, Canada (1996)
6. Maturana, F., Shen, W., Norrie, D.: Metamorph: an adaptive agent-based architecture for intelligent manufacturing. *International Journal of Production Research* 37, 2159–2173 (1999)
7. Cloutier, L., Frayret, J.-M., D'Amours, S., Espinasse, B., Montreuil, B.: A Commitment-Oriented Framework for Networked Manufacturing Co-ordination. *International Journal of Computer Integrated Manufacturing* 14(6), 522–534 (2001)
8. Lyonnais, P., Montreuil, B.: Un environnement distribué de simulation à base d'agents. In: 3e Conférence Francophone De Modélisation et SIMulation (MOSIM) Conception, Analyse et Gestion Des systèmes Industriels, Troyes, France (2001)
9. IBM, Research: BPMAT: A business process modeling and analysis tool, <http://www.research.ibm.com/pdtr/bpmat.htm>
10. Sadeh, N., Hildum, D., Kjenstad, D., Tseng, A.: MASCOT: An agent-based architecture for coordinated mixed-initiative supply chain planning and scheduling. In: 3rd International Conference on Autonomous Agents, Workshop Notes, Agent-Based Decision Support for Managing the Internet Enabled Supply Chain (1999)
11. Baumgaertel, H., Brueckner, S., Parunak, V., Vanderbok, R., Wilke, J.: Agent models of supply network dynamics. *The Practice of Supply Chain Management* (2001), <http://www.anteaters.net/sbrueckner/publications/2001/-AgentModelsOfSupplyNetworkDynamicssubmitted20010209.pdf>
12. Walsh, W.E.: Market protocols for decentralized supply chain formation. Ph.D. thesis, University of Michigan, Ann Arbor, Michigan, USA (2001)
13. Labarthe, O.: Modelisation multi-agent de chaînes logistiques. Mémoire. de Diplôme Etudes Approfondies, Université d'Aix-Marseille 3, Marseilles, France (2000)
14. Bournez, C., Gutknecht, O.: Modèle de contrôle par émergence de coordinations dans un réseau de contrats multiagents. In: Journées Francophones de l'Intelligence Artificielle Distribuée et des Systèmes MultiAgents, Montreal, Quebec, Canada (2001)
15. Abbas, K., Aïmeur, E., Kropf, P.: aKIA Automobile: a Multi-Agent System in E-Commerce. In: International Conference on Electronic Commerce Research (ICECR-5) (2010)
16. Somchart, F., Manpanpanich, P., Sekpon, J.: Multi-Application Authentication Based on Multi-Agent System. In: IMECS 2007, pp. 1316–1321 (2007)
17. Glaschenko, A., Ivaschenko, A., Rzevski, G., Skobelev, P.: Multi-Agent Real Time Scheduling System for Taxi Companies. In: Proc. of 8th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2009), Budapest, Hungary, May 10-15, pp. 29–36 (2009)
18. Negri, A., Poggi, A., Tomaiuolo, M., Turci, P.: Agents for e-Business Applications. In: Proceedings of the Fifth International Joint Conference on Autonomous Agents and Multiagent Systems, Hakodate, Japan (2006)
19. Haibo, Y., Tsunenori, M., Makoto, A.: A Multi-Agent-Based Semantic Web Service System. In: The International Symposium on Information Science and Electrical Engineering 2003 (ISEE 2003), Fukuoka, Japan, November 13-14, pp. 633–636 (2003)

20. Jennings, N.: Agent-based Software Engineering Artificial Intelligence, pp. 277–296 (2000)
21. Moreno, A.: Medical Applications of Multi-Agent Systems. Computer Science and Mathematics Department, University of Rovira, Spain (2003)
22. Stone, P., Veloso, M.: Multi-agent Systems: A Survey from a Machine Learning Perspective. *Autonomous Robots* 8(3), 345–383 (2008), <http://www.springerlink.com/index/W2671G4Q27327305.pdf>, doi:10.1023/A:1008942012299 (retrieved April 20, 2008)
23. Panait, L., Luke, S.: Cooperative Multi-Agent Learning: The State of the Art. *Autonomous Agents and Multi-Agent Systems* 11(3), 387–434 (2005)
24. Padghan, L., Winikopff, M.: Developing Intelligent AgentSystems. Wiley (2004)
25. JADE, Java Agent Development Environment, <http://jade.tilab.com>
26. Chinnici, R., Gudgin, M., Moreau, J.-J., Schlimmer, J., Weerawarana, S.: Web Services Description Language (WSDL) Version 2.0 Part 1: Core Language, W3C Working Draft (2003), <http://www.w3.org/TR/2003/WD-wsdl120-20031110/>
27. Gudgin, M., Hadley, M., Mendelsohn, N., Moreau, J.-J., Nielsen, H.: SOAP Version 1.2 Part 1: Messaging Framework, W3C Recommendation, <http://www.w3.org/TR/2003/REC-soap12-part1-20030624/>
28. Hendler, J.: Agents and the semantic web. *IEEE Intelligent Systems* 16(2), 30–37 (2001)
29. Fayyad, U.M., Pitatesky-Shapiro, G., Smyth, P., Uthurasamy, R.: Advances in Knowledge Discovery and Data Mining. AAAI/MIT Press (1996)
30. Frawley, W., Piatetsky-Shapiro, G., Matheus, C.: Knowledge Discovery in Databases: An Overview. *AI Magazine*, 213–228 (Fall 1992)
31. <http://www.infovis-wiki.net/index.php?title=Image:Fayyad96kdd-process.png>
32. Norwati, M., Manijeh, J., Mehrdad, J.: Expectation Maximization Clustering Algorithm for User Modeling in Web Usage Mining Systems. *European Journal of Scientific Research* 32(4), 467–476 (2009) ISSN 1450-216X
33. Jim, R.: Data Mining using the EM clustering algorithm on Places Rated Almanac Data, <http://www.galaxy.gmu.edu/stats/syllabi/inft979/RogersPaper.pdf>
34. MacQueen, R.: Some Methods for classification and Analysis of Multivariate Observations. In: Proceedings of 5-th Berkeley Symposium on Mathematical Statistics and Probability, vol. 1, pp. 281–297. University of California Press, Berkeley (1967)
35. Hsu, C.-W., Chang, C.-C., Lin, C.-J.: A Practical Guide to Support Vector Classification, <http://www.csie.ntu.edu.tw/~cjlin/papers/guide/guide.pdf>
36. Jakkula, V.: Tutorial on Support Vector Machine. School of. EECS, Washington State University (2006)
37. Kularbhetpong, k.: e-Negotiation based on Multi-agent ssytem. In: *JCSSE 2007 –The International Joint Conference on Computer Science and Software Engineer*, Thailand (2007)
38. Kularbhetpong, K., Clayton, G., Meesad, P.: e-Wedding Based on Multi-agent System. In: Demazeau, Y., Dignum, F., Corchado, J.M., Bajo, J., Corchuelo, R., Corchado, E., Fernández-Riverola, F., Julián, V.J., Pawlewski, P., Campbell, A. (eds.) *Trends in PAAMS. Advances in Intelligent and Soft Computing*, vol. 71, pp. 285–293. Springer, Heidelberg (2010)
39. Kularbhetpong, K., Clayton, G., Meesad, P.: A Hybrid System based on Multi-agent System in Data Preprocessing stage: e-Wedding. *International Journal of Computer Science and Information Security* (2010)
40. Cao, L., Gorodetsky, V., Mitkas, P.: Agent Mining: The Synergy of Agents and Data Mining. *IEEE Intelligent Systems* 24(3), 64–72 (2009)

Author Index

- Aouadi, Hamed 173
Arora, Monika 331
Atkinson, Katie 16
Attar, Vahida 156
- Ben Yahia, Sadok 173
Bhamra, G.S. 30
Bouchard, Guillaume 288
Brahmi, Imen 173
Byrd, Michael 4
- Cao, Jie 316
Cao, Longbing 46
Chaimontree, Santhana 16
Chan, Jonathan 4
Chatzidimitriou, Kyriakos C. 228
Chaudhari, Gaurish 156
Chaudhary, Prashant 156
Christensen, Dion 304
Chrysopoulos, Antonios C. 228
Clayton, Gareth 344
Coenen, Frans 16
Corniglion, Sébastien 58
- Emele, Chukwuemeka David 117
Epstein, Susan L. 132
- Fang, Changjian 316
Fiosina, Jelena 195
Fiosins, Maksims 195
- Gordon, Joshua 132
Goyal, Madhu 248
- Hansen, Henrik Ossipoff 304
Hernandez, Jorge Pablo Cordero 304
- Jansen, Chipp 4
Juul-Jensen, Lasse 304
- Kanjilal, Uma 331
Kaschesky, Michael 288
Kastaniegaard, Kasper 304
Kaur, Preetinder 248
Kowalczyk, W. 261
Kularbphettong, Kunyanuth 344
- Li, Gang 211
Ligorio, Tiziana 132
Lu, Jie 248
- Meesad, Phayung 344
Mitkas, Pericles A. 228
Moemeng, Chayapol 46
Müller, Jörg P. 195
- Neri, Filippo 86
Niu, Wenjia 211
Norman, Timothy J. 117
- Parsons, Simon 117
Passonneau, Rebecca 132
Patel, R.B. 30
Poncelet, Pascal 173
- Rahagude, Sonali 156
- Schut, M.C. 261
Şensoy, Murat 117
Shi, Zhongzhi 211
Sinha, Pradeep 156
Sklar, Elizabeth 4
Sobkowicz, Paweł 288
Stone, Peter 3
Symeonidis, Andreas L. 228
Szlávik, Z. 261
- Tournois, Nadine 58
- Varshney, Dinesh 331
Verma, A.K. 30
- Wang, Can 46
Wang, Xiaofeng 211
Wang, Yuanfei 98
Wu, Zhiang 316
- Yang, Xinghua 211
- Zeng, Yifeng 304
Zhang, Wei 98